ISSN 1213-6972 *Volume 30, Number 1-2, 2022*

Journal of WSCG

An international journal of algorithms, data structures and techniques for computer graphics and visualization, surface meshing and modeling, global illumination, computer vision, image processing and pattern recognition, computational geometry, visual human interaction and virtual reality, animation, multimedia systems and applications in parallel, distributed and mobile environment.

Vaclav Skala – Union Agency

Journal of WSCG

Editor-in-Chief: Vaclav Skala
c/o University of West Bohemia
Faculty of Applied Sciences
Univerzitni 8
CZ 306 14 Plzen
Czech Republic
http://www.VaclavSkala.eu

Managing Editor: Vaclav Skala

Printed and Published by:
Vaclav Skala - Union Agency
Na Mazinach 9
CZ 322 00 Plzen
Czech Republic

Published in cooperation with the University of West Bohemia
Univerzitní 8, 306 14 Pilsen, Czech Republic

Hardcopy: ***ISSN 1213 – 6972***
CD ROM: ***ISSN 1213 – 6980***
On-line: ***ISSN 1213 – 6964***

ISSN 1213-6972 *Volume 30, Number 1-2, 2022*

Journal of WSCG

An international journal of algorithms, data structures and techniques for computer graphics and visualization, surface meshing and modeling, global illumination, computer vision, image processing and pattern recognition, computational geometry, visual human interaction and virtual reality, animation, multimedia systems and applications in parallel, distributed and mobile environment.

Vaclav Skala – Union Agency

Journal of WSCG

Editor-in-Chief: Vaclav Skala
c/o University of West Bohemia
Faculty of Applied Sciences
Univerzitni 8
CZ 306 14 Plzen
Czech Republic
http://www.VaclavSkala.eu

Managing Editor: Vaclav Skala

Printed and Published by:
Vaclav Skala - Union Agency
Na Mazinach 9
CZ 322 00 Plzen
Czech Republic

Published in cooperation with the University of West Bohemia
Univerzitní 8, 306 14 Pilsen, Czech Republic

Hardcopy: ***ISSN 1213 – 6972***
CD ROM: ***ISSN 1213 – 6980***
On-line: ***ISSN 1213 – 6964***

Journal of WSCG

Editor-in-Chief

Vaclav Skala

c/o University of West Bohemia
Faculty of Applied Sciences
Department of Computer Science and Engineering
Univerzitni 8, CZ 306 14 Plzen, Czech Republic
http://www.VaclavSkala.eu

Journal of WSCG URLs: http://www.wscg.eu or http://wscg.zcu.cz/jwscg

Editorial Board

Journal of WSCG

Board of Reviewers

2022

Al-Darraji,S.(Iraq)
Baranoski,G.(Canada)
Barton,M.(Spain)
Baum,D.(Germany)
Benger,W.(Austria)
Bouatouch,K.(France)
Bourke,P.(Australia)
Burova,I.(Russia)
Cakmak,H.(Germany)
Campagnolo,L.Q(Brazil)
Carmen,M.J.(Spain)
Carmo,M.B.(Portugal)
Czapla,(Poland)
David,V.(France)
De Martino,J.M.(Brazil)
Delibasoglu,I.(Turkey)
Drakopoulos,V.(Greece)
Dziembowski,A.(Poland)
Elloumi,N.(Tunisia)
Galo,M.(Brazil)
Gdawiec,K.(Poland)
Giannini,F.(Italy)
Goncalves,A.(Portugal)
Grabska,E.(Poland)
Grajek,T.(Poland)
Gunther,T.(Germany)
Hast,A.(Sweden)
Heil,R.(Sweden)
Hu,C.(Taiwan)
Hu,P.(Belgium)
Chaudhuri,P.(India)
Jacek,K.(Poland)
Karim,S.A.A.(Malaysia)
Kerdvibulvech,C.(Thailand)
Klosowski,J.(United States)
Kuffner dos Anjos,R.(U.K.)
Kurt,M.(Turkey)
Lee,J.K.(United States)
Lefkovits,S.(Romania)
Lisowska,A.(Poland)
Liu,S.(China)
Lobachev,O.(Germany)
Marinkovic,V.(Serbia)
Marques,R.(Spain)
Max,N.(United States)
Meyer,A.(France)
Mieloch,D.(Poland)
Montrucchio,B.(Italy)
Nawfal,S.(Iraq)
Nguyen,S.(Viet Nam)
Norhaida,M.(Malaysia)
Oliveira,J.F.(Portugal)
Pagnutti,G.(Italy)
Pan,R.(China)
Papaioannou,G.(Greece)
Pedrini,H.(Brazil)
Perez,S.(Spain)
Phan,A.(Viet Nam)
Pintus,R.(Italy)
Pombinho,P.(Portugal)
Puig,A.(Spain)
Puppo,E.(Italy)
Raffin,R.(France)
Ramires Fernandes,A.(Portugal)
Reshetov,A.(United States)
Rodrigues,J.(Portugal)
Rodrigues,N.(Portugal)
Rojas-Sola,J.I.(Spain)
Romanengo,C.(Italy)
Sacco,M.(Italy)
Sardana,D.(United States)
Sarwas,G.(Poland)
Savchenko,V.(Japan)
Scaramuccia,S.(Italy)

Journal of WSCG

Vol.30, No.1-2, 2022

Contents

MVP-Net: Multiple View Pointwise Semantic Segmentation of Large-Scale Point Clouds 1
Chuanyu,L., Xiaohan,L., Nuo,C., Han,L., Shengguang,L., Pu,L.

Interactive High-Resolution Simulation of Granular Material 9
Sommer,A., Schwanecke,U., Schoemer,E.

Dynamic Sensor Matching based on Geomagnetic Inertial Navigation 16
Mueller,S., Kranzlmueller,D.

Exploring the necessity of mosaicking for underwater imagery semantic segmentation using deep learning 26
Buskus,K., Vaiciukynas,E., Medelyte,S., Siaulys,A.

Mip-NeRF RGB-D: Depth Assisted Fast Neural Radiance Fields 34
Dey,A., Ahmine,Y., Comport,A.I.

3D Point Set Registration based on Hierarchical Descriptors 44
Dutta,S., Russig,B.,Gumhold,S.

Spatiotemporal redundancy removal in immersive video coding 54
Dziembowski A., Mieloch D., Domanski M., Lee G., Jeong J.Y.

Uncertainty-aware Evaluation of Machine Learning Performance in binary Classification Tasks 63
Sperling,L., Lämmer,S., Hagen,H., Scheuermann,G., Gillmann,C.

Interactive Editing of Voxel-Based Signed Distance Fields 72
Wegen,O., Döllner,J., Trapp,M.

Magnitude of Semicircle Tiles in Fourier-space - A Handcrafted Feature Descriptor for Word Recognition using Embedded Prototype Subspace Classifiers 82
Hast,A.

Color-dependent pruning in immersive video coding 91
Mieloch,D., Dziembowski,A., Domanski,M., Lee,G., Jeong,J.Y.

Design Space of Geometry-based Image Abstraction Techniques with Vectorization Applications 99
Ihde,L., Semmo,A., Döllner,J. Trapp,M.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

MVP-Net: Multiple View Pointwise Semantic Segmentation of Large-Scale Point Clouds

Chuanyu Luo
LiangDao GmbH
Germaniastrasse 18-20
Germany 12099, Berlin
chuanyu.luo@liangdao.de

Xiaohan Li, Nuo Cheng,
Han Li, Shengguang Lei
LiangDao GmbH
Germaniastrasse 18-20
Germany 12099, Berlin
xiaohan.li, nuo.cheng,
han.li, sheng-
guang.lei@liangdao.de

Pu Li
Ilmenau University of
Technology
Ehrenbergstrasse 29
Germany 98693, Ilmenau
pu.li@tu-ilmenau.de

ABSTRACT

Semantic segmentation of 3D point cloud is an essential task for autonomous driving environment perception. The pipeline of most pointwise point cloud semantic segmentation methods includes points sampling, neighbor searching, feature aggregation, and classification. Neighbor searching method like K-nearest neighbors algorithm, KNN, has been widely applied. However, the complexity of KNN is always a bottleneck of efficiency. In this paper, we propose an end-to-end neural architecture, **M**ultiple **V**iew **P**ointwise Net, MVP-Net, to efficiently and directly infer large-scale outdoor point cloud without KNN or any complex pre/postprocessing. Instead, assumption-based space filling curves and multi-rotation of point cloud methods are introduced to point feature aggregation and receptive field expanding. Numerical experiments show that the proposed MVP-Net is 11 times faster than the most efficient pointwise semantic segmentation method RandLA-Net [Qin20a] and achieves the same accuracy on the large-scale benchmark SemanticKITTI dataset.

Keywords

Point Cloud, Semantic Segmentation, Space Filling Curves, Convolutional Neural Networks

1 INTRODUCTION

Lidar is widely used in autonomous driving perception systems. The 3D point cloud captured from Lidar provides important geometric information for complex environment perception tasks like object detection and semantic segmentation.

Unlike the regular structured images in computer vision, point cloud is irregular and unordered, and the outdoor large-scale point cloud is sparse. To overcome these challenges, most researchers transform the irregular point cloud to regular projection-based images or 3D voxels. Although these approaches can achieve satisfactory results, there is information loss in 3D-2D projection-based methods. In voxelization methods, the preprocessing is expensive and the computational and memory cost increases cubically by the increase of resolution [Yul20a]. In addition, when applying small resolution and large voxel size, the specific granular point features are ignored.

The PointNet [Cha17a] is a pioneering pointwise network that directly processes point cloud and outputs the label per point. The key contribution of PointNet is to introduce the single symmetric function, max pooling, to aggregate the global features from unordered point clouds. However, the PointNet is limited to small point clouds and cannot be extended to large-scale point clouds [Qin20a]. The reason is the learned global features by PointNet cannot represent a large-scale point cloud consisting of many objects and complex structures.

Recently, PointNet-based works [Cha17b, Qin20a] were proposed to directly process large-scale point clouds. These pipelines include multi-level point cloud sampling, neighbor searching, and PointNet-based local feature aggregation. RandLA-Net [Qin20a] achieves enhanced performances by efficient point cloud random sampling. However, the pointwise methods do not have explicit point neighboring information, and the time-consuming methods like KNN and ball query [Yul20a] are still applied to neighbor searching.

For computer vision semantic segmentation tasks, features aggregation of pixel does not require any extra

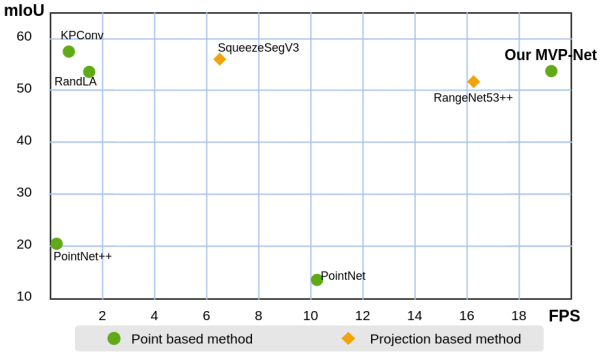


Figure 1: Quantitative comparison results of our approach and other works on the SemanticKITTI dataset.

Layer Type	Complexity per Layer
1D Convolution	$O(M \cdot K \cdot C_{in} \cdot C_{out})$
2D Convolution	$O(M^2 \cdot K^2 \cdot C_{in} \cdot C_{out})$
3D Convolution	$O(M^3 \cdot K^3 \cdot C_{in} \cdot C_{out})$

Table 1: Convolutional layer type and complexity per layer. M is the output feature map size. K is the kernel size, C_{in} is the input channel size, and C_{out} is the output channel size. The output feature map size M can be different between different layer type. The 1D convolution layer could be more efficient because it reduces the complexity caused by kernel size K

neighbor searching step, and it is replaced by applying a 2D convolutional layer. The reason for this simplicity is due to the fact that the structure of an image is regular. The most simple strategy to make point cloud regular is to sort the points, and space filling curves is one sorting method to map high dimensional data to 1D sequence, while preserving the 3D local neighbor information. The preserved local information can be further applied to points feature aggregation and prediction. Details of space filling curves can be found in section 2.4.

Therefore, our basic idea is the 3D Lidar points can be mapped to 1D sequence data, and the proposed 1D convolutional neural network can predict the points by the preserved points local features. As high dimensional neighbor searching method like KNN is deprecated, and high efficient 1D convolution layer is used, our proposed model is more efficient than the other similar benchmark methods by a great margin (see Fig.1). The efficiency of 1D convolution layer and the complexity comparison of other layers can be found in Tab.1.

2 RELATED WORK

In this section, we give an overview of deep learning based approaches in point cloud semantic segmentation tasks, mainly including voxelization based methods, projection based methods, and point based methods. Besides, the sorting algorithm space filling curves (SFC) will also be introduced.

2.1 Voxelization based methods

Voxelization based methods [Xin21a, Yan20a, Jia21a, Hao20a] transform the irregular unordered point cloud into regular 3D grids, and then the powerful 3D convolution is applied in feature extraction and prediction. However, the problem of granular information loss can be caused by using a large voxel size. Meanwhile, the computational and memory cost increases cubically by the increase of the resolution, especially when processing large-scale outdoor sparse point cloud [Yul20a]. To tackle the granular information loss caused by a large voxel size and the expensive cost caused by a small voxel size, the fusion of a point based method and sparse 3D convolution can be a feasible solution.

2.2 Projection based methods

To leverage the success from 2D image processing neural networks, the projection based methods [Che20a] project the 3D point cloud into 2D images, and then the traditional 2D architecture like U-net [Ola15a] is applied for features aggregation. However, the primary limitation lies in the information loss from 3D-2D projection [Yul20a].

2.3 Point based methods

Inspired by the pioneering work PointNet, many works [Cha17b, Qin20a] were introduced by directly taking the point cloud as input and learning the pointwise features by multi-layer perception and symmetric function like max pooling.

To directly process large-scale outdoor point cloud, in most pointwise MLP based works [Cha17b, Qin20a] the pipeline includes sampling, neighbor points searching such as ball query or K-nearest-neighbors (KNN), PointNet-based features extraction, and per-point classification. Recently, RandLA-Net[Qin20a] was developed to utilize random point sampling and showed impressive results. However, for carrying out the neighbor points searching task, RandLA-Net still utilizes the KNN method, which limits its efficiency even using the KDTree [Jon75a] algorithm with complexity $O(nlog(n))$.

KPConv[Hug19a] is a kernel based point convolution method, which in essence takes the neighbor points as input and processes the points with spatial kernel weights. However, in this method, the radius neighborhood approach is applied to neighbor searching, and the network cannot directly train the entire data of large scenes [Jia21a].

2.4 Space filling curves

Space filling curves (SFC) is a sorting method to map high dimensional data to one dimension sequence, while preserving locality of the data points (e.g. the Euclidean neighbor similarity in 3D tends to be kept after

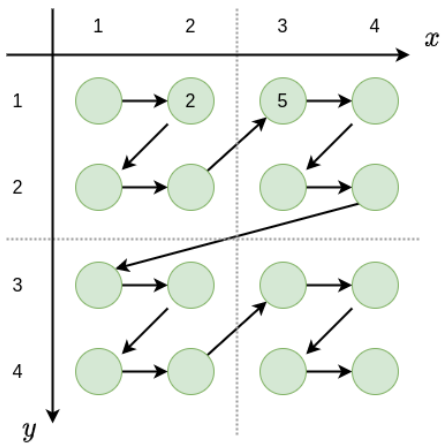


Figure 2: Morton-order in 2D case. The 2D points is sorted while the locality is preserved, but there is still information loss, like the distance between point 2 and 5 in 2D and 1D case is different.

mapped to 1D) [Tha20a]. One of widely applied and high efficient SFC methods is Morton-order [Mor66a], also known as Z-order because of the shape of the curve in the 2D case, e.g. the curve in Fig.2. The other SFC methods [Val05a] include Hilbert, Peano, Sierpinski curves, etc.

In point cloud field, MortonNet [Tha20a] uses Morton-order in self-supervised tasks, which not require semantic labels. The trained MortonNet predicts the next point in a point sequence created by Morton-order, and the results show the learned Morton features can be transferred to semantic segmentation tasks and improve performances. However, like PointNet, MortonNet is limited to small-scale point clouds.

Though SFC can keep the local features of high dimensional data after mapping, there is still information loss. As shown in Fig.2, the distance between point 2 and point 5 in 2D space is 1, but after mapping to 1D sequence, the distance in 1D space is 3. The point 5 might not be considered as the neighbor of point 2 after mapping, which limits the network points feature aggregation and prediction ability.

3 MVP-NET

In this section, we first present our key contribution, the assumption-based space filling curves for 3D point cloud, and the expansion of receptive field per point by rotating the raw points, and at last an overview of our network architecture MVP-Net, **M**ultiple **V**iew **P**ointwise Net, and the implementation details.

3.1 Space filling curves for 3D point cloud

Space filling curves (SFC) are widely applied to the regular dense data like images, matrices and grid cells [Val05a]. To obtain the SFC of large-scale

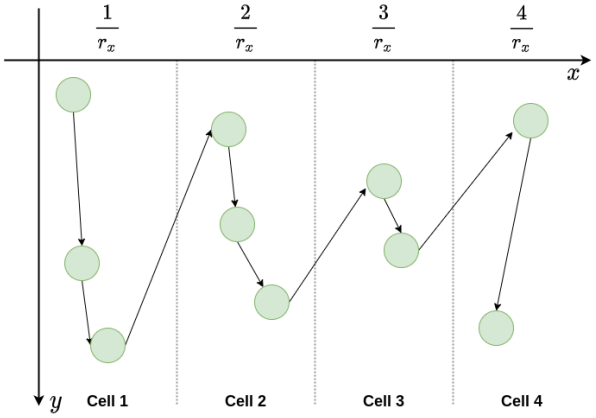


Figure 3: Space filling curves for 2D point cloud by simple scores Eq.1

irregular sparse point cloud, we propose the Eq.8 to calculate the score per point, and sort the points order by the calculated scores.

For illustration purpose, the most simple case of 2D points and scores per point is expressed as Eq.1,

$$scores = k_x \cdot round(x \cdot r_x) + y \tag{1}$$

where x, y denote the coordinates of 2D point, $round()$ is the rounding function to find the nearest integer of the input. The key goal is to sort the points by cells along x axis, and in each cell the points will be sorted along y axis, as shown in Fig.3. The points in Fig.3 will be sorted by the scores in Eq.1 in ascending order. r_x and k_x are two hyperparameters to make the points strictly sorted.

The cell width is $\frac{1}{r_x}$, it can be derivated by inequation 2 and 3

$$-0.5 \leq x \cdot r_x - x_R < 0.5 \tag{2}$$

$$\rightarrow \quad \frac{x_R}{r_x} - \frac{0.5}{r_x} \leq x < \frac{x_R}{r_x} + \frac{0.5}{r_x} \tag{3}$$

where x_R is the nearest integer of $x \cdot r_x$, i.e. $x_R = round(x \cdot r_x)$.

To make sure the points are sorted strictly cell by cell, the condition that the last point score in cell i is less than the first point score in cell $i+1$ should be always satisfied.

If the last point coordinate in cell i is denoted as (x_i, y_i), and the first point in next cell $i+1$ is (x_{i+1}, y_{i+1}), the rounded values along x can be denoted as $x_R^i = round(x_i \cdot r_x)$ and $x_R^{i+1} = round(x_{i+1} \cdot r_x)$. Since cell width is $\frac{1}{r_x}$, the rounded values satisfy $x_R^{i+1} - x_R^i = \frac{1}{r_x}$.

To make sure the last point scores in cell i should be less than the first point scores in cell $i+1$, also $k_x \cdot x_R^i + y_i < k_x \cdot x_R^{i+1} + y_{i+1}$, the hyperparameter k_x should satisfy

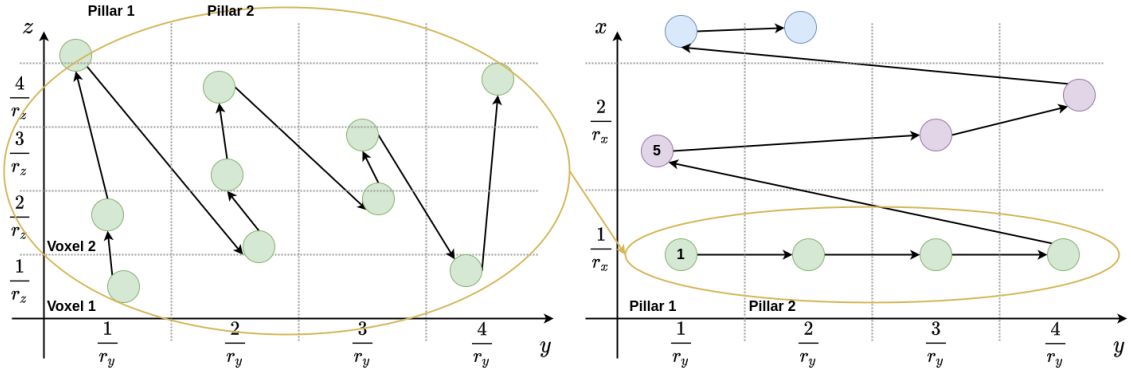


Figure 4: Space filling curves for 3D point cloud by scores Eq.8

$$k_x \cdot x_R^i + y_i < k_x \cdot x_R^{i+1} + y_{i+1} \tag{4}$$

$$\rightarrow \quad k_x > \frac{y_i - y_{i+1}}{x_R^{i+1} - x_R^i} = (y_i - y_{i+1}) \cdot r_x \tag{5}$$

$$\rightarrow \quad k_x > (y_{max} - y_{min}) \cdot r_x, \tag{6}$$

where y_{max} and y_{min} is maximal and minimal coordinate y values in all point clouds. In practice, we focus on the region of interest points, e.g. in a rectangular region of $X_{min} < x < X_{max}$, and $Y_{min} < y < Y_{max}$. As a result, the hyperparameter k_x should meet the condition 7

$$k_x > (Y_{max} - Y_{min}) \cdot r_x. \tag{7}$$

The hyperparameter r_x, which determines the cell size, will be extended and discussed in the full sorting equation.

Based on Eq.1, Fig.3 shows that the points will be sorted at first along y axis, and then cell by cell along x axis, as the x item will contribute more scores after multiplying k_x.

To introduce the complete space filling curves of 3D point cloud in an autonomous driving scene, at first we define two assumptions to aggregate the most important point cloud neighboring features for prediction,

1. For neighbor-points searching, the neighbor per point should be the same object as the point to be classified as much as possible.

2. In an outdoor autonomous driving scene, the points, except ground points, along object height z axis are more possible to belong to the same object than those along the other direction.

The first assumption can be explained by an example that a point is classified as a car is based on the information of the other points on the same car, instead of the other object points, even when the other object points are more spatially close to the to be classified point. The second assumption can be explained by standing object examples like cars, trees, buildings, etc. Except for standing object points, the ground points have distinguishing features and thus can be easily predicted.

From the two assumptions above, we extend Eq.1 to Eq.8, in which the feature along the object height axis is more important, and the points along z axis in each pillar will be at first aggregated.

$$\begin{aligned} scores = {} & k_x \cdot round(x \cdot r_x) + k_y \cdot round(y \cdot r_y) \\ & + k_z \cdot round(z \cdot r_z) + k_\rho \cdot \rho \end{aligned} \tag{8}$$

where

$$\rho = \sqrt{x^2 + y^2} \tag{9}$$

where x, y, z denote the coordinates of each point, $k_x \gg k_y \gg k_z \gg k_\rho$. The hyperparameter r_x and r_y determine the pillar size. Small pillar size will only contain few points in each pillar, and lose feature information. Large pillar size is against the first assumption, because the pillar will include different object points. These parameters, including the pillar size, are set empirically in implementation as Tab.2.

From the discussion in the simplified sorting Eq.1, the 3D points will be sorted at first along z axis voxel by voxel, and then along y axis and at last along x axis pillar by pillar. The space filling curve, as shown in Fig.4, derivated by Eq.8, preserve the most useful local features after mapping from 3D data to 1D sequence.

The complexity of KNN, applied in 3D case, is $O(nlog(n))$ when applying the KDTree [Jon75a] algorithm, but the complexity of the proposed sorting function Eq.8 is only $O(n)$.

3.2 Point cloud rotation

Though the assumption-based sorting Eq.8 keeps the most meaningful features along the height z axis, there is still information loss caused by SFC, which is introduced in section 2.4. In Fig.4, e.g. the pillar points 5

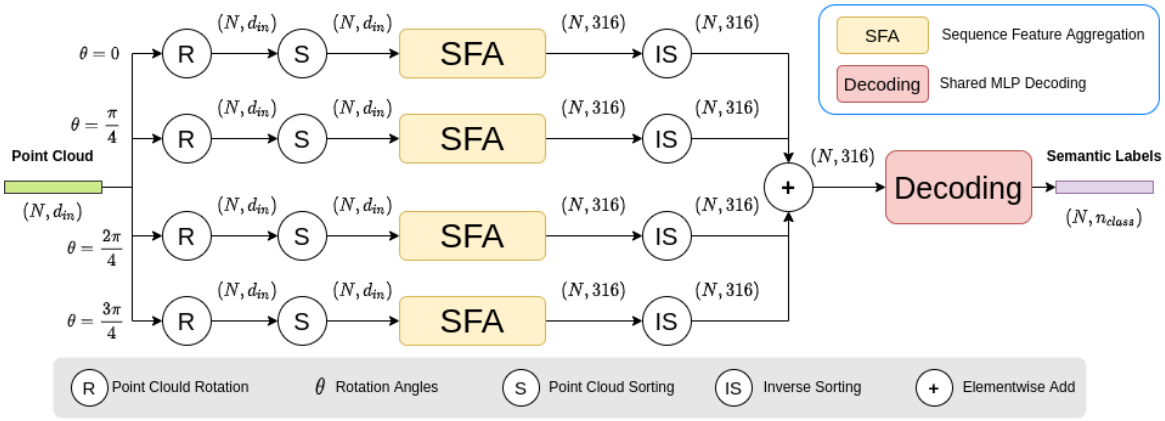


Figure 5: The architecture of the MVP-Net. N represents the number of points. d_{in} denotes the input features of point including the coordinates and other features. n_{class} is the number of classes to be predicted. The point cloud rotation expands the receptive field per point. The point sorting makes the point sequence regular, and SFA can directly aggregate the neighbor information. After inversely sorted to the original order, the aggregated features will be decoded by a share-MLP based network.

Parameter	value
k_x	10^{10}
k_y	10^5
k_z	10^0
k_ρ	10^{-5}
r_x	1.2
r_y	1.2
r_z	4

Table 2: The parameters and values of sorting Eq.8 in implementation

is the neighbor of pillar points 1 in 3D space, but the distance is much longer in 1D space. Therefore, the receptive field per point in 1D is limited. To solve this problem, the raw point cloud will be rotated along z axis multiple times in our work, and the idea is similar to the multiple view in projection methods.

If sorting scores Eq.8 is applied to the rotated by angle $\frac{\pi}{2}$ point cloud, it is equivalent applying Eq.10, where $k_y \gg k_x \gg k_z \gg k_\rho$ and the direction along x and y axis is exchanged compared with Eq.8, to the unrotated point cloud. In this case, the pillar points 5 is the neighbor of pillar 1. In our proposed network, the raw point cloud will be rotated by 4 different angles, and by applying only Eq.8, the neighbor and receptive field per point will be expanded.

$$scores = k_y \cdot round(y \cdot r_y) + k_x \cdot round(x \cdot r_x) + k_z \cdot round(z \cdot r_z) + k_\rho \cdot \rho \quad (10)$$

3.3 Architecture

The architecture of MVP-Net is illustrated in Fig.5. The raw point cloud is at first rotated along the z axis by 4 fixed angles, 0, $\frac{\pi}{4}$, $\frac{2\pi}{4}$, $\frac{3\pi}{4}$. The rotated point clouds are then sorted by Eq.8. After being sorted, the point cloud is regular and has explicit neighbor information.

The sequence feature aggregation backbone SFA, **S**equence **F**eature **A**ggregation, as shown in Fig.6, is a 1D convolutional network to aggregate the local and global features per point.

For each point i, the position difference between p_i and its nearest 8 points $\{p_i^1 ... p_i^k ... p_i^8\}$ along the sorted sequence are explicitly encoded to achieve the translation invariance [Wen19a] of the point cloud as follows:

$$x_i = p_i \oplus (p_i - p_i^k) \oplus p_i^{other\ features} \quad (11)$$

where p_i and p_i^k are the xyz coordinates of the point i and the neighbor point. $p_i^{other\ features}$ represents the other features except xyz positions of the point i, e.g. the intensity of reflection per point. $\oplus$ is the concatenation operation. The features extracted by SFA will be inversely sorted by the sorting function to the original order. The inverse sorted features per point from different angles will be added and then decoded by a multi-layer perception based network with a kernel size of 1.

3.4 Implementation

We implement our network in Pytorch [Ada19a]. For batch training, we sample the input cloud to 10^5 points per frame. We use the Adam optimizer [Die15a] with default parameters and the learning rate is set as 0.0003 without any decay. The most commonly used cross-entropy loss is employed as the loss function at first.

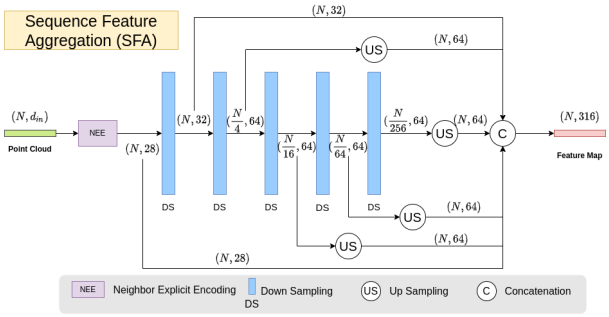


Figure 6: Sequence Feature Aggregation (SFA). SFA is a network consisting of only 1D convolution layer for regular sequence point cloud feature aggregation.

When the network optimization steps into a plateau, we adapt the loss function as

$$\text{Loss} = \text{Cross-entropy-loss} + \text{Lovász-softmax-loss} \tag{12}$$

The Lovász-Softmax-loss [Max18a] is a differentiable loss function designed to maximize the mean intersection-over-union (mIoU) score directly, which is commonly employed in the evaluation of semantic segmentation tasks. For data augmentation, we randomly rotate the original point cloud along z axis at each training step. For validation and testing, we input the whole original point cloud into the network and infer the semantic labels per point without any pre/postprocessing. All experiments are conducted on a Tesla V100 with one GPU 32G.

4 EXPERIMENTS

4.1 Dataset

We trained and tested our MVP-Net on the SemanticKITTI [Jen19a] dataset which provides large-scale semantic annotation per point in autonomous driving scenes. The point annotations for individual scan of the sequence 00-10 are provided. We used sequence 08 as a validation set and the other 10 sequences as a training set. Each point includes the information of the three dimensional coordinates and the remission. The mean intersection-over-union (mIoU) over 19 classes is used as the standard metric.

4.2 Results

Fig.7 presents a qualitative result of the prediction and the ground truth on the validation set.

Tab.3 presents a quantitative comparison of our MVP-Net with the other recently published methods. These methods are grouped by point-based [Cha17a, Cha17b, Qin20a, Hug19a], projection-based [Che20a, And19a], voxelization-based [Xin21a, Yan20a], and fusion-based methods [Hao20a, Jia21a].

The comparison demonstrates that the voxelization-based methods surpass the point-based and projection-based methods on accuracy with large margin. The fusion-based method RPVNet [Jia21a], the fusion of point, projection and voxelization method, ranks 1st of accuracy without much surprise.

Our proposed MVP-Net achieves the highest efficiency over all the other methods. As one point-based method, MVP-Net also achieve the comparable accuracy, and the same mIoU as the pointwise benchmark method, RandLA-Net [Qin20a], which uses KNN searching for point neighbor in 3D space. This result shows that the 3D points local feature can be preserved by space filling curves. The general information loss problem is solved by our assumption-based curves and point cloud rotation. The proposed contributions will also be proved in ablation study section 4.3.

4.3 Ablation Study

All the ablated networks are trained by the sequence 00-07 and 09-10, and evaluated by the sequence 08 of the SemanticKITTI dataset.

1. **Increasing/decreasing point cloud rotation times**. To achieve an excellent trade-off between efficiency and effectiveness, the number of point cloud rotation along z axis is studied.

2. **Removing the point neighbor explicit encoding (NEE) in the SFA unit**. To achieve the translation-invariance of the point cloud, the xyz positions difference of the center point and its neighbor point are explicitly encoded in the SFA unit. The point neighbor explicit encoding step is removed in the ablation study, and the original point features are directly fed into the SFA unit.

3. **Replace the assumption-based sorting function by another function**. The assumption based sorting function is one key contribution in our study, and we think the feature along object height axis is more important. Here we compare the original sorting Eq.8 with Eq.13, which will not preferentially aggregate the points along z axis.

$$\begin{aligned} scores = k_z \cdot round(z \cdot r_z) + k_x \cdot round(x \cdot r_x) \\ + k_y \cdot round(y \cdot r_y) + k_\rho \cdot \rho \end{aligned} \tag{13}$$

The parameters of Eq.13 meet $k_z \gg k_x \gg k_y \gg k_\rho$, and the other parameters are the same as the original sorting Eq.8. All results are listed in Tab.4. It can be seen that the proposed parameters and methods achieve the best trade-off between efficiency and effectiveness. The proposed assumption-based SFC by Eq.8 also surpasses the normal SFC, e.g. Eq.13.

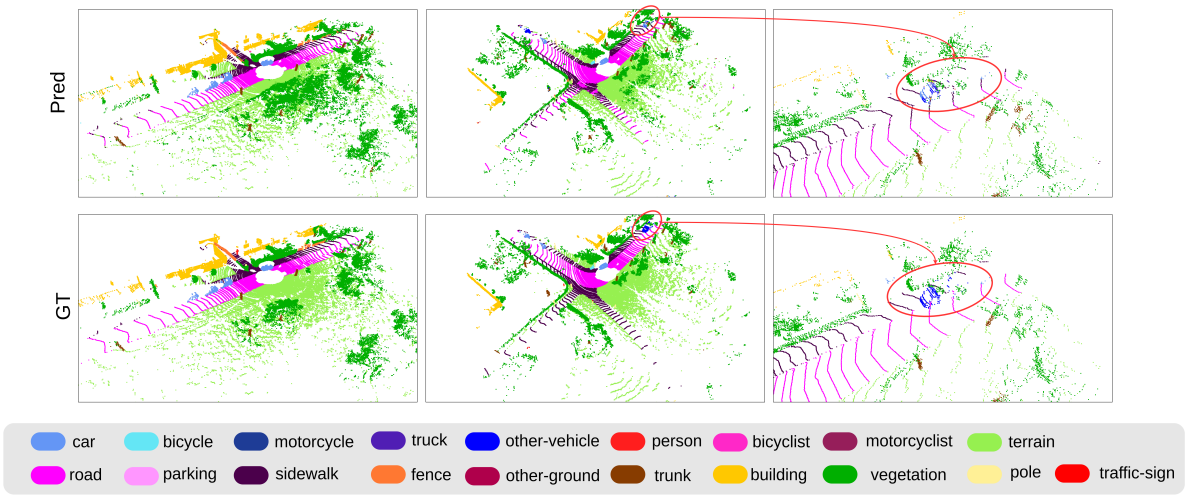


Figure 7: Qualitative comparison of prediction and ground truth on the validation set. The red circle shows the failure case of other-vehicle points classification.

Method	FPS	mIoU(%)	road	sidewalk	parking	other-ground	building	car	truck	bicycle	motorcycle	other-vehicle	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic-sign
PointNet [Cha17a]	10.12	14.6	61.6	35.7	15.8	1.4	41.4	46.3	0.1	1.3	0.3	0.8	31.0	4.6	17.6	0.2	0.2	0.0	12.9	2.4	3.7
PointNet++ [Cha17b]	0.06	20.1	72.0	41.8	18.7	5.6	62.3	53.7	0.9	1.9	0.2	0.2	46.5	13.8	30.0	0.9	1.0	0.0	16.9	6.0	8.9
RandLA [Qin20a]	1.74	53.9	90.7	73.7	60.3	20.4	86.9	94.2	40.1	26.0	25.8	38.9	81.4	61.3	66.8	49.2	48.2	7.2	56.3	49.2	47.7
KPConv [Hug19a]	0.88	58.8	88.8	72.7	61.3	31.6	90.5	96.0	33.4	30.2	42.5	44.3	84.8	69.2	69.1	61.5	61.6	11.8	64.2	56.4	47.4
SqueezeSegV3 [Che20a]	6.49	55.9	91.7	74.8	63.4	26.4	89.0	92.5	29.6	38.7	36.5	33.0	82.0	59.4	65.4	45.6	46.2	20.1	58.7	49.6	58.9
RangeNet53++ [And19a]	16.12	52.2	91.8	75.2	65.0	27.8	87.4	91.4	25.7	25.7	34.4	23.0	80.5	55.1	64.6	38.3	38.8	4.8	58.6	47.9	55.9
Cylinder3D [Xin21a]	2.55	67.8	91.4	75.5	65.1	32.3	91.0	97.1	59.0	67.6	64.0	58.6	85.4	71.8	71.8	73.9	67.9	36.0	66.5	62.6	65.6
PolarNet [Yan20a]	1.96	54.3	90.8	74.4	61.7	21.7	90.0	93.8	22.9	40.3	30.1	28.5	84.0	61.3	65.5	43.2	40.2	5.6	61.3	51.8	57.5
SPVNAS [Hao20a]	0.88	66.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
RPVNet [Jia21a]	-	**70.3**	93.4	80.7	70.3	33.3	93.5	97.6	44.2	68.4	68.7	61.1	86.5	75.1	71.7	75.9	74.4	73.4	72.1	64.8	61.4
MVP-Net	**19.20**	53.9	91.4	75.9	61.4	25.6	85.8	92.7	20.2	37.2	17.7	13.8	83.2	64.5	69.3	50.0	55.8	12.9	55.2	51.8	59.2

Table 3: The comparison results of our network and the other recently published point-based, projection-based, voxelization-based and fusion-based methods on SemanticKITTI dataset. The input of points is fixed as 10^5, and the size of projection image and voxel is set the same as literature. FPS, frames per second, represents the inference speed. mIoU, intersection-over-union, is the accuracy metric over all classes.

Ablation Experiment	FPS	mIoU
Rotating the point cloud once	55.2	47.4
Rotating the point cloud twice	33.2	51.7
Rotating the point cloud eight times	10.5	54.8
Removing the NEE in SFA unit	19.8	50.2
Sorting by Eq.13	19.2	43.7
Original framework	19.2	54.6

Table 4: Ablation experiments based on the original framework

5 CONCLUSION

In this paper, we present a novel end-to-end pointwise network MVP-Net for 3D point cloud semantic segmentation tasks. In contrast to the existing pointwise methods, our network shows the possibility to totally remove the KNN or other neighbor searching methods. Instead, assumption-based space filling curves and point cloud rotation from multiple angles are proposed to achieve comparable accuracy and high efficiency.

Our model is a fully 1D convolutional architecture without any pre/postprocessing, and therefore, the state-of-the-art neural modules from computer vision and natural language processing can be directly imported into our model. The fusion with other types of methods like voxelization will also be explored in future work.

As far as we know, our method is the first to directly apply the space filling curves to large-scale point cloud, and our experiments prove that the local information loss problem caused by SFC can be solved in point cloud field. The defined two assumptions are explained and proved by the experiments and ablation study, which is important for rethinking the features aggregation in autonomous driving scene.

6 REFERENCES

[Qin20a] Qingyong H., Bo Y., Linhai X., Stefano R., Yulan G., Zhihua W., Niki T., and Andrew M.. Randla-net: Efficient semantic segmentation of large-scale point clouds. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[Jon75a] Jon Louis B.. Multidimensional binary search trees used for associative searching. Communications of the ACM, 18(9):509–517, 1975.

[Max18a] Maxim B., Amal Rannen T., and Matthew B B.. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4413–4421, 2018.

[Yul20a] Yulan G., Hanyun W., Qingyong H., Hao L., Li L., and Mohammed B.. Deep learning for 3d point clouds: A survey. IEEE transactions on pattern analysis and machine intelligence, 2020.

[Die15a] Diederik P. K. and Jimmy Ba. A.. A method for stochastic optimization. 3rd International Conference on Learning Representations, ICLR 2015.

[Ada19a] Adam P., Sam G., Francisco M., Adam L., James B., Gregory C., Trevor K., Zeming L., Natalia G., Luca A., et al. Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems, 32:8026–8037, 2019.

[Cha17a] Charles R. Q., Hao S., Kaichun M., and Leonidas J. G.. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[Jen19a] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In Proc. of the IEEE/CVF International Conf. on Computer Vision(ICCV), 2019.

[Cha17b] Charles, Ruizhongtai Q., Li Y., Hao S., and Leonidas J G.. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017.

[Ola15a] Olaf R., Philipp F., and Thomas B.. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention, pp. 234–241. Springer, 2015.

[Hao20a] Haotian T., Zhijian L., Shengyu Z., Yujun L., Ji L., Hanrui W., and Song H.. Searching efficient 3d architectures with sparse point-voxel convolution. In European Conference on Computer Vision, 2020.

[Hug19a] Hugues T., Charles R Qi, Jean-Emmanuel D., Beatriz M., François G., and Leonidas J G.. Kpconv: Flexible and deformable convolution for point clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6411–6420, 2019.

[Wen19a] Wenxuan W., Zhongang Q., and Li F.. Pointconv: Deep convolutional networks on 3d point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9621–9630, 2019.

[Che20a] Chenfeng X., Bichen W., Zining W., Wei Z., Peter V., Kurt K., and Masayoshi T.. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. In European Conference on Computer Vision, pp. 1–19. Springer, 2020.

[Jia21a] Jianyun X., Ruixiang Z., Jian D., Yushi Z., Jie S., and Shiliang P.. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation. CoRR, abs/2103.12978, 2021.

[Yan20a] Yang Z., Zixiang Z., Philip D., Xiangyu Y., Zerong X., Boqing G., and Hassan F.. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9601–9610, 2020.

[Xin21a] Xinge Z., Hui Z., Tai W., Fangzhou H., Yuexin M., Wei L., Hongsheng L., and Dahua L.. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9939–9948, June 2021.

[Tha20a] Thabet A., Alwassel H., and Ghanem B.. Self-Supervised Learning of Local Features in 3D Point Clouds. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).

[Mor66a] Morton, G.M. A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing. 1966.

[Val05a] Valgaerts L. Space-filling curves an introduction. Technical University Munich. 2005 Apr.

[And19a] Andres M., Ignacio V., Jens B., Cyrill S.. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 2019.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Interactive High-Resolution Simulation of Granular Material

Alexander Sommer[1]
alexander.sommer@hs-rm.de

Ulrich Schwanecke[1]
ulrich.schwanecke@hs-rm.de

Elmar Schoemer[2]
schoemer@uni-mainz.de

[1] Computer Vision and Mixed Reality Group, RheinMain University of Applied Sciences Wiesbaden Rüsselsheim, Germany
[2]Institute of Computer Science, Johannes Gutenberg University Mainz, Germany

ABSTRACT

We introduce a particle-based simulation method for granular material in interactive frame rates. We divide the simulation into two decoupled steps. In the first step, a relatively small number of particles is accurately simulated with a constraint-based method. Here, all collisions and the resulting friction between the particles are taken into account. In the second step, the small number of particles is significantly increased by an efficient sampling algorithm without creating additional artifacts. The method is particularly robust and allows relatively large time steps, which makes it well suited for real-time applications. With our method, up to 500k particles can be computed in interactive frame rates on consumer CPUs without relying on GPU support for massive parallel computing. This makes it well suited for applications where a lot of GPU power is already needed for render tasks.

Keywords: position-based dynamics, position-based simulation, real-time simulation, animation

1 INTRODUCTION

Granular materials are composed of many, small bodies that can be clearly separated from each other. The individual components of the granular material can be very different, for example grains, sand, gravel, rubble, but also beans, rice and much more.

The simulation of such materials is particularly challenging, since the macroscopic behavior of the material is determined by the microscopic interactions between the individual grains. A complete, physically correct description of all interactions is virtually impossible, so simplified models are used that accurately reflect reality to some degree. For this purpose, various methods have been developed in the field of civil engineering and later also in the field of computer graphics. While methods from the engineering field must be able to simulate acting forces as accurately as possible, the main focus of computer graphics methods is on generating visually plausible results.

A common problem with current simulation methods in computer graphics is that, despite considerable simplification, it is not possible to achieve interactive frame rates at higher particle counts that are necessary for plausible visualization. In this work we use a position-based simulation [4] with a low particle count that can be easily computed within interactive frame rates and then refine it with an efficient upsampling algorithm to obtain a large particle count. The two steps of the simulation are decoupled from each other and can be computed at different temporal resolutions. The accurate behavior of the continuum is determined by calculating the individual collisions in the first step of the simulation. The behavior of the high-resolution particles in the second step is ensured by interpolating the underlying velocity field and by partially blending in external forces.

2 RELATED WORK

In the field of computer science, the physical simulation of granular material plays an important role, besides physical computing, especially in computer graphics. A distinction is made between continuum methods and purely particle-based discrete methods.

Continuum methods are particularly well suited to simulate granular flow in the most time-effective manner, since there is a decoupling between grain size and resolution of the simulation. This decoupling is at the expense of finer details in areas where the motion is in free flow, for example at surfaces, free-falling grains or in the formation of spatter. Zhu and Bridson [29] used a hybrid Euler-Lagrangian formulation of the Fluid-Implicit (FLIP) method, treating sand as a fluid. Narain et. al. [22] developed a continuum-based model that efficiently calculates internal pressures and frictional stresses with an unilateral incomprehensibility constraint. Their method also allows two-way coupling with rigid-bodies. Lenaerts and Dutré [17] make

use of a pure Lagrangian approach by simulating granular material with the Smoothed Particle Hydrodynamics (SPH) method [19]. Alduán and Otaduy [1] incorporated Narain et. al.'s unilateral incomprehensibility within the predictive-corrective incompressible SPH (PCISPH) method [24]. Klar et. al. use the Drucker-Pager plastic flow model to simulate sand in the Material Point Method (MPM) [26][16]. Hu et. al. introduce the Moving Least Squares Material Point Method (MLS-MPM) [13] to enable the simulation of new phenomena in MPM like two-way coupling with rigid-bodies.

Discrete methods, in contrast to continuum methods, simulate the macroscopic behavior of the material based on contacts and collisions between individual grains or particles. This enables realistic modeling of various physical phenomena. However, accurate simulation of granular media often requires a very small grain size or particle radius, and thus a large number of particles. Collision detection of a large number of particles is computationally intensive. A small particle radius usually also requires an increase of the time steps in the simulation. In practice, therefore, often unnaturally large grain sizes have to be used. The first approaches for simulating granular material with discrete methods by Cundall and Strack [8] stem from Discrete Element Method (DEM) theory in molecular dynamics. Later Bell et. al. [3] model granular material as non-spherical particles following DEM principles with contact and shear forces for animation purposes. Müller et. al. [20] developed position-based dynamics (PBD), a particle-based simulation framework that applies positional changes of particles directly to the position layer without calculating forces between individual particles. Macklin et. al. [18] developed a static and dynamic friction model for PBD to mimic granular material behavior in this unified framework specifically tailored for real-time applications. Frâncu and Moldoveanu [10] formulated an accurate contact and Coulomb friction model suitable for rigid and flexible bodies in PBD.

Recent advances in the field of machine learning have also produced new AI-based approaches to predicting the behavior of granular materials. The neural networks used in these approaches are trained with classically simulated data. Again, there are approaches that use continuum methods like Coombs and Augarde [7] who use MPM and Sanchez-Gonzalez et. al [23] who use SPH and MPM and approaches that use DEM like Wallin and Servin [27]. Furthermore, hybrid techniques exist that combine the strengths of continuum and discrete methods, like the recently proposed method by Yue et. al. [28].

In this work we focus on discrete methods. We try to overcome the weaknesses of discrete methods, namely the size of the individual particles, by splitting our simulation into two steps. First, an accurate PBD simulation that takes into account collisions between individual particles but is computed at a low resolution with large particle radii. Second, we use an efficient refinement algorithm that replaces the results of the actual simulation with a much higher number of particles with a smaller radius. The decoupling of the two simulation parts makes it possible to calculate both parts with different time steps. This way, our method is very efficient, since the upsampling in the second part can be done with much larger time steps than the contact calculation in the low resolution part. The idea of this dual partitioning is not new. It has been already applied by Alduán et. al. [2], who compute their low resolution (LR) guide particles using the continuum method of Bell et. al. [3]. They move their high resolution (HR) visualization particles using the flow of LR particles as well as external forces. Ihmsen et. al. [15] took up this idea. They calculated their LR particles using another continuum based method, namely the friction model in SPH developed by Alduán and Otaduy [1]. Furthermore, they optimized the algorithm to interpolate the motion of the HR particles by superimposing external forces on the velocity field of the LR particles depending on the density of the LR particles. In contrast to the previously mentioned work, we use a discrete method with PBD. This allows real-time simulation of granular media thanks to the speed advantages of PBD over SPH. In addition, we modify the algorithm for advection of HR particles to achieve better interaction with domain boundaries and prevent particles from sticking to rigid bodies.

3 LR SIMULATION

In this section, we first describe the simulation of LR guiding particles. As mentioned before, our LR simulation is based on PBD [20], respectively the modification described by Macklin et. al. [18] for parallel execution by constraint averaging. This allows interactive frame rates as needed in real-time applications even for larger numbers of particles.

PBD and other discrete simulation methods use particles to represent each individual discrete element of the material being simulated. All particles have the same radius r_{LR}. In general these particles can have arbitrary attributes. For our purposes it is sufficient to assign each particle a position $\boldsymbol{x}$, a velocity $\boldsymbol{v}$ and a mass m.

In other discrete simulation methods, the particle motion is determined by time integration of all occurring internal and external forces. In PBD, however, position changes $\Delta\boldsymbol{x}$ due to internal forces are expressed directly by so-called constraints. The resulting velocity of individual particles is then calculated from the position difference between two time steps, i.e.

$$\Delta\boldsymbol{v} = \frac{\Delta\boldsymbol{x}}{\Delta t_{LR}}.$$

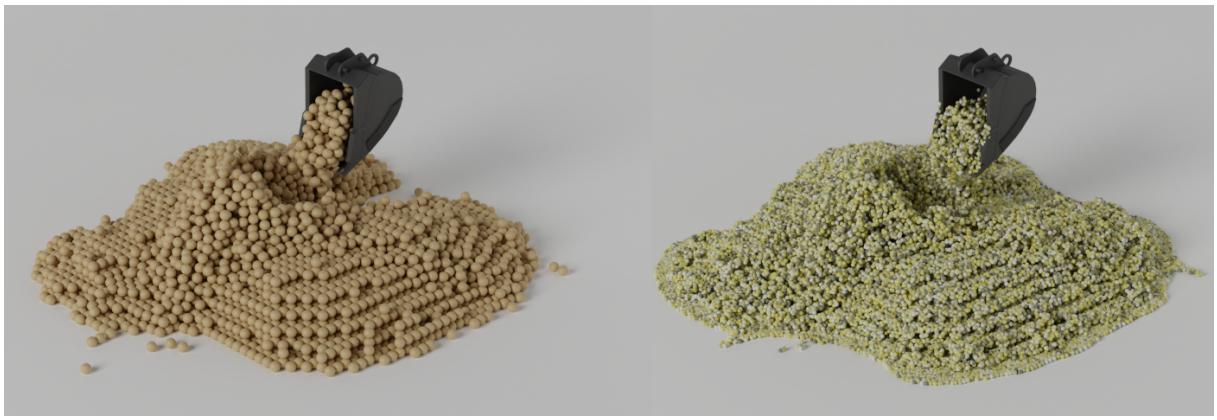


Figure 1: An excavator lifts up sand. left: LR Simulation with 6.8k Particles. right: HR Simulation with 147k.

Each of the constraints used in PBD can affect k particles. In general a constraint can be an equality constraint $C(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_k)=0$ or an inequality constraint $C(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_k)\geq 0$.

For our granular LR simulation we use the friction model proposed by Macklin et. al. [18]. Their model consists of two different contact constraints between a collision pair $\boldsymbol{x}_i$, $\boldsymbol{x}_j$. First, interpenentrations between the two collision partners are solved by displacing the two particles along the collision vector $\boldsymbol{x}_{ij}=\boldsymbol{x}_j-\boldsymbol{x}_i$. The displacement is proportional to the mass ratio:

$$\Delta\boldsymbol{x}_i=-\hat{m}_{ij}\left(2r_{LR}-\left|\boldsymbol{x}_{ij}\right|\right)\cdot\frac{\boldsymbol{x}_{ij}}{\left|\boldsymbol{x}_{ij}\right|} \tag{1}$$

$$\Delta\boldsymbol{x}_j=\hat{m}_{ji}\left(2r_{LR}-\left|\boldsymbol{x}_{ij}\right|\right)\cdot\frac{\boldsymbol{x}_{ij}}{\left|\boldsymbol{x}_{ij}\right|} \tag{2}$$

with

$$\hat{m}_{ij}=\frac{m_i^{-1}}{m_i^{-1}+m_j^{-1}}=\frac{m_j}{m_i+m_j}$$

After resolving collisions the particles i,j have new temporary positions $\tilde{\boldsymbol{x}}_i=\boldsymbol{x}_i+\Delta\boldsymbol{x}_i$, $\tilde{\boldsymbol{x}}_j=\boldsymbol{x}_j+\Delta\boldsymbol{x}_j$. The relative displacement between original positions and these new positions

$$\Delta\boldsymbol{x}_{ij}=(\tilde{\boldsymbol{x}}_i-\boldsymbol{x}_i)-(\tilde{\boldsymbol{x}}_j-\boldsymbol{x}_j)=\Delta\boldsymbol{x}_i-\Delta\boldsymbol{x}_j$$

is used to calculate a frictional position delta. It is based on the tangential component

$$\Delta\boldsymbol{x}_{ij_\perp}=\Delta\boldsymbol{x}_{ij}-(\Delta\boldsymbol{x}_{ij}\cdot\boldsymbol{x}_{ij})\frac{\boldsymbol{x}_{ij}}{\left|\boldsymbol{x}_{ij}\right|^2}$$

relative to the collision vector $\boldsymbol{x}_{ij}$. With this tangential component the positional delta for particle i,j is calculated as

$$\Delta\boldsymbol{x}_i=-\hat{m}_{ij}\begin{cases}\Delta\boldsymbol{x}_{ij_\perp} & \text{if } \left|\Delta\boldsymbol{x}_{ij_\perp}\right|<2r_{LR}\mu_s\\ \Delta\boldsymbol{x}_{ij_\perp}\min_{\text{fric}} & \text{else}\end{cases} \tag{3}$$

$$\Delta\boldsymbol{x}_j=\hat{m}_{ji}\begin{cases}\Delta\boldsymbol{x}_{ij_\perp} & \text{if } \left|\Delta\boldsymbol{x}_{ij_\perp}\right|<2r_{LR}\mu_s\\ \Delta\boldsymbol{x}_{ij_\perp}\min_{\text{fric}} & \text{else}\end{cases} \tag{4}$$

with

$$\min_{\text{fric}}=\min\left(\frac{2r_{LR}\cdot\mu_k}{\left|\Delta\boldsymbol{x}_{ij_\perp}\right|},1\right).$$

The parameters μ_s and μ_k are the dry friction coefficients for static and kinetic friction. For most cases, we use $\mu_s=0.35$ and $\mu_k=0.3$ in our simulation. For a collision pair with differing friction coefficients the cross friction coefficients

$$\mu_{ij}=\sqrt{\mu_i\mu_j}$$

has to be calculated, where μ_i and μ_j are the static respectively kinetic friction coefficients for particle i and j. The static and kinetic cross friction coefficients are used in this case to calculate the positional deltas $\Delta\boldsymbol{x}_i$ and $\Delta\boldsymbol{x}_j$.

While Eq. (1) and (2) resolve only the interpenetration of the particles, Eq. (3) and (4) model the friction effects. In the first case of Eq. (3) respectively (4), static friction effects are modeled by preventing any tangential motion if the particle velocity is below a traction threshold. Kinetic friction effects are treated in the second case by limiting the positional delta depending on the penetration depth.

The friction model described previously is used to calculate the friction effects between two particles. To model the interaction between particles and rigid-bodies or domain boundaries we sample them with particles as well. This allows us to use a unified collision handling and to work with the same constraints for all types of contacts. In the [25], a method is described to sample arbitrary closed volumes of 3D triangle meshes with particles for the needs of particle-based simulation. For stationary objects, the particles are assigned an infinitely large mass or an inverted mass of $m^{-1}=0$, respectively. In this case it is sufficient to only sample the surface of the object. A method to sample arbitrary surfaces of 3D triangle meshes based on the uniform sampling algorithm of Bowers et. al. [5] can also be found in [25]. For moveable rigid-bodies the individual particles are first treated

as if they were unconnected. Then, a shape-matching constraint [21] is applied to the particles of the rigid body to find the particle configuration corresponding to a transformed resting state. The position delta from this constraint is given by

$$\Delta \boldsymbol{x}_i = (\boldsymbol{R}\boldsymbol{x}_{o_i} + \boldsymbol{c}) - \tilde{\boldsymbol{x}}_i$$

where $\boldsymbol{c}$ corresponds to the center of gravity of the new deformed particles of the rigid body. $\boldsymbol{x}_{o_i}$ corresponds to the offset of the i-th particle from the center of gravity of the undeformed particle positions. $\boldsymbol{R}$ is a rotation matrix which is given by the polar decomposition [12] of the covariance matrix

$$\boldsymbol{C} = \sum_i^n (\tilde{\boldsymbol{x}}_i - \boldsymbol{c}) \cdot \boldsymbol{x}_{o_i}^T$$

of the deformed shape. This enables a two-way coupling between the granular medium and rigid-bodies.

All of these previously mentioned constraints, namely contact, friction and shape-matching constraints, can be calculated in parallel. A particle i can receive a position delta $\Delta \boldsymbol{x}_{i_k}$ from several constraints k. Therefore, after all constraints have been calculated, the sum of all positions delta belonging to a particle i is divided by n_i the number of constraints involved:

$$\Delta \boldsymbol{x}_i = \frac{1}{n_i} \sum_k^{n_i} \Delta \boldsymbol{x}_{i_k}.$$

The procedure of constraint solving is repeated a couple of times until a solution is found that satisfies all the constraints. For our calculations between 3 and 5 iterations were sufficient. However, it can happen that no convergence was achieved and particles are in an invalid position at the start of a simulation step. This can lead to particles experiencing an unwanted acceleration in the next time step due to the next constrain to solve. This effect is especially strong the smaller the time step is. To avoid this, 1-2 stabilization iterations of the pure contact constraints, Eq. (1) and (2), are executed per time step. The resulting position deltas are applied to the temporary positions as well as to the regular positions. This prevents unnatural kinetic energy from being added to the system due to these irregularities when calculating new velocities. The complete algorithm for performing one time-step is shown in Algorithm 1.

To achieve a more stable piling of particles, e.g. in sand piles, and to prevent dissolution, a further improvement is made. As described in [18] a scaled mass

$$m_i^* = m_i \cdot e^{-h(\boldsymbol{x}_i)}$$

is assigned to each particle i based on the relative height $h(\boldsymbol{x}_i)$ with respect to the ground plane. These scaled masses m^* are used to solve the contact and friction constraints. Since higher particles exert less pressure on the levels below, the system converges faster and the piles are more stable.

Algorithm 1 One simulation time-step

for all particles i **do**
  $\boldsymbol{v}_i \leftarrow \boldsymbol{v}_i + \Delta t_{LR} \boldsymbol{g}$ ▷ $|\boldsymbol{g}| = 9.81 \frac{m}{s^2}$
  $\tilde{\boldsymbol{x}}_i \leftarrow \boldsymbol{x}_i + \Delta t_{LR} \boldsymbol{v}_i$
  $m_i^* \leftarrow m_i e^{-h(\boldsymbol{x}_i)}$
find_neighboring_particles()
generate_stabilization_constraints()
for stabilization iterations $it = 1, 2$ **do**
  $\Delta \boldsymbol{x} \leftarrow 0, n \leftarrow 0$ ▷ $\forall$ particle i
  $\Delta \boldsymbol{x}, n \leftarrow$ solve_stabilization_constraints()
  for all particles i **do**
    $\tilde{\boldsymbol{x}}_i \leftarrow \tilde{\boldsymbol{x}}_i + \Delta \boldsymbol{x}_i / n_i$
    $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i + \Delta \boldsymbol{x}_i / n_i$
generate_constraints()
for solving iterations $it = 1, \ldots, 5$ **do**
  $\Delta \boldsymbol{x} \leftarrow 0, n \leftarrow 0$ ▷ $\forall$ particle i
  $\Delta \boldsymbol{x}, n \leftarrow$ solve_constraints()
  $\tilde{\boldsymbol{x}}_i \leftarrow \tilde{\boldsymbol{x}}_i + \Delta \boldsymbol{x}_i / n_i$
for all particles i **do**
  $\boldsymbol{v}_i \leftarrow (\tilde{\boldsymbol{x}}_i - \boldsymbol{x}_i) / \Delta t_{LR}$
  $\boldsymbol{x}_i \leftarrow \tilde{\boldsymbol{x}}_i$

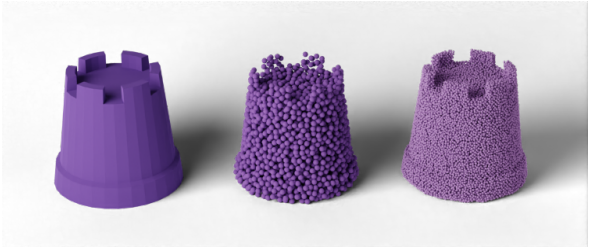


Figure 2: Exemplary illustration of the sampling process. left: Inital sandcastle mesh. middle: LR sampling. right: HR sampling

4 HR UPSAMPLING

In the previous section we described the simulation of LR guide particles taking into account all collisions between particles. We now describe how a finer resolution simulation result can be generated on the basis of the LR particles with the help of an upsampling. Figure 1 shows a comparison between the LR simulation on the left and the result of the HR upsampling on the right. This is done without having to perform complex collision queries between HR particles. Thus this upsampling is extremely effective and the size of the LR time step is decoupled from the size of the HR particles. Each LR Particle can be seen as a representation of several HR Particles.

It is especially important to create a good initial sampling to avoid artifacts. For example, a uniform sampling can lead to repetitive patterns during the simulation. Furthermore, if the HR particles are placed symmetrically around the LR particles, aliasing or staircase patterns can occur already in the initial state. Therefore,

we use the randomized volume sampling algorithm described in [25] for both the LR and the HR particles. We use a triangle mesh as a hull in which the particles should be located. First the bounding box delimiting the mesh is divided into grid cells with a side length of $\frac{2r}{\sqrt{3}}$, with r being the radius of the LR particles r_{LR} respectively HR particles r_{HR}. Then, a large number of randomly but uniformly distributed candidate positions are generated within the mesh and assigned to the respective grid cells. After that, for each grid cell within the mesh, an attempt is made to select a candidate position that does not overlap with already sampled positions. In this fashion the whole volume of the mesh is sampled. For a more detailed description we refer the reader to [25]. This algorithm is executed for both LR particles with radius r_{LR} and HR particles with radius r_{HR} before starting the simulation. Figure 2 shows an example of the sampling process. The initial mesh is displayed on the left, while in the middle the sampling for the LR simulation and on the right the corresponding sampling for the HR upsampling is shown.

Once the simulation is started, the LR particles are set in motion by the LR simulation described in the previous section. The HR particles trace the movement of these LR guide particles. HR particles should follow the flow of the LR simulation but still move individually to avoid the formation of clumps. For this we use the advection method described by Ihmsen et. al. [15]. For this, gravity is smoothly faded in as an external force depending on how densely a HR particle is surrounded by LR particles. If a HR particle is in the vicinity of many LR particles, the velocity field generated by them dominates. For this, distance-based weights w between HR particles i and LR particles j are calculated as

$$w_{ij} = \max\left(0, \left(1 - \frac{|\boldsymbol{x}_{ij}|^2}{9 \cdot r_{LR}^2}\right)^3\right).$$

These weights are used to determine the average velocity $\boldsymbol{v}_i$ at a HR particle i with

$$\tilde{\boldsymbol{v}}_i = \frac{1}{\sum_j w_{ij}} \sum_j w_{ij} \boldsymbol{v}_j.$$

It shall be noted that LR particles j can be granular material as well as the particle representation of rigid-bodies or boundaries. The blending in of external forces is controlled by a parameter

$$\alpha_i = \begin{cases} 1 - \max w_{ij} & \text{if } \max w_{ij} \leq c_1 \text{ or } \frac{\max w_{ij}}{\sum_j w_{ij}} \geq c_2 \\ 0 & \text{else} \end{cases}$$

which differs from zero only in sparse regions. The constant $c_1 = \frac{512}{729}$ is the distance-based weight for a distance equal to the LR particle radius r_{LR} and $c_2 = 0.6$ is an empirically tested value. With this blending parameter the resulting velocity for a HR particle i is calculated as

$$\boldsymbol{v}_i^{t+1} = (1 - \alpha_i)\tilde{\boldsymbol{v}}_i^{t+1} + \alpha_i \left(\boldsymbol{v}_i^t + \Delta t_{HR} \boldsymbol{g}\right)$$

where $\boldsymbol{g}$ is the acceleration due to gravity and Δt_{HR} denotes the time-step size from time t to time $t+1$. The new position $\boldsymbol{x}_i$ for the i-th HR particle is then trivially obtained by time integration through

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \Delta t_{HR} \boldsymbol{v}_i^{t+1}.$$

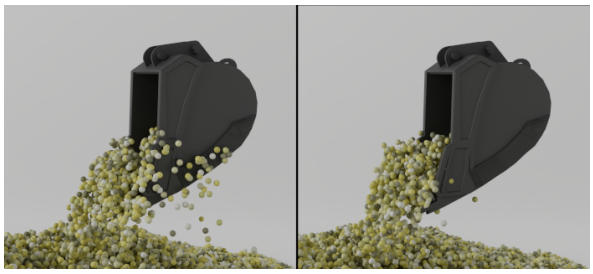


Figure 3: left: Ihmsen et. al.'s original upsampling with sticking artifacts. right: HR Simulation with our modification.

In contrast to Ihmsen et. al. we ignore the LR particles of rigid bodies if there are no LR granular particles in the vicinity. This prevents HR particles from sticking to rigid bodies that are moved externally (see Figure 3).

5 RESULTS

In contrast to previous works, our work focuses on runtimes in the range of interactive frame rates. Therefore, in this section we present some examples of our approach to illustrate the different possible applications.

5.1 Implementation

We have implemented our simulation framework in C++ 14. For more complex mathematical operations and mathematical data structures like vectors and matrices we use the Eigen3 [11] library. We parallelized our algorithms on the CPU using OpenMP [9]. For an efficient neighborhood search we use the CompactNSearch library based on the Compact Hashing approach by Ihmsen et al. [14]. Furthermore, we use the LEAVEN [25] library for sampling volumes and surfaces. All renderings in this work were created with Cycles in Blender [6].

5.2 Experiments & Performance

In this section we would like to underline and evaluate the usefulness of our method by suitable experiments (scenes). For this purpose, we have selected four different test scenarios. For all scenarios we use a default time step $\Delta t_{LR} = 0.005$s for the LR simulation, which is further constrained by the CFL condition if necessary. The time step for upsampling is $\Delta t_{HR} = 0.0167$s.

	Particles		Time per Frame	
	LR	HR	LR	HR
Sandcastle	2.4k	90k	9.75ms	7.31ms
Excavator	6.8k	147k	21.4ms	11.2ms
Hourglass	10k	460k	47.4ms	48.5ms
Squirrel	10k	207k	33.0ms	15.8ms

Table 1: Timing results for four different scenes

The density of the granular medium is $\rho = 1600\frac{\text{kg}}{\text{m}^3}$ and the kinetic and static friction coefficients are $\mu_k = 0.3$ and $\mu_s = 0.35$, respectively. In general, we use LR simulation radii r_{LR} between 0.005m and 0.05m. For the upsampling we use a HR particle radius between $r_{HR} = 0.2 \cdot r_{LR}$ and $r_{HR} = 0.4 \cdot r_{LR}$. This leads to an upscaling factor between 15 and 125.

Scene Sandcastle (see Figure 4) consists of a excavator controllable by user interaction and a sand castle with a relatively small number of particles. The LR particle radius is 0.03m and the upsample radius is 0.01m, resulting in 2.4k LR particles to 90k HR particles. This scene serves to demonstrate the real-time capability of our method, in which relatively few elaborately simulated particles can be used to obtain visually pleasing results by upsampling.

Scene Excavator , shown in Figure 1, also contains an excavator. In this example, a radius of 0.02m generates almost three times as many LR particles (6.8k) as in Sandcastle . The upsampling radius 0.008m generates 147k HR particles. This scene is intended to illustrate that our method can be used to simulate even complicated interactions such as the lifting and lowering of granular material with an excavator.

The Hourglass example should on one hand clarify the behavior and runtime of larger particle numbers and on the other hand, more importantly, give an impression of the stable piling of our method. For this purpose 10k LR particles in the hourglass are upsampled to 460k HR particles. Radii of 0.028m and 0.008m are used.

Scene Squirrel shows the two-way coupling between rigid bodies and the granular material. For this purpose, a squirrel is hurled into the sandcastle with an initial velocity. The squirrel is represented by 1.4k particles with a density of $\rho = 5000\frac{\text{kg}}{\text{m}^3}$, which are simulated in the LR simulation. Afterwards, the original squirrel shape is reconstructed by shape matching as described in Section 3, which accomplishes the two-way coupling. All LR simulation particles have a radius of 0.02m and the upsampling radius of the granular particles is 0.0075m. In addition to the 1.4k particles of the squirrel, there are 10k LR granular particles in the simulation, which are upsampled to 207k.

All scenes were calculated on an Intel Core i9-9980HK with 8 cores on the CPU alone. Table 1 shows the per frame timing results for the four different scenes. For the timing of the LR simulation, the

Figure 4: An excavator is moved by user input through a sand castle

Figure 5: An hourglass with sand containing 460k particles.

Figure 6: A squirrel hurled into a sand castle.

run-times of as many simulation steps as necessary to calculate a time difference $t = \Delta t_{HR} = 0.0167$s were combined. It shows that even for relatively high particle counts of 500k a computation within interactive frame rates is possible with our method on consumer hardware without exploiting massive parallelism on GPUs. Especially where the resources of the GPU are already needed for e.g. computationally intensive physically based rendering for realistic visualization, our pure CPU based simulation can show its advantages.

6 CONCLUSION & FUTURE WORK

With this work we have shown that with the help of an upsampling algorithm it is possible to scale up relatively small numbers of particles to produce visually vivid results within interactive frame rates for the simulation of granular material. It was shown that the up-

sampling method described by Ihmsen et. al. [15] for SPH simulations, which are far away from real-time computation times, is very well suited for application in interactive PBD simulations. We believe that this is advantageous for future real-time applications in interaction with granular material.

7 LIMITATIONS & FUTURE WORK

On one hand, in position-based simulations friction is dependent of the number of iterations, which has a negative impact on the correctness of the friction effects in the LR simulation part. The modeled static and kinetic friction effects are only approximations. On the other hand, static friction cannot be modeled correctly with HR upsampling, which prevents stable piling in some situations.

Furthermore, currently only one constant radius for all particles in the LR simulation and another constant radius for all particles in the HR upsampling is possible.

In the future, we plan to take advantage of the high parallizability of the PBD variant described by Macklin et. al. [18] and the trivial parallelizability of the upsampling algorithm to create a GPU-based version of our simulation that can simulate even larger numbers of particles in real time.

ACKNOWLEDGMENTS

This Project is supported by the Federal Ministry for Economic Affairs and Climate Action (BMWK) on the basis of a decision by the German Bundestag.

REFERENCES

[1] Iván Alduán and Miguel A. Otaduy. Sph granular flow with friction and cohesion. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, pages 25–32, New York, NY, USA, 2011. Association for Computing Machinery.

[2] Iván Alduán, Ángel Tena, and Miguel Otaduy. Simulation of high-resolution granular media. *CEIG 2009: Congreso Español de Informática Gráfica*, 09 2009.

[3] Nathan Bell, Yizhou Yu, and Peter Mucha. Particle-based simulation of granular material. pages 77–86, 01 2005.

[4] Jan Bender, Matthias Müller, and Miles Macklin. A Survey on Position Based Dynamics. In Adrien Bousseau and Diego Gutierrez, editors, *EG 2017 - Tutorials*. The Eurographics Association, 2017.

[5] John Bowers, Rui Wang, Li-Yi Wei, and David Maletz. Parallel poisson disk sampling with spectrum analysis on surfaces. 29(6), December 2010.

[6] Blender Online Community. Blender - a 3d modelling and rendering package. http://www.blender.org, 2021.

[7] William Coombs and Charles Augarde. Ample: A material point learning environment. *Advances in Engineering Software*, 139:102748, 01 2020.

[8] P. Cundall and O. Strack. Discussion: A discrete numerical model for granular assemblies. *Geotechnique*, 30:331–336, 01 1980.

[9] L. Dagum and R. Menon. Openmp: An industry-standard api for shared-memory programming. *Computational Science & Engineering, IEEE*, 5:46 – 55, 02 1998.

[10] Mihai Frâncu and Florica Moldoveanu. Unified simulation of rigid and flexible bodies using position based dynamics. 04 2017.

[11] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[12] Nicholas John Higham. Computing the polar decomposition with applications. *Siam Journal on Scientific and Statistical Computing*, 7:1160–1174, 1986.

[13] Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Trans. Graph.*, 37(4), July 2018.

[14] Markus Ihmsen, Nadir Akinci, Markus Becker, and Matthias Teschner. A parallel sph implementation on multi-core cpus. *Comput. Graph. Forum*, 30:99–112, 03 2011.

[15] Markus Ihmsen, Arthur Wahl, and Matthias Teschner. High-Resolution Simulation of Granular Material with SPH. In Jan Bender, Arjan Kuijper, Dieter W. Fellner, and Eric Guerin, editors, *Workshop on Virtual Reality Interaction and Physical Simulation*. The Eurographics Association, 2012.

[16] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. pages 1–52, 07 2016.

[17] Toon Lenaerts and Philip Dutré. Mixing fluids and granular materials. *Comput. Graph. Forum*, 28:213–218, 04 2009.

[18] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Trans. Graph.*, 33(4), July 2014.

[19] J. Monaghan. Smoothed particle hydrodynamics. *Annual review of astronomy and astrophysics*, 30:543–574, 1992.

[20] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109 – 118, 2007.

[21] Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. Meshless deformations based on shape matching. *ACM Trans. Graph.*, 24:471–478, 07 2005.

[22] Rahul Narain, Abhinav Golas, and Ming C. Lin. Free-flowing granular materials with two-way solid coupling. In *ACM SIGGRAPH Asia 2010 Papers*, SIGGRAPH ASIA '10, New York, NY, USA, 2010. Association for Computing Machinery.

[23] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. *Proceedings of the 37th Internation Conference on Machine Learning*, abs/2002.09405, 2020.

[24] Barbara Solenthaler and Renato Pajarola. Predictive-corrective incompressible sph. In *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, New York, NY, USA, 2009. ACM.

[25] Alexander Sommer and Ulrich Schwanecke. Lightweight surface and volume mesh sampling application for particle-based simulations. *29. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2021*, pages 155–160, 01 2021.

[26] Deborah Sulsky, Shi-Jian Zhou, and Howard Schreyer. Application of particle-in-cell method to solid mechanics. *Computer Physics Communications*, 87:236–252, 05 1995.

[27] Erik Wallin and Martin Servin. Data-driven model order reduction for granular media. *Computational Particle Mechanics*, 02 2021.

[28] Yonghao Yue, Breannan Smith, Peter Chen, Maytee Chantharayukhonthorn, Ken Kamrin, and Eitan Grinspun. Hybrid grains: adaptive coupling of discrete and continuum simulations of granular media. volume 37, pages 1–19, 12 2018.

[29] Yongning Zhu and Robert Bridson. Animating sand as a fluid. *ACM Trans. Graph.*, 24(3):965–972, July 2005.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Dynamic Sensor Matching based on Geomagnetic Inertial Navigation

Simone Müller
0000-0001-5830-8655
Leibniz Supercomputing Centre (LRZ)
Boltzmannstrasse 1
85748 Garching bei München
simone.mueller@lrz.de

Dieter Kranzlmüller
0000-0002-8319-0123
Ludwig-Maximilians-Universität (LMU)
MNM-Team
Oettingenstr. 67
80538 München
kranzlmueller@ifi.lmu.de

Abstract

Optical sensors can capture dynamic environments and derive depth information in near real-time. The quality of these digital reconstructions is determined by factors like illumination, surface and texture conditions, sensing speed and other sensor characteristics as well as the sensor-object relations. Improvements can be obtained by using dynamically collected data from multiple sensors. However, matching the data from multiple sensors requires a shared world coordinate system. We present a concept for transferring multi-sensor data into a commonly referenced world coordinate system: the earth's magnetic field. The steady presence of our planetary magnetic field provides a reliable world coordinate system, which can serve as a reference for a position-defined reconstruction of dynamic environments. Our approach is evaluated using magnetic field sensors of the ZED 2 stereo camera from Stereolabs, which provides orientation relative to the North Pole similar to a compass. With the help of inertial measurement unit informations, each camera's position data can be transferred into the unified world coordinate system. Our evaluation reveals the level of quality possible using the earth magnetic field and allows a basis for dynamic and real-time-based applications of optical multi-sensors for environment detection.

Keywords

Dynamic Matching, Multi-Sensor, Magnetic Inertial Navigation, Real-Time, Computer Vision

1 INTRODUCTION

Many devices use 3D reconstructions of their surroundings for locomotion and interaction in complex visual environments [Kok18]. Epipolar geometry based distance information of depth sensors allows us to compute 3D points as point clouds. Multiple depth sensors can be used efficiently for real-time point cloud expanding and optimising [Pia13, Mue21]. The positional overlay of received depth information reduces sensor and image errors. This involves the positional accuracy and stability of depth sensors. Using global navigation of inertial navigation systems (GNSS/INS) enables a temporal position adjustment of these sensors [Hua19, Vu12]. However, tracking systems like Garmin Oregon 700 are insufficient for the matching of multiple sensors due to positional deviations of 3 m to 25 m [Gar22]. These position deviations affect the quality of rendered point clouds due to incorrect coordinate alignments, scattering, outliers or offsets of neighbouring depth points [Car16, Kad14]. The required visual contact between satellite and receiver additionally limits the local GPS signal [Chu11]. Alternative Visual Inertial Navigation Systems (VINS) offer limited benefits for multiple sensor matching in terms of sensor drifts, measurement errors and range [Car16, Hua19, Kad14].

Our motivation is based on the challenges of using stable inertial navigation systems for multiple sensor matching. The persistence of the geomagnetic field lends itself to our concept. We apply the combination of 3D depth technologies and smart sensor architectures for the use of magnetic fields in an inertial system. Smart sensor architectures offer a gradient transformation between acceleration, angular velocity and magnetic field in the meter defined world coordinate system $W_{(x,y,z)}$ [Han07, Car16]. Our contribution comprises the following aspects:

- Location and time independent matching of multiple sensors based on geomagnetic inertial navigation
- Sensory setup for validation of our approach

Notation	Definition
a	Acceleration $[m/s^2]$
g	Gravity 9.81 $[m/s^2]$
B	Baseline between C_L and C_R [cm]
B_1, B_n	Magnetic field strength $[\mu T]$
b_H	Temperature Dependent Bias $[\mu T]$
C	Transformation Matrix $[\mu T]$
m	Seismic mass of accelerometer $[mg]$
$R_{(\psi,\phi,\theta)}$	Rotation (Pitchψ, Rollϕ, Yawθ)
s	Distance $[m]$
t	Time $[s]$
T	Translation $[m]$
v	Velocity $[m/s^2]$
λ	Depth $[m]$
φ	Rotation angle $[rad]$
θ	Orientation $[°]$
ω	Angular velocity $[rad/s]$
Abbr.	**Definition**
CMOS	Complementary Metal Oxide Semiconductor
GNSS	Global Navigation Satellite Systems
GPS	Global Positioning System
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
INS	Inertial Navigation Systems
LiDAR	Light Detection And Ranging
SLAM	Simultaneous Localisation and Mapping
ToF	Time of Flight
VO	Visual Odometry
VINS	Visual Inertial Navigation Systems
VIO	Visual Inertial Odometry
WCS	World Coordinate System

Table 1: **Notations and Abbreviations**

- Analysis of positioning-based accuracy, reliability and stability in a transformed world coordinate system

Table 1 lists the used definitions, notations and abbreviations. This Paper is organised as follows:

- **Section 2** overviews related works of sensor matching and valid transformations in inertial systems.
- **Section 3** discusses our concept of geomagnetic inertial navigation and coordinate transformation of the magnetic field sensor.
- **Section 4** lists the steps of our methodology for the evaluation.
- **Section 5** overviews setup information of our measurement implementation.
- **Section 6** discusses the static and dynamic results of our geomagnetic inertial navigation.
- **Section 7** describes our conclusion and resulting future work.

2 RELATED WORK

The challenges of dynamic matching demands a referencable and accurate world coordinate system (WCS) in which several depth sensors move in defined positions. In this section, we describe the matching of multiple sensors and transferability to this WCS. Based on related work, we provide a foundation for our hypothesis and methodological approach.

Multiple Depth Sensor Systems Locomotive sensors require information about their translation $T_{x,y,z}$, rotation $R_{\psi,\phi,\theta}$ and local depth λ [Mue21].

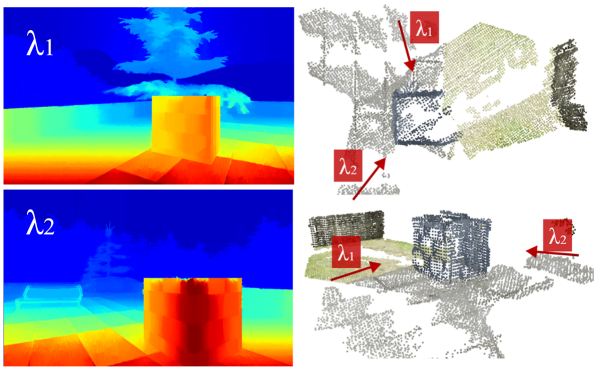


Figure 1: **Point Cloud Matching**: The resulting point cloud (right) is matched from several depth images (top and bottom left). The arrows of λ_1 and λ_2 represent the camera orientation.

3D depth technologies like Light Detection And Ranging (LiDAR), Time of Flight (ToF) or stereoscopic systems receive the emitted light of their surroundings and convert it to electrical signals [Yoe21]. The depth information (λ_1 and λ_2 on the left side of Fig. 1) can be calculated algorithmically from the converted digital sensor signals. One commonly used algorithm for point cloud registration is Iterative Closest Point (ICP), which involves the determination of point cloud specific rotation and translation [Tar10, Mue21]. Approaches like Simultaneous Localisation and Mapping (SLAM), Visual Odometry (VO), visual detection and tracking, as well as visual classification and recognition enable 3D evaluations by means of volumetric rendered point clouds (right side of Fig. 1) [Mue21]. Volumetric data can be converted into polygonal meshes in order to manipulate objects volumetrically on the fly [Tak15].

Quality characteristics of these point clouds can be increased by matching multiple depth images as shown in Fig. 1. Error corrections of distortions, scatter, noise, sensor defects or latencies as well as point cloud deviations characterise such quality features [Kad14, Mue21, Tak15].

Takimotoa et al. [Tak15] examined the matching of multiple point clouds and described the finding of ICP correspondences as challenges when texture-free surfaces of point clouds are to be reconstructed densely

and accurately. Their measurements indicate that low-precision sensors are capable of reasonably good reconstructed objects. They conclude that increased number of acquisitions and SIFT methodology do not improve the final point cloud quality. Instead, they suggest that the adjustment of ICP and reconstruction parameters leads to improvements due to limited sensor accuracy and stability.

Piatkowska et al. [Pia13] optimised spatio-temporal and three-dimensional reconstructions with asynchronous time-based image sensors and extended existing methods for event-based processing. They conclude that dynamic, asynchronous and cooperative implementations are possible using a specialised algorithm.

In [Mue21], we describe the possibility of synchronous and dynamic sensor matching for limited sensor and object ranges. We name the alignment and transferability relevance of precise sensor positions for successful depth matching.

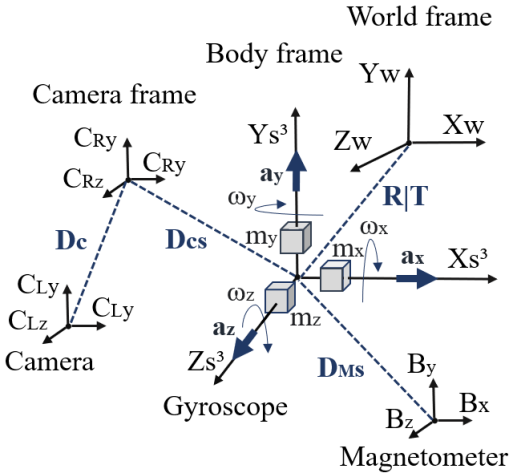


Figure 2: **Combination of 3D Depth Sensing and Smart Sensor Architecture**: Relationships between the stereo camera and IMU integrated gyroscope and magnetometer in a defined world frame coordinate system. The received images of C_R and C_L are located in the extrinsic camera frame [Mue21]. In contrast to the cameras, the IMU is originally located in body frame.

Inertial Navigation Prior research indicates that minor deviations of aligned point clouds can be minimised by accurate positional information. Data fusion of inertial sensors in form of smart sensor architectures (see Fig. 2) reduce the influence of a failure prone single sensor [Car16]. This allows the necessary optimisation of stability and precise movement detection.

Inertial Measurement Unit (IMU) and depth integrated sensors have statically defined distances D_{CS}, D_{MS} and D_c to each other. Estimated sensor positions and orientations are coordinated in camera extrinsic depth sensors or body frame IMUs [Car16, Mue21]. Positional relations of multiple used sensors are not directly transferable into the world frame since each sensor uses its own coordinate definitions. Therefore, rotations and translations have to be transformed from camera and body into a world frame defined coordinate system [Car16]. The attitude representation of quaternions q as shown in Eq. 1 describes Euler's principle rotations φ from inertial to body frame [Vec22, Car16]. We obtain the current position by measuring actual speed v, angular velocity ω, acceleration a, gravity g, and local magnetic field B (Eq. 2, 3 and 4) [Car16].

$$q = [cos\frac{\varphi}{2}, \varphi \cdot sin\frac{\varphi}{2}]^T \tag{1}$$

$$\frac{\partial q}{\partial t} = \frac{1}{2} q \cdot \omega \tag{2}$$

$$\frac{\partial v}{\partial t} = \omega \times v + a + q \cdot g \cdot q^1 \tag{3}$$

$$\frac{\partial B}{\partial t} = \omega \times B + \nabla B \cdot v \tag{4}$$

We use ∇ as 3×3 gradient matrix. A continuous and time-dependent initialisation of body oriented dynamic sensors in the transformed world frame is necessary to ensure positional accuracy.

Smart sensor architectures include IMUs consisting of three accelerometers, gyroscopes and magnetometer [Auf11]. The body frame denotated angular velocity ω of the gyroscope (Eq. 5) can be determined from the measured angular velocity ω_m, temperature dependent bias b_t and additive η of zero-mean Gaussian noise [Vec22, Kok18].

$$Gyroscope: \omega = \omega_m + b_t + \eta \ \{ \ \eta \sim N(0, \sigma^2_{gyro}) \tag{5}$$

ω_m defines the sensor's angular velocity of body frame with respect to earth inertial frame [Kok18].

$$Orientation: \theta_{(t+\Delta t)} \approx \theta_{(t)} + \frac{\partial}{\partial t}\theta_{(t)}\Delta t + \varepsilon \tag{6}$$

The orientation θ (Eq. 6) of gyroscope measurements can be approximated by Taylor expansion [Vec22]. $\theta_{(t+\Delta t)}$ describes the angle at current step and $\theta_{(t)}$ defines last changed time step Δt. ε is the approximation error. R is presented by Euler angles (ϕ, θ, ψ) [Kok18, Mue21].

$$R_{x(\phi)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\phi & -sin\phi \\ 0 & sin\phi & cos\phi \end{pmatrix} \tag{7}$$

$$R_{y(\theta)} = \begin{pmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{pmatrix} \tag{8}$$

$$R_{z(\psi)} = \begin{pmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{9}$$

We can calculate the translation by measuring the linear acceleration a_{lin} (Eq. 10) of accelerometer (Acc) [Kok18].

$$Acc : a_{\text{lin}} = a^{(g)} + a^{(l)} + \eta \ \{ \ \eta \sim N(0, \sigma^2_{\text{acc}}) \qquad (10)$$

In motionless state, we measure the noisy gravity vector $a^{(g)}$ and zero-mean Gaussian noise η with a magnitude of $9.81 m/s^2 = 1g$. In case of movement, the external force $a^{(l)}$ interacts additively [Kok18]. Accelerometers are suitable for long-term measurements due to absence of drifts and constant positions of earth's gravity centre [Car16]. However, noise behaviour is evident. The lack of information about yaw θ allows correct tilts only for roll ψ and pitch ϕ [Vec22].

The magnetometer enables the determination of θ since the actual direction of ϕ, θ, ψ depends on latitude and longitude [Vec22]. They refer to an Earth-Fixed Coordinate System (ECEF). This transformation from inertial to earth-fixed coordinate system is described as a rotation since a common reference is used.

Magnetic Inertial Navigation A further method for inertial navigation is the position determination by means of magnetic sensors.

Shi et al. [Shi21] examined the navigated indoor positioning using optimisation algorithms for magnetic reference maps. They demonstrated in their motion experiment that positional accuracy and matching with inertial navigation devices is significantly improved through magnetic references.

Kok et al. [Kok18] demonstrated empirically that magnetic field maps achieve efficient position estimates. They identified the necessary proximity between sensor and magnetic field generating coils for radial positions as well as altitude error reduction and concluded that further magnetic disturbance decreases the information content of measurements.

Caruso et al. [Car16] showed that fused estimation of magnetometer arrays and VINS are able to reconstruct outdoor trajectories where Magneto-Inertial Dead-Reckoning (MI-DR) techniques fail due to gradient leaks. However, magnetic estimation techniques and leakage of suppressed magnetic information need to be improved. They concluded that magnetic navigation could expand the application range in unfavourable environments and reduce power consumptions.

VINS Initial orientations of navigated devices can be estimated with IMUs. The combined merging of camera and IMU data allows extensive image information for efficient position solutions [Yan19]. VINS use points and lines with online spatial and temporal calibrations. Additional feature observations from different keyframes enable a reduction of trajectory sensor drifts [Hua19].

Huang [Hua19] investigated short-term compatibility for 3D motion tracking by comparing VIO and SLAM for local navigation. VINS is not suitable for long-term, large-scale, safety-critical deployments and under difficult conditions. This was mainly due to poor illumination and movement. Geometric features such as points, lines and areas, as used in current VINS for localisation, are unsuitable for semantic localisation and mapping. The real-time implementation of VINS continues to be challenging, despite initial efforts. Auxiliary sensors for specific environments and movements, such as sonar or LiDAR, enable better detection of dynamic movements. Huang asserts that simple integration of high-frequency IMU measurements is unreliable for long-term navigation, due to noise and distortion behaviour.

Tardif et al. [Tar10] demonstrated a robust image-based inertial navigation system for rural and urban environments based on VINS. Experimentally, the prototype showed low deviations at a maximum speed of $70 km/h$.

In [Vu12], experimental results have shown that the combination of DGPS and a single visual feature measurement at $1Hz$ is sufficient to achieve $1m$ positional accuracy. Vu et al. used visual features like traffic lights to generate a better positional and situational awareness in their tightly coupled real-time vision and DGPS-based INS.

3 GLOBAL SENSOR MATCHING

This section describes the underlying concept for global matching of multiple sensors in a new WCS. We define the first local sensor as the new origin of this WCS and transform the initialised data of all remaining sensors once to the first sensor point $\{B_1\}$. This enables future extensible matchings with VINS or feature detection and additional inclusions of local fused sensor information.

To represent movements, it is necessary to transform intrinsic coordinates of depth and position sensors into a new WCS. We observe the IMU integrated gyroscope and magnetometer of two different stereo cameras (1 and n). A common reference coordinate allows the calculation of positional IMU relationships. The defined points $\{B_1\}$ and $\{B_{\text{n}}\}$ of the magnetometers refer to the magnetic North Pole as common origin $\{N_{\text{p}}\}$ (Fig. 3).

From this consideration we can form the mathematical relationships between $\{N_{\text{p}}\}$, $\{B_1\}$ and $\{B_{\text{n}}\}$. The point P_{w} marks the new origin of our common WCS ($P_{\text{w}} \hat{=} \{B_1\}$). We define the scalar values $\vec{D}_{\text{nNp}}$, $\vec{D}_{1\text{n}}$ and $\vec{D}_{1\text{Np}}$ between $\{N_{\text{p}}\}$, $\{B_1\}$, $\{B_{\text{n}}\}$ (Fig. 3) and denote the rotational coordinate transformations as ${}^{B_{\text{n}}}_{N_{\text{p}}}R^T$, ${}^{B_1}_{B_{\text{n}}}R^T$ and ${}^{B_1}_{N_{\text{p}}}R^T$.

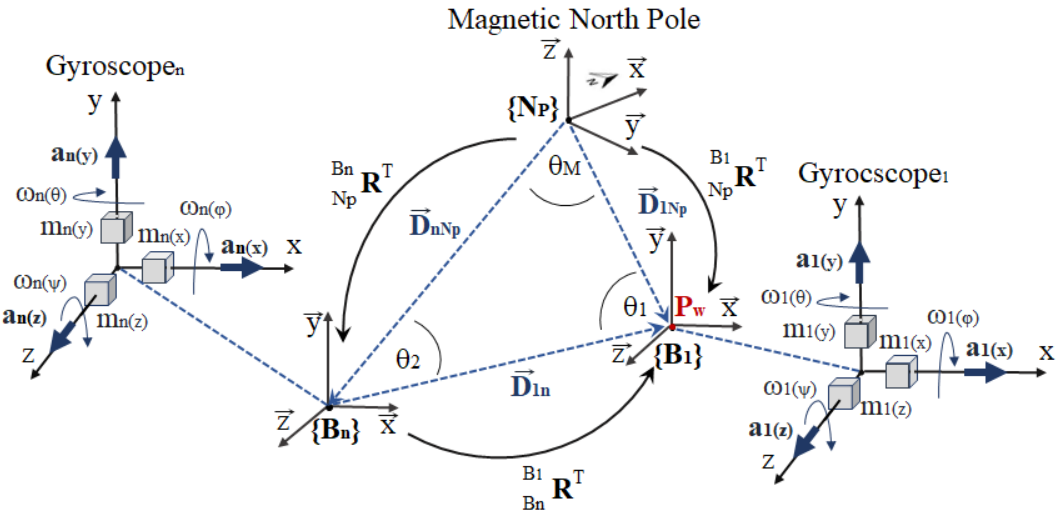


Figure 3: **Conceptual Representation of Global Sensor Matching**: Description of the geometric relationships between the sensor points $\{B_1\}$,$\{B_n\}$ and common origin $\{N_p\}$ in the WCS.

The following relationships apply to the transformation of $\{B_n\}$, $\{B_1\}$ and $\{N_p\}$:

$$B_n = {}^{B_n}_{N_p}R^T \cdot (N_p - \vec{D}_{nNp}) \tag{11}$$

$$B_1 = {}^{B_1}_{B_n}R^T \cdot (B_n - \vec{D}_{1n}) \tag{12}$$

$$B_1 = {}^{B_1}_{N_p}R^T \cdot (N_p - \vec{D}_{1Np}) \tag{13}$$

$$\vec{D}_{1N_p} = \vec{D}_{nN_p} + \vec{D}_{1n} \tag{14}$$

By transforming Eq. 11 and Eq. 13 according to $\{N_p\}$, we obtain the distance $\vec{D}_{1n}$ between the position of the magnetic sensors $\{B_1\}$ and $\{B_n\}$.

$$\frac{B_n}{{}^{B_n}_{N_p}R^T} + \vec{D}_{nNp} = \frac{B_1}{{}^{B_1}_{N_p}R^T} + (\vec{D}_{nN_p} + \vec{D}_{1n}) \tag{15}$$

$$\vec{D}_{1n} = \frac{B_n}{{}^{B_n}_{N_p}R^T} - \frac{B_1}{{}^{B_1}_{N_p}R^T} \tag{16}$$

Eq. 16 shows the relationship between $\{B_n\}$ and $\{B_1\}$. The initial position of $\{B_1\}$ is only influenced by the rotation ratio between $\{B_n\}$ and $\{B_1\}$ in case of $|D_{1n}| = 0$. The related distance dependencies of $\{B_1\}$,$\{B_n\}$ to $\{N_p\}$ can be determined by substituted vector length D_{1n} from Eq. 14 to Eq. 16.

$$\vec{D}_{1N_p} - \vec{D}_{nN_p} = \frac{B_n}{{}^{B_n}_{N_p}R^T} - \frac{B_1}{{}^{B_1}_{N_p}R^T} \tag{17}$$

The new coordinate system at origin $\{B_1\}$ can be calculated using Eq. 12. Therefore we substitute Eq. 16 into Eq. 12.

$$B_1 = {}^{B_1}_{B_n}R^T \cdot \left(B_n - \left(\frac{B_n}{{}^{B_n}_{N_p}R^T} - \frac{B_1}{{}^{B_1}_{N_p}R^T}\right)\right) \tag{18}$$

We receive the following result solving Eq. 18:

$$B_1 = B_n \cdot {}^{B_1}_{B_n}R^T \cdot \frac{\left(1 - \frac{1}{{}^{B_n}_{N_p}R^T}\right)}{\left(1 - \frac{{}^{B_1}_{B_n}R^T}{{}^{B_1}_{N_p}R^T}\right)} \tag{19}$$

We combine the rotational transformations of Eq. 19 to R_{B_1}.

$$B_1 = B_n \cdot R_{B_1} \tag{20}$$

The trajectory position information of R,T can be calculated by integrated velocity over a discrete time period. We can determine the measured acceleration a and angular velocity ω in the body frame oriented IMU. The associated determination of Euler angles in the direction cosine matrix C allows the velocity transformation from body^b to inertial^i frame [Vec22] (Eq. 21).

$$C^b \approx \begin{bmatrix} 1 & \psi & -\theta \\ -\psi & 1 & \phi \\ \theta & -\phi & 1 \end{bmatrix} \tag{21}$$

$$\Delta v^i = C^b \int_{t_n}^{t_n+\Delta} a^b_{lin}\, dt \tag{22}$$

The accelerometer integrated seismic mass affects motion-dependent special force measurements of coriolis a^b_{cl}, centrifugal a^b_{cf} and gravitational a^b_g acceleration (Eq. 24) [Cla15, Kok18, Vec22].

$$a^i_{cor} = a^b_{lin} + a^b_{cf} + a_{cl} - a^b_g \quad \{\ a^b_{cl} \ll a^b_{cf} \tag{23}$$

$$a^i_{cor} = a^b_{lin} + \omega \times v^b - R^b \cdot a^b_g \tag{24}$$

Assuming that the navigation frame is fixed on earth's

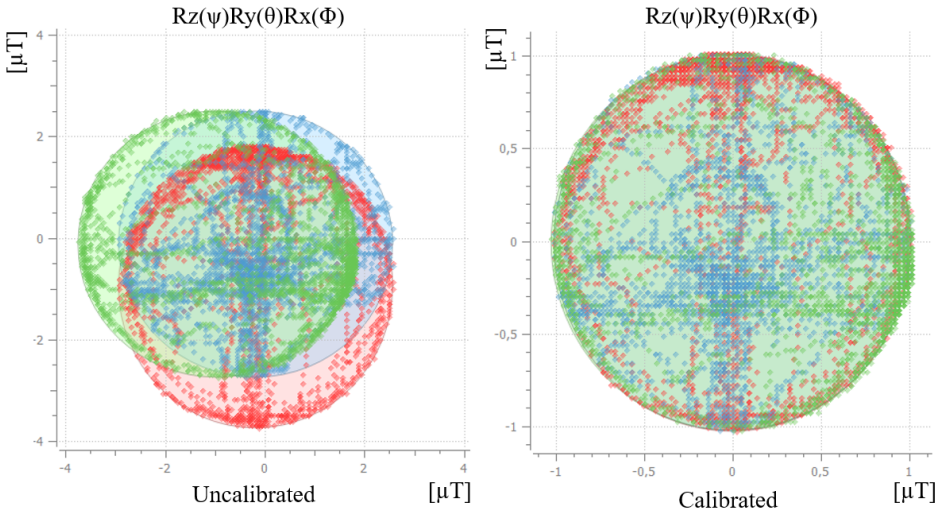


Figure 4: **Calibration of the Magnetic Field Sensor**: The magnetometer is integrated in the chassis of the ZED 2 from Stereolabs. Each colour represents a direction of rotation (red: R_ϕ, green: R_θ, blue: R_ψ).

frame position, we can derive a relation for the corrected velocity (Eq. 25) [Kok18, Vec22] .

$$\Delta v_{\text{cor}} = \Delta v^I + \Delta t \left(a_{\text{g}}^b - 2\omega_\oplus \times \int_{t_0}^{t_{\text{n}}} a^i \; dt \right) \tag{25}$$

$\omega_\oplus$ defines the earth's angular rate [Cla15]. The earth fully rotates every 23.9345 hours with an approximated rate of $\omega_\oplus = 7.29 \cdot 10^{-5}$ rad/s relative to the stars [Kok18]. The position s can be calculated from the relationships of Eq. 25.

$$\Delta v_{\text{k+1}} = \int_{t_0}^{t_{\text{n}}} a_{\text{cor}} \; dt + \Delta v_{\text{cor}} \tag{26}$$

$$\Delta s_{\text{k+1}} = \left(\iint_{t_0}^{t_{\text{n}}} a_{\text{cor}}^i \, dt \right) + \Delta t \left(\int_{t_{\text{n}}}^{(t_{\text{n}}+\Delta)} a_{\text{lin}} \, dt \right) + \frac{\Delta t}{2} \Delta v_{\text{cor}} \tag{27}$$

4 METHODOLOGY OF INITIAL SPACE MOVEMENT

Our methodological approach as shown in Fig. 5 can be classified into the steps of calibration, initialisation, transformation and locomotion.

Since we define a magnetically referenced WCS at timestep t_0, small positional fluctuations are inevitable in the presence of external influences. A general calibration is carried out for this reason. The sensor initialisation allows the determination of a specified transfer function for unit transformation $\nabla B \rightarrow \nabla s$. The space transformation converts the intrinsic sensor coordinates into the magnetically defined WCS. Using the motion equations in case of dynamic movement allows the space positional translation and rotation.

The accuracy of magnetic field sensors is affected by external temperature as well as ferro-, para- and diamagnetic influences [Cla15, Ren10, Vec22].

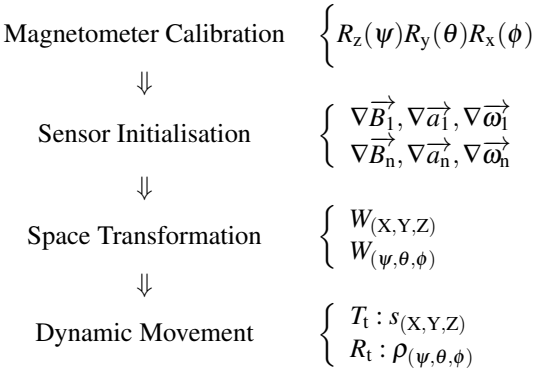


Figure 5: **Pipeline of Space Determination:** The different steps of movement transformations in a WCS.

The homogeneity of earth's magnetic field is determined by local anomalies, dipole and external fields. The earth field's detectability depends on the sensitivity of the magnetic field sensor [Cla15]. Since these effects can occur in- and outdoors, magnetometers must be calibrated. Fig. 4 demonstrates the difference between a calibrated and uncalibrated sensor. Under ideal conditions, the measured points (X,Y,Z) are located at the centre {0,0,0} of overlayed perfect spheres (RGB) since their position changes in all spatial directions. The non-calibrated state on the left side of Fig. 4 shows the inconsistent relationship between the axes (red: R_ϕ, green: R_θ, blue: R_ψ). This shift condition can be corrected by the Hard and Soft Iron calibration [Ren10, Vec22].

The magnetic disturbances and error sources can be corrected mathematically. Therefore, we consider the magnetometer B_{c} with the Eq. 21 based transformation matrix C and temperature dependent bias b_{H} to the

vector of non-calibrated magnetic field data $\tilde{B}$ [Ren10, Vec22].

$$\begin{bmatrix} B_{cx} \\ B_{cy} \\ B_{cz} \end{bmatrix} = \begin{bmatrix} C_{00} & C_{01} & C_{02} \\ C_{10} & C_{11} & C_{12} \\ C_{20} & C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} \tilde{B}_x - b_{H_0} \\ \tilde{B}_y - b_{H_1} \\ \tilde{B}_z - b_{H_2} \end{bmatrix} \quad (28)$$

Rotating the magnetometer around the gravity vector for several 360° cycles allows calibration. The sensor should inclinate between 5° and 10° [Vec22]. A successful calibration contains the measuring points of all rotation directions in a circle (Right diagram - Fig. 4).

The next step refers to the initial position of the different sensor parameters. P_{w1} (left coordinate system in Fig. 6) is at the origin of our magnetic WCS $\{B_1\}$ at the timestep t_0. The susceptibility to external influences makes the magnetic measurement data unsuitable for permanent dynamic position determination. For this reason, we transfer the magnetic unit to a metre unit (right coordinate system in Fig. 6) and initialise it with the measurement data of the gyroscope as well as the accelerometer. The initialisation and measurement fusion of all available sensors in the transformed metric system helps to reduce measurement fluctuations and influences of a single sensor.

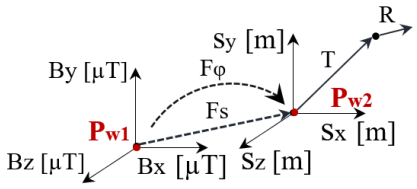


Figure 6: **Unit Transformation** $[\mu T] \rightarrow [m]$: Transformation of the magnetic field density B$[\mu T]$ into the distance $s[m]$ ($F_s \cdot \nabla B[\mu T] \rightarrow \nabla s[m]$). F_s and F_φ describes the transfer function from $[\mu T]$ to $[m]$.

The origin P_{w1}, related to $\{B_1\}$, can be calculated with Eq. 19. The IMU integrated magnetometer and gyroscope have a sensor specific step size during the spatial movement $(R|T)$. Gradient formation enables the conversion of the step size to a unit of measurement. The transfer functions of translation F_s and rotation F_φ describe the unit transformation from magnetic ∇B to distance ∇s at t:

$$F_s = \frac{\nabla s}{\nabla B} \quad \frac{[m]}{[\mu T]} \quad \begin{cases} t > 0 \\ B > 0 \end{cases} \quad (29)$$

$$F_\varphi = {}^{P_{w2}}_{P_{w1}}R^T \quad [^\circ] \quad \{ t > 0 \quad (30)$$

Since the distances between the components of gyroscope, acceleration and magnetic field sensor are sufficiently small, the coordinate systems converge ($F_s \cdot \Delta B \rightarrow \Delta s$). The initial translation and rotation values ∇s and $\nabla \varphi$ can be used approximately as starting values due to converging systems.

$$R \rightarrow \varphi[^\circ]: \quad \Delta\varphi_{(t)} = F_\varphi + \int \omega \, dt \quad (31)$$

$$T \rightarrow s[m]: \quad \Delta s_{(t)} = F_s \cdot \nabla B + \iint a \, dt \quad (32)$$

The transfer functions F_s and F_φ can be determined for each sensor (Eq. 29 and Eq. 30). This allows the sensor unit-specific transmission of measured values in the WCS. The direct sensor relations to each other can be determined. Fluctuations, noise or susceptibilities due to external influences are evident. By comparing the position data between two defined coordinate systems P_{w1} and P_{w2}, magnetic interference or sensor-specific drift deviations can be corrected.

The analysing of locomotive position sensors is an important part for the validation of dynamic matching. Therefore, we focused on the differences between static and dynamic states of multiple matched sensors and the extent of environmental influences in our methodological considerations.

Figure 7: **Experimental Setup:** Selected measurement locations (left side) and their associated depth images (right side). Top: Indoor space of a Lab; Middle: Model vehicle in a residential building; Bottom: Outside in a moving vehicle

We select different experimental locations (Fig. 7) for our validation and gather criteria (Tab. 2) such as ferromagnetic influenced building materials, lighting and colour conditions, textures as well as velocities from the challenges of previous works.

	Lab	**Model**	**Vehicle**
Velocity $[km/h]$	$0 < 3.6$	0	$0 < 80$
Location size [m]	$30 <$	$1 <$	< 2000
Magnetic influences	Indoor	Indoor	Outdoor
Texture variations	Low	Low	High
Light	Low	Average	High
Colour Contrasts	Low	Average	High

Table 2: **Criteria of Measurement:** Description of the selection criteria for different experimental locations.

	σ_{nc} $[\mu T]$	σ_{c} $[\mu T]$	$\varepsilon_{c/nc}$ [%]	Uncalibrated	Calibrated
Model:	$3.00 \pm 12.86 \cdot 10^{-3}$	$2.05 \pm 23.00 \cdot 10^{-3}$	31.67		
Lab:	$0.99 \pm 11.10 \cdot 10^{-3}$	$0.97 \pm 8.09 \cdot 10^{-3}$	2.51		
ICEV Centre:	$1.29 \pm 0.00 \cdot 10^{-3}$	$1.28 \pm 10.83 \cdot 10^{-3}$	0.35		
ICEV Front:	$6.72 \pm 3.11 \cdot 10^{-3}$	$1.52 \pm 11.48 \cdot 10^{-3}$	77.50		
EV Centre:	$2.56 \pm 0.01 \cdot 10^{-3}$	$0.96 \pm 15.39 \cdot 10^{-3}$	62.53		
EV Front:	$1.89 \pm 2.22 \cdot 10^{-16}$	$1.15 \pm 10.16 \cdot 10^{-3}$	39.27		

Table 3: **Stability of the Magnetometer:** σ_c and σ_{nc} (σ_c :calibrated, σ_{nc}: non-calibrated) describe the deviation of magnetic values during measured time t_c and t_{nc} (t_c: calibrated, t_{nc}: non-calibrated). The calibrated and uncalibrated states of the magnetometer are represents roll: red, pitch: green and yaw: blue.

5 EVALUATION

We investigate the static and dynamic behaviour of ZED 2 integrated sensors for the evaluation of geomagnetic inertial navigation and positional sensor stability. Our computing hardware has an integrated Intel Core i5 processor (@2.5GHz), 64GB RAM, external GPU1 Nvidia GeForce GTX 1050 Ti and onboard GPU0 of Intel HD Graphics 630.

Static Sensor State The stability behaviour is evaluated through revision of ZED 2 integrated magnetometer, accelerometer and gyroscope in stationary case. We consider local measurement fluctuations (Fig. 7) in calibrated and uncalibrated state to explore the suitability of geomagnetic initialisation at different times. In addition to different indoor spaces, we examine sensor behaviours in Internal-Combustion-Engine Vehicles (ICEV) and Electric Vehicle (EV). We investigate static influences on magnetometers caused by Faraday cage, electric motors, battery, active sensors, embedded graphic and combination instruments. Our static results are shown in Tab. 3.

Static Evaluation The location-dependent comparison in our evaluation show necessity of magnetometer calibration and accelerometer compension in static case. The magnetometer reveal local susceptibilities. Electromagnetic fields from devices act on the sensor at spatial locations and within the vehicles. Previous sensor positions or micro movements deviate from actual calibrated conditions between 31.67% and 77.5%. We notice in the vehicles (ICEV and EV) that magnetometers could not be calibrated at any local position due to possible magnetic fields. A strong vulnerability occurs in front of the position area where fluctuations are measurable despite calibrated states. The centre vehicle area of ICEV prove as well suitable for calibration position. We suspect that engine compartments and embedded graphic and combination instruments induce fields. The Faraday cage exhibit no direct effects in our measurements. Contrary to the ICEV, we are able to achieve stable measurement results at several static positions in the EV. Both vehicles show the consistently best results at the calibrated position. A sensor shift from the calibrated location cause a direct deterioration of position accuracy. We conclude that the magnetometer is suitable for one-time initialisation. The ideal moment is immediately after calibration. Compressing accelerometers drift and gravity at the time of magnetic calibration allows efficient positional accuracy.

Dynamic Sensor State The accuracy and transferability of multiple sensor trajectory representations is directly related to the sensor behaviour comprising stability, resolution, accuracy and speed response. Sensory noises and limitations correlate with the necessary tolerance band of trajectory positions. However, keeping within the tolerance band is an important criterion for the successful implementation of our methodology. Therefore, we compare and analyse the dynamic behaviour with the resting IMU state.

Dynamic Evaluation In resting state, the linear acceleration (Eq. 10) and angular velocity (Eq. 5) exhibit noise and offset behaviours in all directions (Fig. 8).

As a result, the double integrated acceleration after time leads to major position deviations. Adjusting the offset reduces the deviation error but does not eliminate it. By implementing a Kalman and low-pass filter, we are able to improve the noise and following position behaviour. Especially, Kalman filter is suitable for the

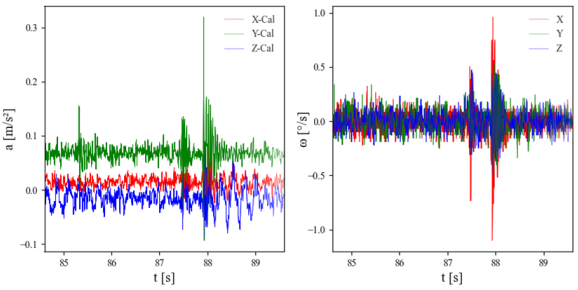


Figure 8: **Sensory Ground Truth:** The left diagram shows the acceleration. The right diagram represents the angular velocity.

fusion of several sensor signals in a dynamic system, which means that we can compensate the accelerometer signal error with the magnetometer. The section of human positional trajectory representation (Fig. 9) shows inaccuracies of sensor orientation (black arrows).

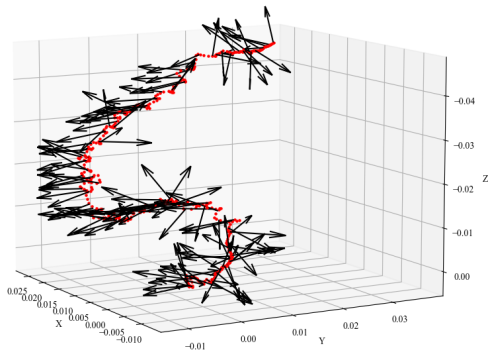


Figure 9: **Trajectory Representation of Climb Stairs:** Black arrows show the sensor orientation. Red dots represent the calculated position.

We expect that the human sinusoidal oscillation leads to a harmonic behaviour of the sensor alignment. However, our results show a non-harmonic progression. The orientation inaccuracies result from noise behaviour and sensitivity of the measured angular velocity. The use of Kalman and low-pass filters may also increase the orientation quality.

6 CONCLUSION AND FUTURE WORK

In this paper we present a concept for global and dynamic matching of multiple depth cameras using dynamic sensor data. Our motivation is a stable and verifiable inertial navigation system for in- and outdoor depth sensing. Limiting ranges and external influences such as contrast, textures, temperature, magnetic interferences and insulating materials affect the accuracy of conventional methods like GPS or image initialisation. Inaccurate position data decrease the quality of matched point clouds due to visible scattering, distortion, noise and offsets effects. Based on these challenges, we propose a concept to transfer multi-sensor data into a magnetically referenced WCS: the geomagnetic field. Global depth sensor matching allows the environmental reconstruction of individual geographic positions, while alternative navigation systems are insufficient. Geomagnetic sensor matching can be used wherever a stable external magnetic field is measured.

The positional matching of dynamic depth sensors is a promising technique for the expanding and optimising of 3D reconstructions. The suitability of global adjustment for in- and outdoor applications is based on the measurability of earth's magnetic field. A WCS can be generated by magnetic field sensors. Referencing the geomagnetic North Pole allows a direct proportionality of the magnetic field sensors. Coordinate transformations between the magnetometer and IMU can compensate the magnetic susceptibility, drifts and noise effects. The sensors show fluctuations during our in- and outdoor measurements. Combining multiple sensors reduce the position error and following offsets in merged point clouds. Calibration of magnetic field sensors increases stability of measurements despite magnetic interference sources. Future approaches can implement Kalman or low-pass filters to decrease integration-related position and orientation deviations.

The measurement radius will be extended for future evaluations of external influences and functionalities of geomagnetic matching. We will expand our data sets with different geographical locations, higher movement speeds, long-term measurements and in- and outdoor combination. Extended data sets enable us to analyse geographical sensor stability and continuous influence of dynamically variable point cloud mapping. This allows a thorough investigation of system boundaries and external influences on global depth sensor alignment.

7 REFERENCES

[Auf11] Aufderheide D., Krybus W., Dodds D., MEMS-based Smart Sensor System for Estimation of Camera Pose for Computer Vision Applications. 2011 Proceedings of the University of Bolton Research and Innovation Conference 2011 (pp. 28-29)

[Car16] Caruso D., Sanfourche M., Besnerais G., Vissiere D., Infrastructureless Indoor Navigation With an Hybrid Magneto-inertial and Depth Sensor System. 2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN), https://doi.org/10.1109/IPIN.2016.7743690

[Chu11] Chung J., Donahoe M., Schmandt C., Kim I., Razavai P., Wiseman M., Indoor Location Sensing Using Geo-Magnetism. 2011 ACM, https://doi.org/10.1145/1999995.2000010

[Cla15] Clauser C., Einführung in die Geophysik. 2015 Springer, https://doi.org/10.1007/978-3-642-04496-0

[Gar22] Garmin Ltd, https://www.garmin.com/de-DE/p/550460, last visited March 19th 2022

[Han07] Han X., Seki H., Kamiya Y., Hikizu M., Wearable Handwriting Input Device Using Magnetic Field. 2007 SICE Annual Conference, https://doi.org/10.1109/SICE.2007.4421009

[Hua19] Huang G., Visual-Inertial Navigation: A Concise Review. 2019 International Conference on Robotics and Automation (ICRA), https://doi.org/10.1109/ICRA.2019.8793604

[Kad14] Kadambi A., Bhandari A., Raskar R., 3D Depth Cameras in Vision: Benefits and Limitations of the Hardware. 2014 Springer, https://doi.org/10.1007/978-3-319-08651-4_1

[Kok18] Kok M., Hol J.D., Schön T.B., Using Inertial Sensors for Position and Orientation Estimation. 2018 Foundations and Trends in Signal Processing No. 1-2, pp 1-153, https://doi.org/10.1561/2000000094

[Mue21] Müller S., Kranzlmüller D., Dynamic Sensor Matching for Parallel Point Cloud Data Acquisition. 2021 International Conferences in Central Europe on Computer Graphics (WSCG), https://doi.org/10.24132/CSRN.2021.3101.3

[Pia13] Piatkowska E., Belbachir A.N., Gelautz M., Asynchronous Stereo Vision for Event-Driven Dynamic Stereo Sensor Using an Adaptive Cooperative Approach. 2013 International Conference on Computer Vision Workshops (ICCV Workshops) pp. 45-50, https://doi.org/10.1109/ICCVW.2013.13

[Vu12] Vu A., Ramanandan A., Chen A., Farrell J., Barth M., Real-Time Computer Vision/DGPS-Aided Inertial Navigation System for Lane-Level Vehicle Navigation. 2012 IEEE Transactions on Intelligent Transportation Systems, https://doi.org/10.1109/TITS.2012.2187641

[Ren10] Renaudin V., Afzal M., Lachapelle G., Complete Triaxis Magnetometer Calibration in the Magnetic Domain. 2010 Hindawi Publishing Corporation Journal of Sensors, https://doi.org/10.1155/2010/967245

[Shi21] Shi S., Gao T., Gao D., Ding Z., Zhang Z., Inertial navigation aid indoor navigation based on the establishment of accurate magnetic reference map. 2021 Journal of Physics: Conference Series, https://doi.org/10.1088/1742-6596/1802/4/042022

[Sir19] Sirakov N., Muge F., A System for Reconstructing and Visualising Three-dimensional Objects. 1999 Computers and Geosciences Volume 27, https://doi.org/10.1016/S0098-3004(00)00055-8

[Ste22] Stereolabs-ZED2, https://www.stereolabs.com/zed-2, last visited May 03th 2022

[Tar10] Tardif J., Goerge M., Laverne M., A New Approach to Vision-Aided Inertial Navigation. e 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, https://doi.org/10.1109/IROS.2010.5651059

[Tak15] Takimotoa R., Tsuzuki M., Vogelaar R., Martins T., Satoa A., Iwao Y., Gotohb T., Kagei S., 3D reconstruction and multiple point cloud registration using a low precision RGB-D sensor. 2015 Mechatronics, https://doi.org/10.1016/j.mechatronics.2015.10.014

[Tes22] Tesla, https://www.tesla.com/de_DE/autopilot, last visited March 19th 2022

[Vec22] Vectornav, https://www.vectornav.com/resources/inertial-navigation-primer/specifications–and–error-budgets/specs-hsicalibration, last visited March 19th 2022

[Woe13] Wöhler C., Triangulation-Based Approaches to Three-Dimensional Scene Reconstruction. 2013 Springer, https://doi.org/10.1007/978-1-4471-4150-1_1

[Yan19] Yan Y., Geneva P., Eckenhoff K., Huang G., Visual-Inertial Navigation with Point and Line Features. 2019 IEEE International Workshop on Intelligent Robots and Systems (IROS), https://doi.org/10.1109/IROS40897.2019.8967905

[Yoe21] Yeong J., Velasco-Hernandez G., Barry J., Walsh J., Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review. 2021 Sensors 2021, https://doi.org/10.3390/s21062140

[Zha16] Zhang Z., Cameras and Inertial/Magnetic Sensor Units Alignment Calibration. 2016 IEEE Transactions on Instrumentation and Measurement, https://doi.org/10.1109/TIM.2016.2518418

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Exploring the necessity of mosaicking for underwater imagery semantic segmentation using deep learning

Kazimieras Buskus
Kaunas University of Technology, Kaunas, Lithuania
kazimieras.buskus@ktu.edu

Evaldas Vaiciukynas
Kaunas University of Technology, Kaunas, Lithuania
evaldas.vaiciukynas@ktu.lt

Saule Medelyte
Klaipeda university, Klaipeda, Lithuania
saule.medelyte@ku.lt

Andrius Siaulys
Klaipeda university, Klaipeda, Lithuania
andrius.siaulys@jmtc.ku.lt

ABSTRACT

Deep learning applications are attracting considerable interest nowadays and image analysis pipelines are no exception. Benthic studies often rely on the subjective evaluation of video material recorded using underwater drones. The demand for automatic image segmentation and quantitative evaluation arises due to the large volume of video data collected. This study performed a semantic segmentation task by training the PSPNet architecture with ResNet-34 backbone for 50 epochs using imagery prepared by simply extracting a few video frames or stitching a multitude of frames into a large 2D mosaic. Mosaicking is a particularly resource-intensive step, therefore, the possibility to skip such preprocessing would result in a more rapid analysis. The effect on the resulting segmentation quality was investigated by estimating the seabed coverage of three classes (*Furcellaria lumbricalis, Mytilus edulis trossulus*, and boulders) in a video material obtained from the Baltic Sea. Segmentation success, measured by intersection over union, varied between 0.56 and 0.84, usually slightly better for frames than for the mosaic overall. Absolute differences in estimated coverage were negligible (mosaic vs. frames): 0.24% vs. 1.26% for furcellaria, 0.44% vs. 2.46% for mytilus, and 4.02% vs. 2.06% for boulders. Due to the differences between predicted coverage and the mosaic-based ground truth being in an acceptable range, the findings suggest that the mosaicking step could be safely skipped in favor of a few equally spaced sample frames.

Keywords

underwater imagery, mosaicking, semantic segmentation, deep learning, PSPNet, ResNet, Baltic sea

1 INTRODUCTION

Maritime space is increasingly used for renewable energy installations, oil and gas exploitation, naval shipping and fishing, ecosystem monitoring and biodiversity conservation, aquaculture production, and many other purposes. The demand for maritime space requires integrated planning and management strategies focused on solid scientific knowledge and reliable seabed mapping [Men20], with underwater images [Urr21] being one of the most widely used seabed mapping input sources. The key advantage of underwater imagery is its simplicity, enabling the rapid collection of vast amounts of data, especially through the use of underwater drones and hence cost-effectiveness. Unfortunately, only a small part of the information stored in these image archives is being extracted due to labor-intensive and time-consuming analysis procedures. A promising way to process large amounts of images is computer-aided analysis, i.e., converting seabed video to 2D mosaic maps, image segmentation, and quantifying segmentation results. However, the mosaicking step is computationally demanding, and the resulting photomosaic often differs with respect to the tool used (see Fig. 1). A comprehensive list of various, primarily commercial, tools was evaluated for airborne imagery by [Son16] with Pix4DMapper found to be the most precise and Autostich the fastest.

Automatic segmentation of underwater imagery, compared to other types of images, is a new and challenging direction of research. According to a survey [Gra17] the first publications on seabed segmentation task (also termed seafloor classification) appeared 25 years ago and are still scarce, the common ground between them being the use of "hand-crafted" image features and traditional machine learning algorithms, for example, random forest [Rim18]. New deep learning architectures of neural networks could replace image features and help analyze images more effectively, accurately

(a) CCOM [Rzh06] (b) LAPM v1 [Mar13] (c) Microsoft ICE 2 (d) ArcSoft PM 6 (e) Sample frames

Figure 1: Converting underwater video material (from Atlantic Ocean) to images: marine mosaics for benthic studies (a-b), commercial software mosaics for panoramic view (c-d), and sample of equally spaced frames (e).

and quickly than ever before. The initial efforts to apply deep learning to underwater images concerned corals [Alo19] and other broad categories [Liu20, Isl20] (such as fish, plants, divers, and stones) without preoccupation with the sea floor and, therefore, without the need to stitch images through mosaicking with the goal of reconstructing a precise floor map for the estimation of biologically valid organism counts or visual coverage.

The most similar related works that use deep learning and convolutional architectures to explicitly segment seabed images are recent techniques for estimating the coverage of seagrass [Mar18, Wei19, Bur20], macroalgae [Bal20], and kelp [Mah20]. The discrepancy between the expert-identified and model-predicted coverage in absolute percentage points was between 0.54% and 9.88% for kelp [Mah20] species (see Table 6 there), while the remaining mentioned seabed studies reported only segmentation accuracy without a comparison of the resulting coverage percentages.

This study evaluates both semantic segmentation success and the resulting coverage estimates using seabed imagery in the form of 2D mosaics, or several sample frames from short 30 s video transects as input to a convolutional architecture deep learning model. Both mosaics, as in Fig. 1 (a), and frames, as in Fig. 1 (e), were annotated by marine biologists to solve the detection task of 2 biological species – *Furcellaria lumbricalis* and *Mytilus edulis trossulus* – and 1 geological class – boulders. Performed experiments use mosaic-based annotations as a golden standard to measure how much coverage summaries differ when skipping a resource-intensive mosaicking step in favor of a few sample frames from an underwater video.

2 UNDERWATER IMAGERY

The underwater material was filmed in coastal and offshore reefs of Lithuanian marine waters of the south-eastern Baltic Sea. In the coastal area, the video was taken by scuba divers at 4–7 meters depth using a hand-held video camera with 1920×1080 resolution along multiple 10–meter transects (designations SM_07_1, SM_07_2, SM_08_1A, SM_08_1B). Additionally, in the offshore region at 35 – 40 meters depth, a remotely operated vehicle (ROV) equipped with a vertically mounted camera (3 CCD, high-quality Leica Dicomar lenses, and 10× optical zoom) with 1920×1080 resolution and a lighting system consisting of 16 LED

in 4 × 4 array was deployed, and two 30 second long transects (designations Denoflit_30s and Denoflit_2) were filmed. The raw video material was later transformed into 2D mosaics or several relevant frames were picked for each transect.

In this study, the video mosaicking method, developed by the Center for Coastal and Ocean Mapping [Rzh06], was used. The method consists of the following steps:

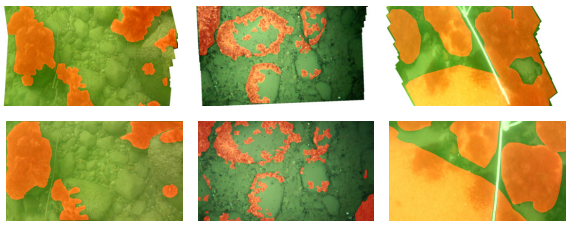
1. To reduce processing time, video transects of 30 s had frame rate reduced from 50 to 5 fps and frame size from 1920×1080 to 960×540.
2. The roll and pitch of the filming platform were adjusted by image transformations and some video enhancements were applied to each frame.
3. The enhanced footage was subjected to automatic frame-to-frame pair-wise registration (a method that calculates the overlap of neighboring frames).
4. Using the pair-wise registration data from the previous step, video mosaics were constructed from non-enhanced video.

For an alternative approach, experts assigned a specific number of frames, typically 12-16, for each video transect, and equally spaced frames of 960×540 size were extracted using a command-line tool *ffmpeg*. The aim of several representative sample frames was to cover the video material with the least overlap between frames to roughly correspond with the mosaic's visual scope. Summary of the prepared image data and classes represented in the images is in Table 1, where the size of the resulting 2D mosaics and the corresponding number of representative frames (of 960×540 resolution) can be compared.

The prepared imagery (large mosaics and many fixed-size frames) was annotated by two marine biologists (without overlap) using polygons and striving for pixel-level accuracy in an online collaborative annotation platform Labelbox [Rie20]. Examples of these annotations for the mosaic segment and the corresponding frame are shown in Fig. 2.

Transect	Mosaic size	Frames	Classes
SM07_1	2434×8774	16	*Furcellaria* Boulder
SM07_2	5021×5107	12	*Furcellaria*
SM08_1A	4191×5379	12	*Furcellaria* Boulder
SM08_1B	4745×5379	12	*Furcellaria* Boulder
Denoflit_2	2434×8774	11	*Mytilus* Boulder
Denoflit_30s	1580×5480	11	*Mytilus*

Table 1: Summary of underwater imagery used.



(a) *Furcellaria* (b) *Mytilus* (c) Boulders

Figure 2: Mosaic (*top*) and frame (*bottom*) annotations for (roughly) the same transect region.

In general, all visible objects of 3 classes were annotated: biological *Furcellaria* species had 102 instances in 3 mosaics and 120 instances in frames; biological *Mytilus* species had 148 instances in 2 mosaics and 62 in frames; geological Boulder class had 167 instances in 3 mosaics and 166 in frames.

3 METHODS

The prepared underwater imagery, either in the form of large mosaics or representative frames, was patched using a sliding window idea to prepare training and testing data for the deep learning model with convolutional architecture. Training patches were augmented to increase data amount and as a simple form of regularization. The evaluation was performed by splitting the transects in half to achieve a 2-fold cross-validation. Segmentation success was measured by the intersection over union metric and by comparing visual coverage estimates.

3.1 Image patching and augmentation

Due to the limitations of the available computational resources, both mosaics and frames were sliced into overlapping 288×288 size patches. The overlap was the result of sliding window or block processing idea with vertical and horizontal strides of 144 pixels (see Fig. 3). The mosaics contained many white pixels, as a result of the mosaicking process, so only patches with a minimum of 70% non-white pixels were considered as input images. Furthermore, to increase the amount of training data, a few traditional augmentation techniques, such as vertical and horizontal flip, and one marine-specific technique – removal of water scattering (RoWS) [Cha10] – were used on input image patches.

3.2 Convolutional architecture

In the experiments, we used a deep convolutional neural network with pyramid spatial pooling architecture - PSPNet [Zha17] model with ImageNet pre-trained ResNet-34 [He16] as the backbone. The model was implemented using the *Keras* framework (version 2.3.1),

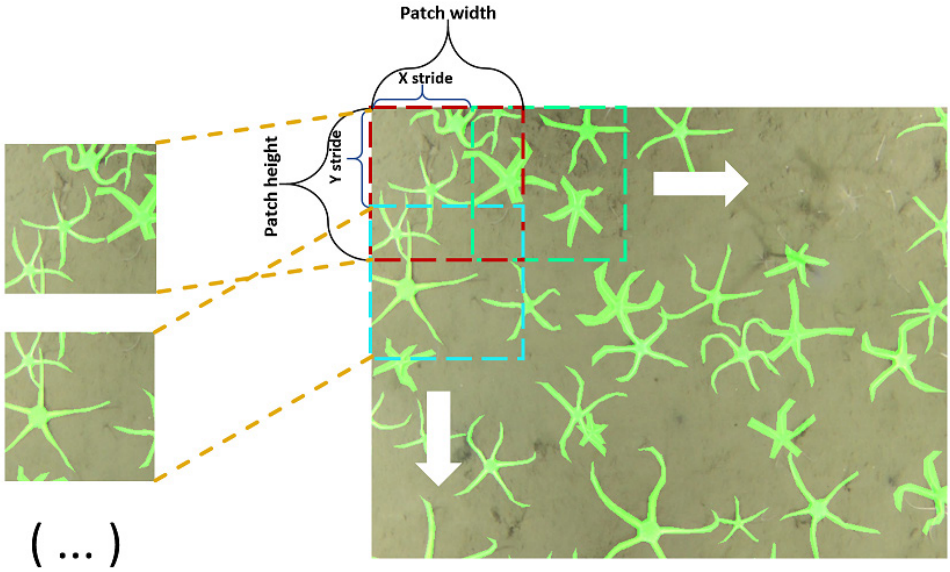


Figure 3: Block processing of underwater photos into overlapping patches of 288×288 sized input images.

running on the *Tensorflow* back-end (version 2.1.0), with the help of the package *segmentation-models* (version 1.0.1) [Yak19]. Model training settings were as follows: batch size (number of images used for the weight tuning step) - 8 images, training duration - 50 epochs, downsampling rate (backbone depth in the PSPNet model) - 8, minimized loss - additive combination of Jaccard [Jac08] and Focal [Lin17] losses. Final model had 2 746 058 parameters (2 737 860 trainable).

3.3 Evaluation of segmentation

Evaluation of segmentation performance was done using 2-fold transect-stratified cross-validation (CV), where each transect was split in half – top and bottom parts – and training was performed on one part while testing on the other part. For example, after training on all bottom parts of mosaics (first half of the corresponding frame set), testing was performed on all top parts and vice versa. Both training and testing were carried out using 288×288 size images and after obtaining results on all mosaic (or frames) patches tested, overlapping parts were averaged and a class threshold of 0.5 was used to obtain the final predicted mask. Then the intersection over union (IOU) – a commonly used measure of segmentation success, comparing the ground truth with the predicted mask – was used to summarize the results of both testing folds of 2-fold CV in a micro-average fashion. In addition, final prediction masks were used to estimate the visual coverage of the class in question. The coverage itself was interpreted as a ratio between predicted or the ground truth masks with either relevant pixels (excluding white pixels) in the mosaic setting and all pixels in the frame setting, see the Pixels column in Tables 2– 4.

4 EXPERIMENTS

We used 2D mosaics and representative transect frames for training and testing the convolutional neural network model for two biological and one geological classes.

4.1 Setup

The hardware used was as follows: Intel(R) Core(TM) i7-8700 CPU @3.2 GHz, 32 GB of operating memory, NVIDIA RTX 2070 with 8 GB of memory. Software used: Windows 10 Enterprise (build 1809) 64-bit operating system, CUDA 10.1, CuDNN 6.4.7 and Python 3.6.8. During training, 4608 patches were used for *Furcellaria* class mosaic setting, 4200 for frame settings; *Mytilus* class - 1944 for mosaic, 1848 for frames; Boulder - 9810 mosaic, 8652 patches in frame settings.

4.2 Results

Segmentation performance. Segmentation performance was summarized using the intersection over union (IOU) metric. The amount of imagery used in experiments is reported as Pixels column in Tables 2 – 4, with frames usually having slightly fewer pixels overall, except for *Furcellaria* class transects. Results demonstrate that the best achieved IOU score was 84% using mosaics for *Furcellaria* class segmentation (see Table 2). The IOU results between mosaics and frames indicated slight differences in all but one case for *Furcellaria* class, where the absolute difference in IOU scores was almost 10%. Also, in two out of four transects for *Furcellaria* class, the frames had a better IOU score (by 1.04% and 1.5% points). For *Mytilus* class (see Table 3), frames had marginally better IOU results (by 2.8% and 6.1% points). However, mosaics appeared to be more advantageous for the geological

Furcellaria transects	Mosaic		Frames		
	Pixels (mln.)	IOU	Pixels (mln.)	IOU	Δ IOU
SM07_1_bio	9.04	0.703	8.29	0.718	-0.015
SM07_2_bio	7.06	0.839	6.22	0.793	0.046
SM08_1A_bio	6.64	0.661	6.22	0.765	-0.104
SM08_1B_bio	6.85	0.726	6.22	0.726	0
Totals:	29.59	0.711	26.96	0.746	-0.035

Table 2: Segmentation performance, as measured by the IOU score, for *Furcellaria* class using mosaics and frames.

Mytilus transects	Mosaic		Frames		
	Pixels (mln.)	IOU	Pixels (mln.)	IOU	Δ IOU
Denoflit_2_bio	6.56	0.560	5.70	0.621	-0.061
Denoflit_30s_bio	5.17	0.671	5.70	0.699	-0.028
Totals:	11.73	0.613	11.40	0.6649	-0.051

Table 3: Segmentation performance, as measured by the IOU score, for *Mytilus* class using mosaics and frames.

Boulder transects	Mosaic		Frames		
	Pixels (mln.)	IOU	Pixels (mln.)	IOU	Δ IOU
Denoflit_2_geo	6.56	0.661	5.70	0.592	0.070
SM07_1_geo	9.04	0.606	8.29	0.603	0.003
SM08_1A_geo	6.63	0.819	6.22	0.798	0.021
SM08_1B_geo	6.85	0.802	6.22	0.808	-0.006
Totals:	29.09	0.744	26.44	0.733	0.011

Table 4: Segmentation performance, as measured by IOU, for Boulder class using mosaics and frames.

boulder class (see Table 4), with a maximum difference of 7% points. Overall, total IOU scores were higher for frames than mosaics by 3.5% for *Furcellaria* and 5.1% for *Mytilus* class (see Totals in Tables 2 and 3), except for the boulder class (see Totals in Table 7) with 1% point difference.

Segmentation totals. When summarizing the segmentation performance by the Totals in Tables 2–4 we can make the following insights. Segmentation success, measured by the IOU metric, was in an acceptable range between 61.3% (for *Mytilus* mosaics) and 74.6% (for *Furcellaria* frames). Segmentation performance using frames was better for biological classes (by 3.5% for *Furcellaria* and 5.1% for *Mytilus* transects), but slightly worse for the geological boulder class (by 1.1%).

Coverage estimates. With respect to visual coverage estimates, it is important to note that for all but the *Mytilus* class, transects have very different coverage levels: for example, transect *SM07_1_bio* had 9.75% while transect *SM08_1B_bio* had 57.48% coverage for the *Furcellaria* class (see the Mosaic GT column in Table 5). Three deltas summarize the results of visual coverage estimates in Tables 5–7, measuring the absolute difference in coverage from the mosaic ground truth: the first delta (Mosaic Δ DL) shows the effect of using predictions from mosaics, the second delta (Frames Δ GT) shows the effect of using annotated frames instead of mosaics without any prediction, and the third delta (Frames Δ DL) shows the effect of using predictions from frames. Biological classes had negligible differences when using mosaic predictions with an underestimate of 0.86% and an overestimate of 0.41%, while geological class had greater differences with an underestimate of 1.61% and an overestimate of 6.12%. Surprisingly, even annotators were unable to achieve a high correspondence in visual coverage estimates with slight differences for geological class and more considerable differences for biological classes, even reaching an underestimate of 8.24% points for *Furcellaria* transect with the highest coverage (transect *SM07_2_bio*). The last and most important differences in our study were observed using frames predictions with an underestimate of 5.89% and an overestimate of 8.41%, both for the *Furcellaria* class. The differences for other classes were distributed relatively uniformly in that range. There was a tendency to overestimate geological class and underestimate biological classes when estimating visual coverage from frame predictions.

Coverage totals. When summarizing the coverage estimates by the Totals in Tables 5–7 we can make the following insights. Taking mosaics as the ground truth,

Furcellaria transects	Mosaic			Frames			
	GT	DL	Δ DL	GT	Δ GT	DL	Δ DL
SM07_1_bio	9.75	9.44	0.31	8.32	1.43	9.01	0.74
SM07_2_bio	11.91	11.05	0.86	9.88	2.03	9.25	2.66
SM08_1A_bio	43.15	43.83	-0.68	47.56	-4.41	49.04	-5.89
SM08_1B_bio	57.48	57.07	0.41	49.23	8.24	49.07	8.41
Totals:	28.81	28.57	0.24	27.18	1.63	27.55	1.26

Table 5: Coverage estimates for *Furcellaria* class. Abbreviations: GT – results from ground truth annotations; DL – results from deep learning model-based predictions; Δ – difference from mosaic-wise ground truth annotations.

Mytilus transects	Mosaic			Frames			
	GT	DL	Δ DL	GT	Δ GT	DL	Δ DL
Denoflit_2_bio	18.11	17.92	0.20	15.25	2.86	16.25	1.86
Denoflit_30s_bio	22.83	22.07	0.76	21.61	1.21	19.21	3.62
Totals:	20.19	19.75	0.44	18.43	1.76	17.73	2.46

Table 6: Coverage estimates for *Mytilus* class. Abbreviations: GT – results from ground truth annotations; DL – results from deep learning model-based predictions; Δ – difference from mosaic-wise ground truth annotations.

Boulder transects	Mosaic			Frames			
	GT	DL	Δ DL	GT	Δ GT	DL	Δ DL
Denoflit_2_geo	29.65	28.05	1.61	30.62	-0.97	25.84	3.82
SM07_1_geo	34.95	41.07	-6.12	34.52	0.43	36.24	-1.29
SM08_1A_geo	76.41	81.95	-5.54	77.05	-0.64	81.17	-4.77
SM08_1B_geo	76.55	81.70	-5.15	76.72	-0.17	80.90	-4.35
Totals:	53.01	57.03	-4.02	53.61	-0.60	55.08	-2.06

Table 7: Coverage estimates for boulder class. Abbreviations: GT – results from ground truth annotations; DL – results from deep learning model-based predictions; Δ – difference from mosaic-wise ground truth annotations.

we can notice lower coverage for biological classes (28.81% for *Furcellaria* and 20.19% for *Mytilus*) than for the geological class (53.01% for boulders). Expert-based annotations of frames deviated from the annotations of mosaics only slightly: more for biological classes (by 1.63% for *Furcellaria* and by 1.76% for *Mytilus*) and less for the geological class (by 0.6% for boulders). Model-based predictions when training on the transects' bottom half and testing on the top half (or vice versa), if compared to previously mentioned deviations of expert-based frames annotations, deviated only slightly more with underestimates of 1.26% for *Furcellaria* and 2.46% for *Mytillus* classes and an overestimate of 2.06% for boulders class.

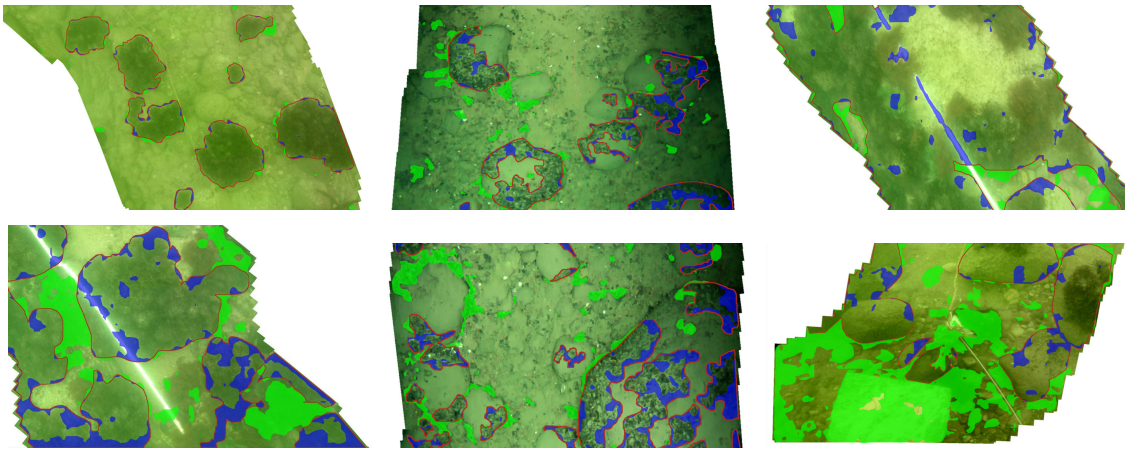
Visual comparison. Examples of segmentation predictions are provided in Figures 4–5, where we can compare how poor results differ from acceptable with segmentation false positives/negatives in green/blue colors.

5 CONCLUSIONS

Segmentation success was better than average with intersection over union varying between 0.56 and 0.84, depending on the class and transect, but slightly better for frames than for the mosaics overall. Lower visual coverage estimates from ground truth mosaic annotations were for biological classes (28.81% for *Furcellaria* and 20.19% for *Mytilus*) than for the geological class (53.01% for boulders).

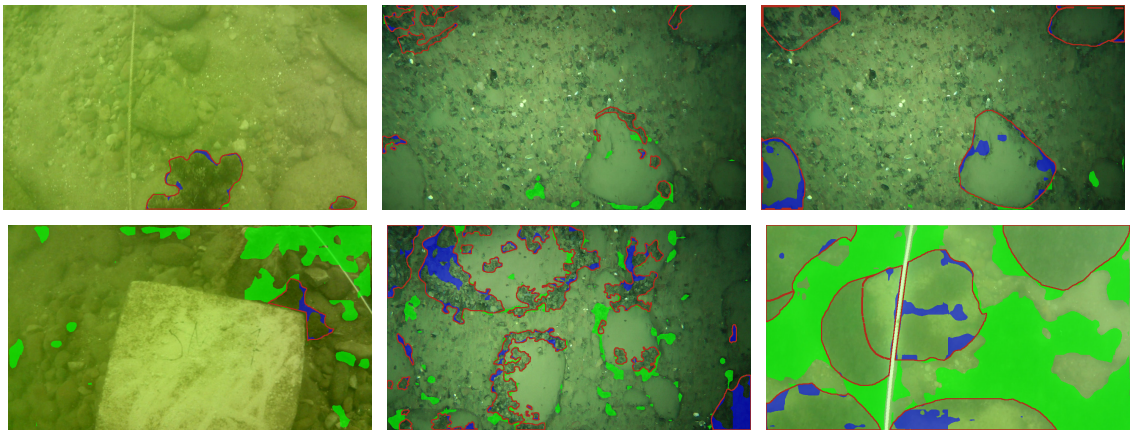
Expert-based annotations of frames deviated from the annotations of mosaics only slightly (largest deviation of 1.76%), model-based predictions - a bit more (largest deviation of 2.46%). The largest differences were observed for *Mytilus* class, which was composed of many tiny objects and had the lowest coverage. Due to the deviations being in an acceptable range, the reported results of visual coverage estimates suggest that the mosaicking step could be safely skipped in favour of a few equally spaced sample frames.

The main limitation of the experiments performed is the balance between training and testing data amounts, and the transect-stratified type of cross-validation, ensuring that the model has seen part of the tested transect during training. Future work should address these limitations.



(a) *Furcellaria* (b) *Mytilus* (c) Boulders

Figure 4: Segmentation success using mosaics: acceptable (*top*) and poor (*bottom*) results. Color coding: false negative pixels are marked in blue, false positive pixels in green, and ground truth annotations are outlined in red.



(a) *Furcellaria* (b) *Mytilus* (c) Boulders

Figure 5: Segmentation success using frames: acceptable (*top*) and poor (*bottom*) results. Color coding: false negative pixels are marked in blue, false positive pixels in green, and ground truth annotations are outlined in red.

6 ACKNOWLEDGMENTS

This work was supported by the project DEMERSAL "A deep learning-based automated system for seabed imagery recognition" (funded by the Research Council of Lithuania under the agreement No. P-MIP-19-492). The authors thank the Labelbox [Rie20] team.

REFERENCES

[Alo19] Alonso, I., Yuval, M., Eyal, G., Treibitz, T., and Murillo, A. C. CoralSeg: Learning coral segmentation from sparse annotations. Journal of Field Robotics 36, No. 8, 2019, pp. 1456–1477. doi:10.1002/rob.21915.

[Bal20] Balado, J., Olabarria, C., Martínez-Sánchez, J., Rodríguez-Pérez, J. R., and Pedro, A. Semantic segmentation of major macroalgae in coastal environments using high-resolution ground imagery and deep learning. International Journal of Remote Sensing 42, No. 5, 2020, pp. 1785–1800. doi:10.1080/01431161.2020.1842543.

[Bur20] Burguera, A. Segmentation through patch classification: A neural network approach to detect Posidonia oceanica in underwater images. Ecological Informatics 56, 2020, p. 101053. doi:10.1016/j.ecoinf.2020.101053.

[Cha10] Chao, L. and Wang, M. Removal of water scattering. In: 2010 2nd International Conference on Computer Engineering and Technology (ICCET). 2, 2010, pp. 2–35. doi:10.1109/ICCET.2010.5485339.

[Gra17] Gracias, N., Garcia, R., Campos, R., Hurtos, N., Prados, R., Shihavuddin, A., Nicosevici, T., Elibol, A., Neumann, L., and Escartin, J. Application Challenges of Underwater Vision. In: Computer Vision

in Vehicle Technology. 2017, pp. 133–160. doi:10.1002/9781118868065.ch7.

[He16] He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In: 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.

[Isl20] Islam, M. J., Edge, C., Xiao, Y., Luo, P., Mehtaz, M., Morse, C., Enan, S. S., and Sattar, J. Semantic Segmentation of Underwater Imagery: Dataset and Benchmark. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2020. doi:10.1109/iros45743.2020.9340821.

[Jac08] Jaccard, P. Nouvelles Recherches Sur la Distribution Florale. Bulletin de la Societe Vaudoise des Sciences Naturelles 44, 1908, pp. 223–70. doi:10.5169/seals-268384.

[Lin17] Lin, T., Goyal, P., Girshick, R., He, K., and DollÃ¡r, P. Focal Loss for Dense Object Detection. In: 2017 IEEE International Conference on Computer Vision (ICCV). 2017, pp. 2999–3007. doi:10.1109/ICCV.2017.324.

[Liu20] Liu, F. and Fang, M. Semantic Segmentation of Underwater Images Based on Improved Deeplab. Journal of Marine Science and Engineering 8, No. 3, 2020, p. 188. doi:10.3390/jmse8030188.

[Mah20] Mahmood, A., Ospina, A. G., Bennamoun, M., An, S., Sohel, F., Boussaid, F., Hovey, R., Fisher, R. B., and Kendrick, G. A. Automatic Hierarchical Classification of Kelps Using Deep Residual Features. Sensors 20, No. 2, 2020, p. 447. doi:10.3390/s20020447.

[Mar13] Marcon, Y., Sahling, H., and Bohrmann, G. LAPM: a tool for underwater large-area photo-mosaicking. Geoscientific Instrumentation, Methods and Data Systems 2, No. 2, 2013, pp. 189–198. doi:10.5194/gi-2-189-2013.

[Mar18] Martin-Abadal, M., Guerrero-Font, E., Bonin-Font, F., and Gonzalez-Cid, Y. Deep Semantic Segmentation in an AUV for Online Posidonia Oceanica Meadows Identification. IEEE Access 6, 2018, pp. 60956–60967. doi:10.1109/access.2018.2875412.

[Men20] Menandro, P. S. and Bastos, A. C. Seabed Mapping: A Brief History from Meaningful Words. Geosciences 10, No. 7, 2020, p. 273. doi:10.3390/geosciences10070273.

[Rie20] Rieger, B., Rasmuson, D., and Sharma, M. Labelbox: the leading training data platform for data labelling. 2020. URL: http://labelbox.com.

[Rim18] Rimavičius, T., Gelžinis, A., Verikas, A., Vaičiukynas, E., Bačauskienė, M., and Šaškov, A. Automatic benthic imagery recognition using a hierarchical two-stage approach. Signal, Image and Video Processing 12, No. 6, 2018, pp. 1107–1114. doi:10.1007/s11760-018-1262-4.

[Rzh06] Rzhanov, Y., Mayer, L., Beaulieu, S., Shank, T., Soule, S., and Fornari, D. Deep-sea Geo-referenced Video Mosaics. In: OCEANS 2006. 2006. doi:10.1109/oceans.2006.307018.

[Son16] Song, H., Yang, C., Zhang, J., Hoffmann, W. C., He, D., and Thomasson, J. A. Comparison of mosaicking techniques for airborne images from consumer-grade cameras. Journal of Applied Remote Sensing 10, No. 1, 2016, p. 016030. doi:10.1117/1.jrs.10.016030.

[Urr21] Urra, J., Palomino, D., Lozano, P., González-García, E., Farias, C., Mateo-Ramírez, Á., Fernández-Salas, L. M., López-González, N., Vila, Y., Orejas, C., Puerta, P., Rivera, J., Henry, L.-A., and Rueda, J. L. Deep-sea habitat characterization using acoustic data and underwater imagery in Gazul mud volcano (Gulf of Cádiz, NE Atlantic). Deep Sea Research Part I: Oceanographic Research Papers 169, 2021, p. 103458. doi:10.1016/j.dsr.2020.103458.

[Wei19] Weidmann, F., Jager, J., Reus, G., Schultz, S. T., Kruschel, C., Wolff, V., and Fricke-Neuderth, K. A Closer Look at Seagrass Meadows: Semantic Segmentation for Visual Coverage Estimation. In: OCEANS 2019 - Marseille. 2019. doi:10.1109/oceanse.2019.8867064.

[Yak19] Yakubovskiy, P. Segmentation Models. GitHub repository, 2019. 2019. URL: https://github.com/qubvel/segmentation_models.

[Zha17] Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. Pyramid Scene Parsing Network. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017, pp. 6230–6239. doi:10.1109/CVPR.2017.660.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Mip-NeRF RGB-D: Depth Assisted Fast Neural Radiance Fields

Arnab Dey
I3S-CNRS/Université Côte d'Azur
Sophia-Antipolis, France
adey@i3s.unice.fr

Yassine Ahmine
I3S-CNRS/Université Côte d'Azur
Sophia-Antipolis, France
ahmine@i3s.unice.fr

Andrew I. Comport
I3S-CNRS/Université Côte d'Azur
Sophia-Antipolis, France
Andrew.Comport@cnrs.fr

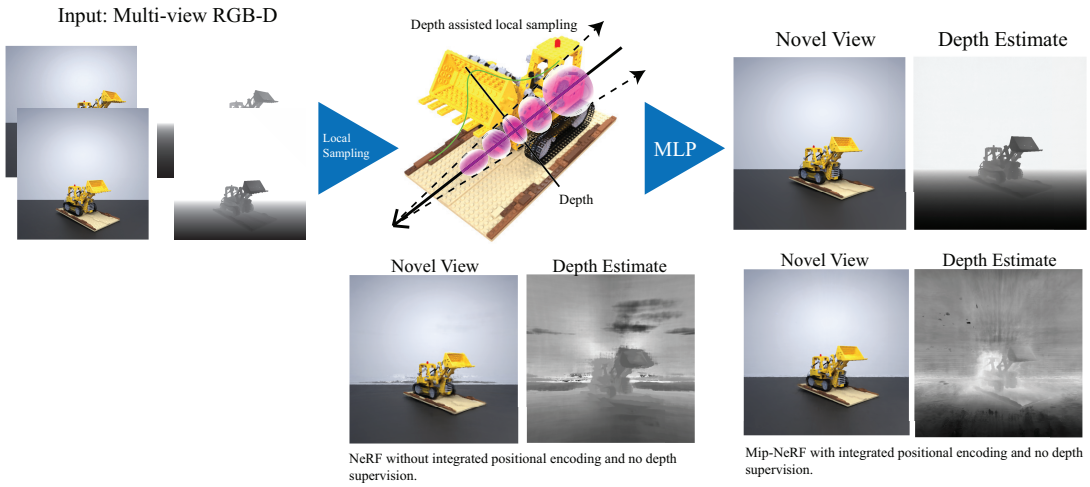


Figure 1: Mip-NeRF RGB-D uses RGB-D frames to represent 3D scenes using neural radiance fields. Depth information is used for local sampling and geometric loss. It produces significantly better photometry and geometry.

ABSTRACT

Neural scene representations, such as Neural Radiance Fields (NeRF), are based on training a multilayer perceptron (MLP) using a set of color images with known poses. An increasing number of devices now produce RGB-D(color + depth) information, which has been shown to be very important for a wide range of tasks. Therefore, the aim of this paper is to investigate what improvements can be made to these promising implicit representations by incorporating depth information with the color images. In particular, the recently proposed Mip-NeRF approach, which uses conical frustums instead of rays for volume rendering, allows one to account for the varying area of a pixel with distance from the camera center. The proposed method additionally models depth uncertainty. This allows to address major limitations of NeRF-based approaches including improving the accuracy of geometry, reduced artifacts, faster training time, and shortened prediction time. Experiments are performed on well-known benchmark scenes, and comparisons show improved accuracy in scene geometry and photometric reconstruction, while reducing the training time by 3 - 5 times.

Keywords

Computer vision, RGB-D NeRF, NeRF, Neural Scene Representation, Neural Rendering, Volume Rendering.

1 INTRODUCTION

Recent advances in neural scene representations [Sitzmann et al., 2019, Mildenhall et al., 2020] have demonstrated that neural networks can be used to represent 3D scenes as weights of a neural network for the purpose of rendering novel photorealistic views. Methods such as [Mildenhall et al., 2020, Saito et al., 2019, Lombardi et al., 2019] learned a volumetric representation from a sparse set of RGB images captured from color camera sensors. This method requires pre-

computation of camera poses and uses two multilayer perceptron networks to represent scene geometry and lighting effects. Although the NeRF models and their variants have shown impressive results, the underlying model is computationally inefficient, largely due to its volumetric search space for intersecting viewing rays, leading to extended training times. For example, volume rendering involves sampling points along each viewing ray (256 for NeRF) to calculate the color of the ray from the volume density and radiance of each sample point. Furthermore, the multiview triangulation problem is sometimes intractable from only images, which leads to artifacts and inaccurate geometry.

Although color-only approaches work well for applications that only have RGB images available, this approach can be improved by considering depth information alongside color. Many devices, including mobile phones, now include RGB-D sensors, and the aim of this paper is to investigate and devise a methodology to incorporate depth information into a neural scene representation.

Only a few methods have been proposed to take advantage of depth measurements simultaneously with color within the volumetric rendering pipeline [Neff et al., 2021, Deng et al., 2021]. However, these methods do not explicitly model the uncertainty of the sensor. To successfully incorporate noisy depth measurements into the volumetric rendering pipeline, the recent Mip-NeRF approach [Barron et al., 2021] provides a framework that accounts for the uncertainty of color pixel with varying depth by replacing classic 3D ray sampling with conic region sampling. This approach provides an elegant framework for including multivariate Gaussian uncertainty and will be extended in this paper to include depth uncertainty.

In this paper, it will be demonstrated that considering depth information can improve geometry considerably compared to only color information at several different levels. First, this method shows how local sampling along the rays, guided by surface information from RGB-D frames, can reduce the number of samples along the ray and replace the coarse network of NeRF. Second, a joint color-and-depth-loss term will be shown to allow the network to learn the geometry and color of the scene from a limited number of input views. Third, the proposed method shows how depth uncertainty can be incorporated into a multivariate Gaussian method to query the MLP. Finally, an adaptive training method will be proposed that allows the network to learn multiple scales of uncertainty within the representation.

To sum up, the proposed method is based on a RGB-D neural radiance field combined with an implicit occupancy representation that takes into account both color and depth observations.

The key contributions of the article are summarized as follows:

- Depth information is used for efficient sampling.
- The representation is optimized simultaneously on scene geometry and photometry.
- Depth uncertainty is handled adaptatively via a new local sampling strategy.

2 RELATED WORK

The proposed Mip-NeRF RGB-D uses a set of RGB-D inputs to learn a volumetric scene representation of the observed scene using a multilayer perceptron by leveraging both depth and color information. In the following, related work to this research will be discussed.

2.1 *Novel view synthesis from images*

Image-based view synthesis uses a number of techniques to generate novel images, such as transforming or warping an existing set of images using estimated geometry and camera poses to create novel views [Hedman et al., 2016, Gortler et al., 1996]. [Heigl et al., 1999] used a sequence of images and directly rendered the views by projective mapping of all images to a common plane of mean geometry. To generate a novel view from a set of captured images of different poses requires blending them to target views; even though the geometry of the static objects is constant in different views, the appearance can change depending on lighting and object properties. To overcome these drawbacks [Hedman et al., 2018, Thies et al., 2020] used artificial neural networks to reduce artifacts and view-dependent effects in the generated novel views.

2.2 *Implicit neural surface representation*

These methods use neural networks to learn a neural surface representation of the object using voxels, meshes, and point cloud data. Although they are capable of achieving impressive results, they are limited by their internal resolution and high-frequency details. Mescheder et al. [Mescheder et al., 2019] used a neural network to learn a continuous 3D occupancy function; Given 3D points as input to an occupancy network, the network predicts binary occupancy at that 3D location. Later, [Chen and Zhang, 2019] used an MLP to predict occupancy from a feature vector and the 3D coordinates of the location. On the other hand, [Park et al., 2019] learned a signed distance function(SDF) instead of occupancy to improve the quality of the reconstruction. [Saito et al., 2019] showed that it is possible to infer 3D surfaces and texture from a single image using an implicit function.

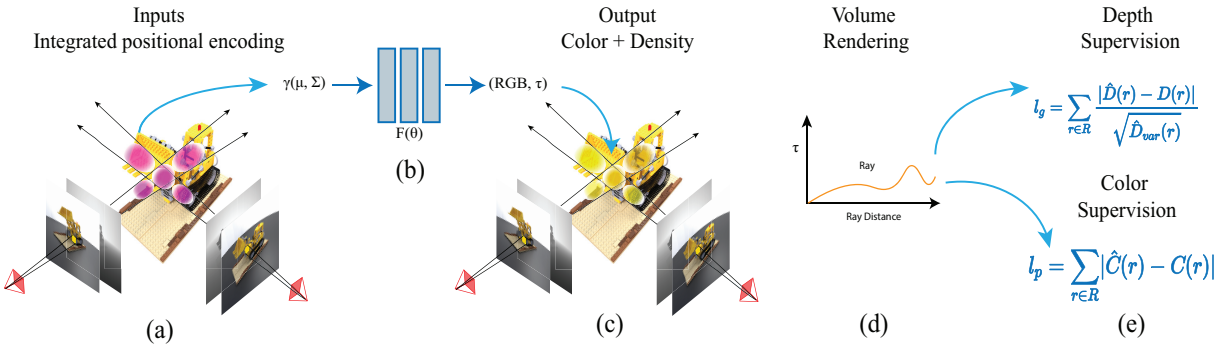


Figure 2: An overview of the proposed Mip-NeRF RGB-D. **(a)** The input to the network is the integrated positional encoding of a conical frustum segment. **(b)** The network outputs volume density and color. **(c)** The color and depth of a ray is generated using the classic volume rendering method. **(d)** The network is optimized using a color and depth loss.

2.3 *Neural volume rendering*

Lombardi et al. [Lombardi et al., 2019] initially introduced volume rendering for novel view synthesis using a CNN-based encoder and an MLP-based decoder to produce density and color for each point in space. The well-known Neural Radiance Fields approach was introduced in [Mildenhall et al., 2020] and demonstrated compelling results with a simple method that takes 3D points and the associated view direction as input to an MLP and outputs density and color. Some of the drawbacks of NeRF are its long training time, its long rendering time, the need to train a separate model for each scene, and it only works on static scenes. Various investigations have been conducted since the original NeRF to address these problems. [Liu et al., 2020, Neff et al., 2021, Garbin et al., 2021, Reiser et al., 2021, Yu et al., 2021] addressed the slow inference time of NeRF by using a tiny MLP and a better sampling strategy. [Deng et al., 2021] used sparse depth supervision during training to improve the training time of the NeRF model. [Pumarola et al., 2020, Gafni et al., 2020, Noguchi et al., 2021] address the problem of static scenes. [Yu et al., 2020, Schwarz et al., 2020, Tancik et al., 2020, Chan et al., 2020] have generalized NeRF models using fully convolutional image features, a generator discriminator, and meta-learning. Saito et al.[Barron et al., 2021] focused on NeRF aliasing and sampling problems. They proposed an integrated positional encoding, which uses a conical frustum defined by the mean and covariance of the rays, and the neural radiance field is integrated over the region represented by 3D Gaussian encoding.

2.4 *Neural radiance field with depth*

To solve the problem of incorrect geometry prediction when a limited number of input views are given, [Deng et al., 2021] proposed using depth as alternate supervision. Ds-NeRF uses a sparse 3D point cloud and then reprojects the errors between the detected 2D keypoints and projected 3D points, generated by commonly used structure-from-motion (SfM) algorithms which are error-prone. They optimize the model over a combined color and depth loss function. Similarly, NerfingMVS [Wei et al., 2021] uses a monocular depth network to generate depth prior from SfM reconstruction of the scene. The adapted depth priors are used to guide the sampling process of points along the ray. Unlike DS-NeRF, it generates a dense depth prior from sparse SfM points using a pretrained depth network. Azinovic et al.[Azinović et al., 2021] also demonstrated the incorporation of depth with NeRF to produce a better and more detailed reconstruction than simply using color or depth alone. Unlike others, it uses a truncated signed distance function(TSDF) instead of volume density to represent the underlying geometry. It still uses two networks that significantly affect training and prediction time. On the other hand, iMap [Sucar et al., 2021] shows that NeRF can be used to represent scenes in a real-time SLAM system. It jointly optimizes the 3D map and camera pose using keyframes. iMap uses a smaller MLP (4 layers) than NeRF and does not consider the viewing direction to model lighting effects. DONeRF [Neff et al., 2021] proposed a compact dual network design to reduce evaluation cost, leading to a faster prediction time. The coarse NeRF network is replaced by a *depth oracle network* based on a classification network. To reduce the number of samples along the rays, they suggested nonlinear transformation and a local sampling strategy, which helped them to achieve a similar result to NeRF with a fraction of the samples, but the method is limited to forward facing scenes.

3 METHODS

Now the proposed Mip-NeRF RGB-D will be presented. First, the implicit scene representation used by

NeRF based methods will be over-viewed, followed by the explanation of the rendering process. After that, an efficient network architecture will be proposed and a joint optimization method using RGB-D data will be presented. Finally, the local sampling strategy used to reduce the number of samples along the rays and reduce training time will be described.

3.1 Implicit scene representation

The proposed system is based on the Mip-NeRF [Barron et al., 2021] method which is an extension of NeRF for handling anti-aliasing. Vanilla NeRF based methods use a set of images and corresponding poses to train a MLP network that represents the scene by outputting the emitted radiance and volume density of 3D locations. Given 5D coordinates(3D location + viewing direction) as input, the network F_Θ learns an implicit function that estimates color $C = (r,g,b)$ and volume density τ as:

$$F_\Theta : (x,y,z,\theta,\phi) \rightarrow (C,\tau). \tag{1}$$

First, rays $r(t) = \mathbf{o} + t\mathbf{d}$ passing through each pixel of the image are generated, where the ray origin $\mathbf{o}$ is the camera center and $\mathbf{d}$ is the ray direction. Then N sample points are placed along the ray stratified manner between predefined near and far bounds. The color of each pixel is computed using a radiance and a volume density along the ray. In Mip-NeRF the rays are replaced with cones generated using the camera center and the pixel size. The cone is split into N intervals $\mathcal{T}_) = [t_i, t_i + 1)$ and for each interval the integrated positional encoding of the mean and the covariance (μ,Σ) of the corresponding conical frustum is computed. Integrated positional encoding encodes the Gaussian approximation of the conical frustum as follows:

$$\gamma(\mu,\Sigma) = \left\{ \begin{bmatrix} \sin(2^l\mu) exp(-2^{l-1} diag(\Sigma)) \\ \cos(2^l\mu) exp(-2^{l-1} diag(\Sigma)) \end{bmatrix} \right\}_{l=0}^{L-1}, \tag{2}$$

where Σ is the covariance of the Gaussian approximation:

$$\Sigma = \sigma_t^2 (\mathbf{d}\cdot\mathbf{d}^{\mathrm{T}}) + \sigma_r^2 \left(I - \frac{\mathbf{d}\cdot\mathbf{d}^{\mathrm{T}}}{||\mathbf{d}||_2^2} \right). \tag{3}$$

The variance along the ray is denoted by σ_t^2 and the variance perpendicular to the ray is σ_r^2. Mip-NeRF uses this integrated positional encoding instead of the frequency positional encoding as input to the neural network. One of the key difference between Mip-NeRF and the proposed method is the local sampling strategies, which will be discussed in the subsequent part of this article.

3.2 Volume rendering

Similarly to NeRF, a volume rendering formula was used to calculate the color and the depth of pixels from radiance and volume density of the conical frustum. The volume density $\tau(P)$ at location $P = (x,y,z)$ can be interpreted as the differential probability of ray termination. The expected color $C(r)$ of a camera ray $r(t) = \mathbf{o} + t\mathbf{d}$ with near and far bounds t_n and t_f is:

$$C(r) = \int_{t_n}^{t_f} T(t)\tau(r(t))c(r(t),\mathbf{d})dt, \tag{4}$$

where

$$T(t) = exp(-\int_{t_n}^{t} \tau(r(s))ds). \tag{5}$$

The function $T(t)$ denotes the accumulated transmittance along the ray from t_n to t, i.e., the probability that the ray travels from t_n to t without hitting any other particle. In the stratified sampling approach $[t_n, t_f]$ is partitioned into N evenly-spaced bins and then one sample is drawn uniformly at random from within each bin. The samples are used to estimate predicted color $\widehat{C}(r)$ as:

$$\widehat{C}(r) = \sum_{i=1}^{N} T_i (1 - exp(-\tau_i\delta_i))c_i, \tag{6}$$

where

$$T_i = exp(-\sum_{j=1}^{i-1} \tau_j\delta_j). \tag{7}$$

Here $\delta_j = t_{j+1} - t_j$ is the distance between adjacent samples. Similarly [Wei et al., 2021], the depth can be represented with volume density using:

$$\widehat{D}(r) = \sum_{i=1}^{N} T_i (1 - exp(-\tau_i\delta_i))t_i, \tag{8}$$

where T_i is the accumulated transmittance. To optimize the network, NeRF uses a squared error between the rendered and true pixel colors.

3.3 Optimization

The network parameters θ are optimized using a set of RGB-D frames, each of which has a color, depth, and camera pose information. The proposed method minimizes the geometric and photometric loss together on a set of frames as the rendering functions are completely differentiable. The photometric loss l_p is the absolute difference (L1-norm)[Sucar et al., 2021] between the predicted color and the ground truth color of the ray. The photometric loss over a set of rays is defined as:

$$l_p = \sum_{r\in R} |\hat{C}(r) - C(r)|. \tag{9}$$

The geometric loss is the absolute difference between predicted and true depths, normalized by the depth variance [Sucar et al., 2021] to discourage weights with high uncertainty:

$$l_g = \sum_{r\in R} \frac{|\hat{D}(r) - D(r)|}{\sqrt{\hat{D}_{var}(r)}}, \tag{10}$$

where $\hat{D}_{var}(r) = \sum_{i=1}^{N} T_i(1 - exp(-\tau_i\delta_i))(\hat{D}(r) - t_i)^2$ depth variance of the image. The neural network can be optimized by combining photometric, and geometric losses together using empirically chosen scale factors λ_p:

$$min_\theta(l_g + \lambda_p l_p). \quad (11)$$

3.4 Network architecture

The network architecture is similar to the original NeRF with some modifications. The proposed method uses only one network with 4 hidden layers of feature size 256. The skip connection is used in layer 3. The viewing direction is concatenated to the fourth layer before the color and volume density are output. The integrated positional encoding was applied to the conical frustum and a positional encoding of frequency 4 is applied to the viewing directions as was done in Mip-NeRF. By decreasing the network size, faster training and prediction time were achieved without significantly compromising the novel view quality.

3.5 Local sampling

NeRF based methods estimate pixel color by placing samples on viewing rays traced through the pixels. The final color of the pixels is calculated by the alpha composition [Max, 1995] of the volume density and the radiances of the samples along the ray. Samples relevant to the volume produce higher volume density, so samples close to the surface are more relevant to the pixel's final color. NeRF uses 256 samples in a stratified manner and 2 networks to ensure that samples are placed on relevant parts of the ray. To compute a pixel color, each sample on that ray needs a full network evaluation, so the training time increases exponentially with the number of samples on the ray. Although Mip-NeRF uses a conical frustum instead of viewing rays, it still requires 2 network passes and a large number of bins to create integrated positional encoding. In reality, the majority of the scene volume is empty space (for 360 scenes), and the samples placed on the empty space have less contribution to the final color. Therefore, given the depth information of an image, it is possible to place fewer samples and to place them directly on the relevant parts of the ray, while achieving similar quality results. Finally, in this case, it is also possible to eliminate the coarse network with local sampling, which NeRF uses to find important sampling locations along the ray. Various depth-guided sampling strategies have been considered. Figure:3 shows the comparison between the proposed local sampling and the baseline approaches.

3.5.1 Stratified sampling

The so-called stratified sampling strategy is very similar to the original Mip-NeRF sampling, but the conical

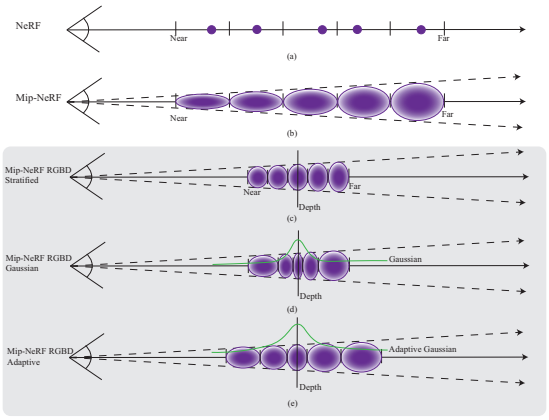


Figure 3: Visualization of the proposed sampling strategies(c, d, e) compared to NeRF(a) and Mip-NeRF(b). Black arrows represent ray direction and purple ellipsoids represent a Gaussian approximation of the conical frustum.

frustums are generated only close to the surface based on depth information(Figure:3(c)). Here the space between the near and far bounds $[t_n, t_f]$ is divided into N evenly spaced bins and a sample is drawn uniformly at random from each bin where $t_n = D - \alpha_n$ and $t_f = D + \alpha_f$, α_n and α_f are empirically chosen based on the depth uncertainty. The samples are then used as the bounds of the conical frustum. This allows the network to avoid empty space and eventually decrease the number of bins needed for each ray.

3.5.2 Gaussian sampling

In this strategy, instead of placing the bins equidistantly around the surface, the limits of the conical segments are selected from a normal distribution where the mean is the depth and the standard deviation ς is empirically chosen based on the depth uncertainty. In this way, it ensured that the conical frustums are smaller (toward the ray direction) on the relevant part of the ray(close to the true depth as in Figure:3(d)) to emphasize the high-frequency details on the surface. This allows the network to handle the generalized uncertainty present in the depth estimate.

3.5.3 Adaptive sampling

The adaptive sampling strategy uses a normal distribution with a varying standard deviation $\varsigma(r)$ based on the number of epochs and the depth of the ray. Therefore, the normal distribution (the mean is the depth measurement) is used to define the limits of the conical frustums(Figure:3(e)). $\varsigma(r)$ varies during training according to the number of epochs in a coarse to fine manner to improve the fine photometric details. Additionally, this sampling strategy takes into account the depth uncertainty, which increases with distance. The standard deviation of each ray is calculated as follows:

$$\varsigma(r) = \frac{D(r)}{4}(\exp^{-\lambda_r i} + \lambda_m), \quad (12)$$

where i is the epoch number, λ_r is the rate of decrease, λ_m is minimum standard deviation, and $D(r)$ is the true depth of the ray. λ_r and λ_m are empirically chosen based on dataset and depth uncertainty.

4 EXPERIMENTAL RESULTS

In this section, the proposed methods are evaluated on various datasets and compared with other state-of-the-art NeRF based methods.

4.1 Experimental Setup

4.1.1 Datasets

Simulated datasets were used for all experiments. Each data set contained RGB images, depth maps, and their corresponding camera poses. All poses in the datasets belong to an upper hemisphere, where the object is placed in the center. Four different scenes were considered for the experiments: Lego[1], Cube, Human [2], and Drums[3]. The input images have 800×800 resolution, and the depth measurements are in meters. Each of the datasets has three versions, in which the number of training images is 8, 30, and 100.

4.1.2 Implementation Details

The proposed method is implemented using a combination of PyTorch and CUDA. The ADAM optimizer with a learning rate of 5×10^{-4} and an exponential decay of the learning rate of 5×10^{-1} in every five epoches has been used. A batch size of 2048 on 2 Nvidia Rtx 3090 GPUs was used for all experiments. 16 frequency bands were used for integrated positional encoding of the conical frustum and 4 frequency bands to encode viewing directions with positional encoding.

For all experiments, the following parameters are used as default: the photometric scale factor for the loss function is $\lambda_p = 100$, standard deviation for Gaussian sampling is 0.3, $\lambda_r = 0.09$ and $\lambda_m = 0.1$ for adaptive sampling.

4.1.3 Metrics

Four metrics are used to evaluate the predicted RGB image quality and depth map: *Peak Signal-to-Noise Ratio (PSNR in dB)*: to compare the quality of the RGB reconstruction, the higher is better; *Absolute Relative distance* (Abs Rel in m): to compare the quality of the generated depth map, the lower is better; *Structural Similarity Index* (SSIM in %) [Wang et al., 2004]: quantifies the degradation of image quality in the reconstructed image, the higher is better; *Learned Perceptual Image Patch Similarity* (LPIPS) [Zhang et al., 2018]: the distance between the patches of the image, the lower means that the patches are more similar.

[1] https://www.blendswap.com/blend/11490
[2] https://renderpeople.com/free-3d-people/
[3] https://www.blendswap.com/blend/13383

4.2 Comparison

First, local sampling strategies are compared in Section 4.2.1. Then, the effect of a different number of samples on the proposed model is discussed. After that, the proposed method is applied in different scenes. Finally, in Section 4.2.5, the proposed method is compared with other NeRF-based methods that use depth supervision.

4.2.1 Comparison between different local sampling strategies

Instead of placing samples over the entire ray, local sampling places samples only on the relevant regions of the ray using depth information. In this section, proposed local sampling strategies are compared. Table 1 shows the quantitative performance of the three sampling strategies mentioned.

	Metrics			
Strategy	PSNR ↑	SSIM ↑	Abs Rel↓	LPIPS ↓
Equidistant	19.86	0.87	0.02	0.0023
Gaussian	20.75	0.88	0.04	0.0022
Adaptive	21.09	0.89	0.04	0.0024
NeRF	19.21	0.88	0.34	0.0036

Table 1: Comparison of three different sampling strategies. The Lego spherical dataset containing 8 training images has been used for all experiments. All experiments used 16 sampling points per ray. Best values are highlighted by green, significant wrose values by red, and darker shades represent best values.

The results show that adaptive sampling performs best among the proposed sampling strategies. All local sampling strategies improve the underlying geometry compared to NeRF.

4.2.2 Effect of ray sample size

NeRF based methods use a large number of ray samples to estimate volume density and produce fine output details. The number of samples is arguably the most important parameter for any NeRF model because it is directly related to the training time, the prediction time, and the quality of the novel view. The following experiments in Table 2 demonstrate that the proposed method performs significantly well even when the number of samples is low. The 64 samples provide the best compromise between novel view quality and training time.

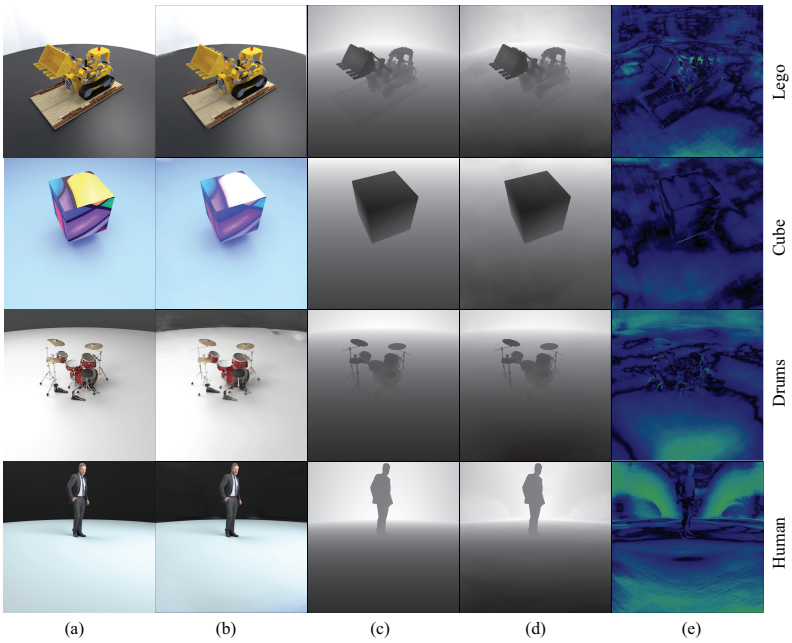


Figure 4: Qualitative comparison on blender scenes: Visual comparison between generated RGB ground truth images and the true depth maps generated by the proposed method, where (a) Ground truth RGB images; (b) Predicted RGB images; (c) True depth maps; (d) Predicted depth maps; (e) Absolute error between predicted and true depth maps.

	Number of samples		
Metrics	16	64	128
PSNR ↑	19.4	21.18	22.13
SSIM ↑	0.86	0.89	0.9
Abs Rel ↓	0.05	0.04	0.05
LPIPS ↓	0.0025	0.0023	0.0018
Training Time ↓	38m	44m	1.54h

Table 2: A comparison between training time and novel view quality based on the number of samples per ray.

4.2.3 *Different datasets*

The proposed method was evaluated with 4 different datasets with different characteristics to demonstrate its robustness in different types of scenes. The cube has a simple geometry but a complicated texture. Alternatively, the drums have a complicated and very detailed 3D structure. The Lego scene is a good mix of photometric and geometric details. The human scene mimics some real-world applications. Quantitative results are shown in Table 3 and qualitative results are shown in Figure 4.

	Metrics			
dataset	PSNR↑	SSIM↑	AbsRel↓	LPIPS↓
Lego	28.21	0.93	0.02	0.0013
Cube	22.78	0.95	0.01	0.0001
Human	37.7	0.97	0.02	0.00008
Drums	27.85	0.9	0.02	0.0012

Table 3: Performance of the proposed method on 4 different simulated datasets.

4.2.4 *Fewer input views*

To demonstrate that the proposed method can perform well even when the number of training images is limited, three different datasets with different numbers of training images have been considered for experiments. Table 4 shows a comparison between different datasets. Although increasing the number of inputs increases the quality of the novel view, the training time also increases significantly.

	Number of input views		
Metrics	8	30	100
PSNR ↑	21.18	25.25	28.21
SSIM ↑	0.89	0.92	0.93
Abs Rel ↓	0.04	0.04	0.02
LPIPS ↓	0.0023	0.0017	0.0013
Training Time ↓	44m	2.15h	6.45h

Table 4: More samples in the training set provides more supervision for the network to learn the scene representation. With an increasing number of input views, geometric and photometric metrics improve. Subsequently, training time increases significantly. A dataset with 100 images takes 11 times longer to train than dataset with 8 images.

4.2.5 *Scene representation*

In this section, the proposed method is compared with other state-of-the-art NeRF-based methods.

NeRF [Mildenhall et al., 2020]: The implementation of PyTorch Lighting of the NeRF by [Quei-An, 2020]

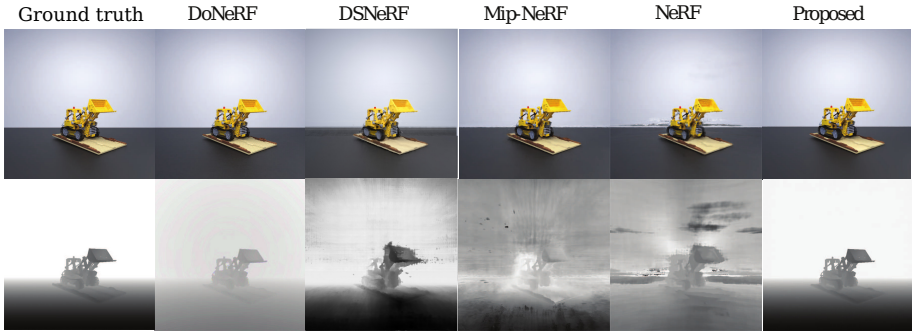


Figure 5: Qualitative comparison: Visual comparison results between the proposed method and other state-of-the-art methods.

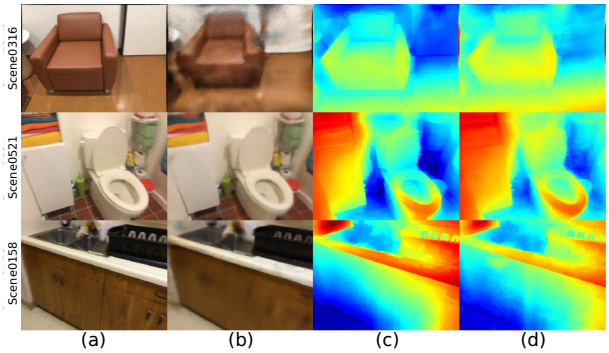


Figure 6: Qualitative result of the proposed method on 3 different real sensor datasets. (a) Ground truth RGB images; (b) Predicted RGB images; (c) Ground-truth depth maps; (d) Predicted depth maps.

	Metrics				
Method	PSNR ↑	SSIM ↑	Abs Rel↓	LPIPS ↓	Time ↓
DSNeRF	29.31	0.87	0.489	0.003	3:37h
DONeRF	39.23	0.98	0.008	0.00001	5:08h
NeRF	27.36	0.94	0.34	0.0015	3.40h
MipNeRF	30.66	0.95	0.334	0.006	1:27h
Proposed	32.72	0.95	0.001	0.0004	1.15h

Table 5: Quantitative comparison for novel view synthesis and depth estimation between the proposed method and state-of-the-art methods. The Lego dataset is used for all these experiments.

	Metrics			
dataset	PSNR↑	SSIM↑	AbsRel↓	LPIPS↓
scene0521	25.48	0.724	0.025	0.0004
scene0316	16.99	0.57	0.05	0.001
scene0158	24.93	0.74	0.02	0.0007

Table 6: Performance of the proposed method on 4 different real RGB-D datasets.

has been considered for the experiments. NeRF can be trained using simulated 360-degree Blender data or real data. For these particular experiments, simulated Blender scenes were used.

DSNeRF [Deng et al., 2021]: DSNeRF works only on the forward facing scenes where depth supervision data is generated using Colmap [Schonberger and Frahm, 2016]. The official implementation of DSNeRF was used for these experiments.

DONeRF [Neff et al., 2021]: DONeRF works only on forward-facing datasets where all poses belong to a view cell. This method works only with simulated data with a dense depth map. The official implementation of the DONeRF was used for the experiments.

Mip-NeRF [Barron et al., 2021]: The official Mip-NeRF implementation on JAX was converted to PyTorch for convenience of comparison.

All experiments were carried out on the same Lego scene dataset that contains 30 training images with resolution 800×800. The quantitative results in Table: 5 and the qualitative results in Figure: 5 show that DONeRF [Neff et al., 2021] can produce the best photometric quality, but is limited to forward-facing scenes, a longer training time, and oracle network-based depth prediction, where the proposed method uses only one smaller network (less space requirement), trains faster (4 times faster), and produces more accurate geometry.

Three different real-world datasets have been used from the NerfingMVS [Wei et al., 2021] for experimenting on real acquired depth images. Pre-processed depth maps were used instead of using raw depth maps because the raw depth maps contain areas without depth information (holes where sensors cannot estimate depth). NerfingMVS uses a monocular depth

prediction network to complete the missing depths. Alternatively, holes in the raw depth maps could be handled as in the classic RGB NeRF implementation, however, this randomly affects computational performance and comparisons. Table:6 shows the quantitative results of the proposed method on 3 different datasets. The adaptive sampling strategy with 16 samples was used for all of these experiments. The qualitative results of the experiments are shown in the Figure:6. The ground-truth depth shows that it is not very detailed and that some areas have wrong depth measurements, which results in some artifacts in the predicted image generated by the proposed method.

4.3 Analysis

Fewer views: The proposed method can learn a scene representation from fewer views, as depth supervision provides additional supervision and effective sampling. Depth supervsion allows the network to learn scene geometry and multi-view constancy from a very limited number of views.

Faster training: The results show a quantifiable speed improvement in training time with the proposed method compared to other state-of-the-art methods. Faster training was achieved using fewer samples, a smaller network architecture, and local sampling. The Mip-NeRF RGB-D method is $3-5\times$ faster compared to other similar NeRF-based methods.

Accurate depth estimation: The proposed method is capable of producing a more accurate geometry compared to other state-of-the-art methods. The network can learn accurate geometry from small number of inputs as few as 8 frames.

5 DISCUSSION

In this article, a new method was presented for representing 3D scenes from RGB-D data using recent neural radiance fields. Instead of learning the radiance field from RGB images, the proposed method uses RGB-D frames, which allows achieving better underlying geometry and faster training and prediction times. Additional depth supervision of dense depth maps is shown to have a significant improvement on the training time through local sampling. The proposed method trains $3-5\times$ faster and improves the novel view and depth quality. The experiments show significant improvements over the state-of-the-art methods, both quantitatively and qualitatively. Future perspective will be focused on extending this approach to dynamic scenes.

6 ACKNOWLEDGEMENTS

This project has received funding from the H2020 COFUND program BoostUrCareer under Marie Sklodowska-Curie grant agreement no. 847581. It also received funding from the EU H2020 MEMEX research project under grant agreement No. 870743.

7 REFERENCES

[Azinović et al., 2021] Azinović, D., Martin-Brualla, R., Goldman, D. B., Nießner, M., and Thies, J. (2021). Neural rgb-d surface reconstruction. *arXiv preprint arXiv:2104.04532*.

[Barron et al., 2021] Barron, J. T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., and Srinivasan, P. P. (2021). Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864.

[Chan et al., 2020] Chan, E., Monteiro, M., Kellnhofer, P., Wu, J., and Wetzstein, G. (2020). pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis. *https://arxiv.org/abs/2012.00926*.

[Chen and Zhang, 2019] Chen, Z. and Zhang, H. (2019). Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948.

[Deng et al., 2021] Deng, K., Liu, A., Zhu, J.-Y., and Ramanan, D. (2021). Depth-supervised nerf: Fewer views and faster training for free. *arXiv preprint arXiv:2107.02791*.

[Gafni et al., 2020] Gafni, G., Thies, J., Zollhöfer, M., and Nießner, M. (2020). Dynamic neural radiance fields for monocular 4D facial avatar reconstruction. *https://arxiv.org/abs/2012.03065*.

[Garbin et al., 2021] Garbin, S. J., Kowalski, M., Johnson, M., Shotton, J., and Valentin, J. (2021). Fastnerf: High-fidelity neural rendering at 200fps. *https://arxiv.org/abs/2103.10380*.

[Gortler et al., 1996] Gortler, S. J., Grzeszczuk, R., Szeliski, R., and Cohen, M. F. (1996). The lumigraph. *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54.

[Hedman et al., 2018] Hedman, P., Philip, J., Price, T., Frahm, J.-M., Drettakis, G., and Brostow, G. (2018). Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (TOG)*, 37(6):1–15.

[Hedman et al., 2016] Hedman, P., Ritschel, T., Drettakis, G., and Brostow, G. (2016). Scalable inside-out image-based rendering. *ACM Transactions on Graphics (TOG)*, 35(6):1–11.

[Heigl et al., 1999] Heigl, B., Koch, R., Pollefeys, M., Denzler, J., and Van Gool, L. (1999). Plenoptic modeling and rendering from image sequences taken by a hand-held camera. *Mustererkennung 1999, Springer*, pages 94–101.

[Liu et al., 2020] Liu, L., Gu, J., Lin, K. Z., Chua, T.-

S., and Theobalt, C. (2020). Neural sparse voxel fields. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33.

[Lombardi et al., 2019] Lombardi, S., Simon, T., Saragih, J., Schwartz, G., Lehrmann, A., and Sheikh, Y. (2019). Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*.

[Max, 1995] Max, N. (1995). Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108.

[Mescheder et al., 2019] Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. (2019). Occupancy networks: Learning 3d reconstruction in function space. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470.

[Mildenhall et al., 2020] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer.

[Neff et al., 2021] Neff, T., Stadlbauer, P., Parger, M., Kurz, A., Mueller, J. H., Chaitanya, C. R. A., Kaplanyan, A. S., and Steinberger, M. (2021). DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 40(4).

[Noguchi et al., 2021] Noguchi, A., Sun, X., Lin, S., and Harada, T. (2021). Neural articulated radiance field. *arXiv preprint arXiv:2104.03110*.

[Park et al., 2019] Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. (2019). Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174.

[Pumarola et al., 2020] Pumarola, A., Corona, E., Pons-Moll, G., and Moreno-Noguer, F. (2020). D-NeRF: Neural radiance fields for dynamic scenes. *https://arxiv.org/abs/2011.13961*.

[Quei-An, 2020] Quei-An, C. (2020). Nerf_pl: a pytorch-lightning implementation of nerf. `https://github.com/kwea123/nerf_pl/`.

[Reiser et al., 2021] Reiser, C., Peng, S., Liao, Y., and Geiger, A. (2021). Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps.

[Saito et al., 2019] Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., and Li, H. (2019). Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2304–2314.

[Schonberger and Frahm, 2016] Schonberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113.

[Schwarz et al., 2020] Schwarz, K., Liao, Y., Niemeyer, M., and Geiger, A. (2020). Graf: Generative radiance fields for 3D-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33.

[Sitzmann et al., 2019] Sitzmann, V., Zollhöfer, M., and Wetzstein, G. (2019). Scene representation networks: Continuous 3d-structure-aware neural scene representations. *arXiv preprint arXiv:1906.01618*.

[Sucar et al., 2021] Sucar, E., Liu, S., Ortiz, J., and Davison, A. J. (2021). imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238.

[Tancik et al., 2020] Tancik, M., Mildenhall, B., Wang, T., Schmidt, D., Srinivasan, P., Barron, J. T., and Ng, R. (2020). Learned initializations for optimizing coordinate-based neural representations. *https://arxiv.org/abs/2012.02189*.

[Thies et al., 2020] Thies, J., Zollhöfer, M., Theobalt, C., Stamminger, M., and Nießner, M. (2020). Image-guided neural object rendering. *8th International Conference on Learning Representations*.

[Wang et al., 2004] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612.

[Wei et al., 2021] Wei, Y., Liu, S., Rao, Y., Zhao, W., Lu, J., and Zhou, J. (2021). Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5610–5619.

[Yu et al., 2021] Yu, A., Li, R., Tancik, M., Li, H., Ng, R., and Kanazawa, A. (2021). Plenoctrees for real-time rendering of neural radiance fields. *arXiv preprint arXiv:2103.14024*.

[Yu et al., 2020] Yu, A., Ye, V., Tancik, M., and Kanazawa, A. (2020). pixelNeRF: Neural radiance fields from one or few images. *https://arxiv.org/abs/2012.02190*.

[Zhang et al., 2018] Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

3D Point Set Registration based on Hierarchical Descriptors

Somnath Dutta
Technische Universität Dresden, Germany
somnath.dutta@tu-dresden.de

Benjamin Russig
Technische Universität Dresden, Germany
benjamin.russig@tu-dresden.de

Stefan Gumhold
Technische Universität Dresden, Germany
stefan.gumhold@tu-dresden.de

ABSTRACT

Registering partial point clouds is crucial in numerous applications in the field of robotics, vision, and graphics. For arbitrary configurations, the registration problem requires an initial global alignment, which is computationally expensive and often still requires refinement. In this paper, we propose a pair-wise global registration method that combines the fast convergence made possible by global hierarchical surface descriptors with the arbitrarily fine sampling enabled by continuous surface representations. Registration is performed by matching descriptors of increasing resolution – which the continuous surfaces allow us to choose arbitrarily high – while restricting the search space according to the hierarchy. We evaluated our method on a large set of pair-wise registration problems, demonstrating very competitive registration accuracy that often makes subsequent refinement with a local method unnecessary.

Keywords

3D shape registration, surface descriptors, similarity measure.

1 INTRODUCTION

Acquiring 3D geometry of physical objects and environments has become an integral part of numerous disciplines, ranging from applications in mechanical engineering and computer vision to asset generation for movies and games. Rapid development of 3D sensing technology over the last decades led to affordable high precision devices being widely available. Most techniques follow a sequential scanning paradigm where an object or scene is scanned from different view points. The result of this acquisition process is a set of scans, each in a local coordinate system defined by the pose of the scanning device, with different degrees of overlap. These scans have to be aligned with respect to a unique reference to obtain a final reconstructed model of an object. The accuracy of this registration step is of significant importance to the quality of the final model but is often hampered due to low-overlap between scans and inherent noise. Along with that, the registration errors may lead to artificial creases and reconstruction artifacts (oscillations or even holes).

The goal of pairwise registration is to find a transformation that aligns a source scan as close as possible to the surface represented by a target scan. The possible approach that could assist in precise registration accuracy is combination of coarse alignment followed by an iterative local refinement that results in higher convergence. The local refinement is performed effectively using variants of Iterative closest point (ICP) [Bes92] which highly depends on a good initial guess.

Coarse registration approaches usually depend on a descriptor [MPD06,Zah12,Tom10,Pet15] associated with a set of feature points encoding useful geometric information from the data based on a local neighborhood. Correspondences between descriptors are established based on similarity, resulting in a rigid transformation that best aligns the set of feature points and using some robust estimator.

The object surface is generally not sampled at the same locations between partial scans (an effect which we refer to as sampling discrepancy), and every sample is subject to measurement noise. Methods that directly rely on descriptor computation are inherently limited by these measurement issues. In addition to these external factors, methods relying on localized feature descriptors often suffer from a certain percentage of false matches, typically caused by insufficient discriminating capabilities of the descriptor and/or an underperforming similarity measure. These factors, to varying degrees, prevent them from achieving higher levels of accuracy, and the resulting rigid transformation negatively affects the convergence of local iterative algorithms employed for refinement. We propose a method that mitigates these drawbacks. Our contributions are as follows:

- novel adaption of an existing hierarchical surface descriptor to continuous surface representations
- based on that, a new pair-wise global registration pipeline for partial scans, featuring:

This work has received funding from the two DFG grants 389792660 (TRR 248, CPEC) and 390696704 (EXC 2050/1, CeTI) as well as from BMBF grant 01/S18026A-F (ScaDS.AI Dresden/Leipzig)

- high precision that compares favorably to state-of-the-art global methods
- competitive processing times
- either eliminates the need for refinement with a local method completely or gives close-to-optimal initial guesses

Additionally, the paper contributes a comprehensive evaluation of the proposed approach and compares it to the state-of-the-art global registration method by [Zho16], demonstrating measurable improvements in terms of accuracy and robustness.

2 RELATED WORK

Registration is a widely researched problem with vast applicability in various domains like 3D scanning, shape analysis, or motion capturing. For an overview of existing registration algorithms, we refer the reader to the respective surveys [Rus01], Salvi et al. [Sal07], Tam et al. [Tam13], Bellekens et al. [Bel14], Huang et al. [Hua21]. According to the aforementioned surveys, registration approaches can be classified with respect to the following criteria:

Pairwise / Multi-Way. Pairwise registration methods [Bel14] aim at registering two partially overlapping scans. Overall registration of a data set can be achieved by successively adding scans and registering them to their predecessors or to all preceding scans. On the other hand, multi-way registration [Goj09] optimizes all transforms at the same time, usually by iterating pairwise registrations or by modeling the problem with a single registration objective.

Rigid / Non-Rigid. Rigid registration methods [Bel14] only allow rigid body transforms for the individual scans, whereas non-rigid methods allow arbitrary ones. However, non-rigid approaches [Hua22] usually regularize the objective in order to avoid degenerate solutions (e.g. by demanding the transform to be as rigid as possible).

Local / Global. Local methods [Bes92, Hua11] use an initial transform estimation (e.g. provided by the user) and refine this transformation to optimize an objective. Global methods [Zho16, Pet15] do not require any input other than the geometry and find the registration transforms with arbitrary initial alignment of scans. Such global methods usually produce coarse registrations that can be refined with local methods. Furthermore these methods are not directly based on spatial proximity (but on, e.g., geometric feature descriptors).

Learning-based Registration. In recent years, data-driven approaches are emerging for registering point clouds primarily owing to exemplary results in both 2D and 3D applications. Researchers have used neural networks (NNs) in different ways, either to extract feature from individual point cloud followed by correspondence search or even focused on end-to-end pipeline for transformation estimation. Feature learning methods [Zen17, Hao18, Goj19] use the deep neural network to learn a robust feature correspondence search. Then, the transformation matrix is obtained by one step estimation (SVD, RANSAC) without iteration. Those NNs could provide robust and accurate correspondence searching but are fairly limited by availability of large training data and lesser generalization capabilities, resulting in a large-scale failure for unknown scenes with different distribution compared to the training data. End-to-end learning-based method [Yan19, Wan19] solves the problem of registration with complete end-to-end network, i.e, along with the correspondence search, transformation estimation is also embedded into the framework and differs from feature learning method whose focus in entirely on point feature learning. DeepGMR [Yua20] relies on a neural network to initially learn pose-invariant point-to-distribution parameter correspondences. Then, these correspondences are fed into the GMM optimization module to estimate the transformation matrix.

Descriptors

Many 3D descriptors based on point clouds have been introduced and importantly applied to the task of point cloud registration. A comprehensive reviews for point cloud descriptors in terms of computational efficiency and accuracy are detailed in [Ale12, Guo16, Han18, Ran20, Han18]. Based on the process of descriptor extraction, the descriptors can be approximately classified as local, global, and hybrid. Local descriptors, for every point, embeds spatial distribution or geometric attributes extracted in the neighborhood and is produced by a histogram representing the descriptors for each point. It is less discriminating exactly, but suitable for cluttered scenes. Spin Image [Joh99] is a popular descriptor often used in applications of shape matching, object recognition. It creates a cylindrical support around a keypoint, divides it into radial and vertical volumes, and then counts the number of points that falls within each volume. On the other hand, a global descriptor, for the entire point cloud, is produced with a single context vector by integrating a set of features or incorporating spatial distribution of the complete data. Global-based features are generally used in 3D object recognition and object categorization, additionally some global descriptors can be used for pose estimation of the objects as they also contain local descriptors information, such as Viewpoint Feature Histogram (VFH) and Local-to-Global Signature (LGS) descriptor. Moreover, global-based methods demands less computation compared with the local features and better to describe the whole point cloud but are majorly affected by occlusions and clutter [Had14]. Hybrid-based descriptors incorporate local and global descriptors together with most of the advantages from them.

Sampling Discrepancy

As briefly mentioned when motivating our work in section 1, sampling discrepancy is a major factor limiting the accuracy of point cloud registration, as it means that in general, no exact correspondences can be found. Thus, many strategies have been proposed to reduce its impact. Chen and Medioni [Chen91] project points from the source shape onto their closest position in the target depth image with an iterative algorithm now known as *normal shooting* [Rus01]. The resulting point will yield an interpolated closest point on the target.

The Adaptive moving least squares (AMLS) surface based approach [Hua11] for ICP registration emphasize on the efficiency of the technique to overcome issues of sampling discrepancy. The essential concept of the AMLS method is to reconstruct a smooth and accurate representation of a surface for ICP based registration by adaptively selecting the width of a Gaussian kernel based on the principal curvature of the MLS surface through local integral invariant analysis [Yan06]. Instead of trying to minimize the distance between corresponding entities, [Mit04, Pot06] estimate the distance field of the underlying surface with local quadratic approximates and directly optimize the distance of the scan and the target surface. Kubacki et al. [Kub12] integrated multiple depth images into a model represented by implicit moving least squares (IMLS) on a grid based on signed distance function updates. Similar ideas appeared in [Par03, Mun07]. However, any such method inherently suffers from discretization artifacts at reasonable resolutions.

3 METHODOLOGY

The proposed method relies on the core concept of descriptor matching in order to estimate the rigid transformation. Our approach is based on the *Circon* descriptor proposed by Ferrero et al. [Car12] that represents an ordered set of radial contours around a point of interest within a point cloud. The descriptor associated with a particular point-of-interest is used to express the point cloud in a local frame. The environment around a point-of-interest is divided into sectors each representing an angle ρ_θ, and are further divided radially into cells with length ρ_r.

The points are mapped into cells of the descriptor in a way that the cell-value represents the height (z-value) with respect to the local reference frame. Figure 1 depicts a descriptor associated with a point-of-interest (cf. section 3.1) encoding the structural information from a point cloud.

We built upon the registration method Ferrero et al. [Car12] tailored for this descriptor. While the core hierarchical approach is the same, the way they use the point cloud directly to build the descriptors ties the

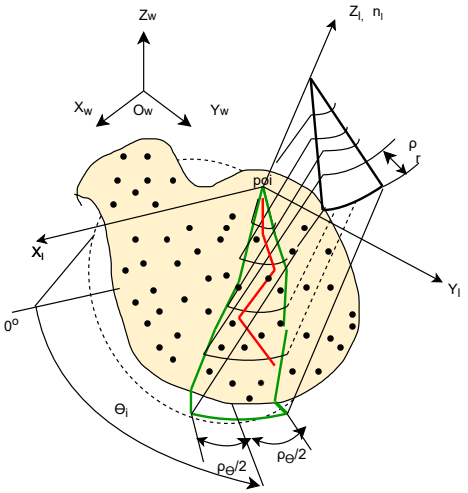


Figure 1: Descriptor spawned by a point-of-interest defining a local reference frame with cell division into sectors represented by green and red shows contour formed by maximum z within each cell.

achievable accuracy to the point cloud resolution. We propose to utilize a continuous surface representation instead, enabling us to construct descriptors of arbitrarily high resolution.

The exact choice of surface representation poses a trade-off of surface expressiveness vs. computational complexity. We opted for NURBS surfaces, as their desirable properties include the ability to exactly represent relevant algebraic shapes (like ellipsoids), and many manufactured objects are likely to have been CAD-designed using NURBS originally. They also render computation of the Circon descriptor non-trivial, which required additional measures to tackle the added computational cost. Our proposed algorithm can be summarized as follows:

AL.1 (pre-processing) Fit NURBS surfaces to the source and target point clouds.[1] We first perform Euclidean clustering [Rad09] (see **Appendix A** for details) on both to account for highly fragmented scans, and each cluster is fitted with its own surface. We refer to the union of all surfaces fitted to a point cloud as *the* surface of the point cloud.

AL.2 (pre-processing) Sample each fitted surface at a resolution of 256×256 in parameter space to get a collection of (u_0, v_0) footpoints needed later for descriptor construction (see section 3.2).

AL.3 (pre-processing) Point-of-interest (*poi*) selection for both surfaces (section 3.1).

[1] We fit trimmed cubic surfaces using the iterative tangent distance-based refinement strategy proposed by Mörwald et al. [Moe13] as implemented in PCL [Rus11] using default parameters.

AL.4 (pre-processing) Initial sorting of *poi* (section 3.4) pairs at a coarse resolution according to their similarity score (section 3.3), so step AL.5 can start with the most promising pairs.

AL.5 Descriptor computation across resolutions (section 3.4) for a pair of *poi* from source and target surfaces.

AL.6 Estimate transformation from point-pairs with highest similarity score at maximum resolution (user-defined) and evaluate stopping criterion.

AL.7 Repeat from AL.5 until the stopping is criterion satisfied.

We elaborate on the key details of the registration methods in the following sections.

3.1 Selecting Points-of-Interest

Points-of-interest (*poi*) for the source and target shapes are essential for the proposed approach. They define where descriptors are initially constructed and should be chosen in a way such that the resulting descriptors cover every location on the shape at least once. The original registration algorithm by Ferrero et al. [Car12] adapts a strategy to select non-edge points by performing edge detection and thresholding based on the Laplacian of the normal vectors, essentially restricting their selection to relatively flat areas. We did not find this to provide measurable benefits, so we chose our *poi* based on a random sampling of locations on the respective surfaces to obtain the final set of points-of-interest, represented by: $s_{poi} = \{s^1_{poi}, s^2_{poi}, s^3_{poi} ..., s^m_{poi}\} \subset S$, $t_{poi} = \{t^1_{poi}, t^2_{poi}, t^3_{poi} ..., t^n_{poi}\} \subset T$, $m < N_1, n < N_2$, where N_1, N_2 are total no. of samples from the continuous surface representation of the source and target scan.

3.2 Descriptor Construction

Descriptors are natural choice for estimating transformation in a global registration framework as detailed in section 2. It encodes local or global structural information of a 3D shape.

The local reference frame of a Circon descriptor is defined by the normal at the point-of-interest z_l (queried from the parametric surface representing the shape), some perpendicular vectors y_l and $x_l = z_l \times y_l$, as well as the keypoint itself as origin. We refer to the plane with normal z_l which the kepoint lies on as the *reference plane* of the descriptor.

Circon descriptors can form a hierarchy, as each cell in turn implies another keypoint that spawns a descriptor. By doubling the resolution at each level, a tree of course-to-fine representations is formed. The points-of-interest we determined in section 3.1 form roots of such a hierarchy.

So, at a given resolution, we can iterate over each valid cell (i, j) of the descriptor and further build descriptors associated with the 3D point of the cell. In this regard, the descriptors are identical to [Car12], but we largely modified the construction process to work with parametric surfaces as follows: To obtain a cell value, we cast a ray from the cell along the z-direction of the local reference frame and compute the intersection of the ray with the surface transformed to this local frame. Since we use NURBS surfaces, the intersection has to be determined numerically as no exact analytical solution is known: We start at certain point $S(u_0, v_0)$ on the surface and optimize its position (u, v) so that it lies on the ray originating from the cell center $\vec{o}$ along its direction $\hat{d}$ defined by the local frame z-axis. This yields the following optimization problem:

$$\underset{u,v}{argmin} \left\| (\vec{o} + < S(u,v) - \vec{o}, \hat{d} > \hat{d}) - S(u,v)) \right\|_2^2 \quad (1)$$

We use the simple Nelder-Mead algorithm [Nel65] to solve this, since the problem is low-dimensional and the evaluation of the objective function is cheap. This strategy requires an initial guess though, which we obtain in the following way:

During pre-processing, we perform a regular sampling of the NURBS surfaces in parameter space, giving us a database of footpoints on the surface and their associated parameters (u_0, v_0). When transforming a surface into the local frame of the current descriptor, we can transform these footpoints along with it. Using a simple regular grid on the descriptor reference plane, descriptor cells can collect those footpoints that project close to them. We then select the most promising initial guess by applying a multi-criterion sort according to *(a)* footpoint height z above the descriptor reference plane and *(b)* closeness to the ray.

We accomplish *(a)* by binning the footpoints in the cell according to their height z, and *(b)* results from sorting them inside each bin. For selecting the initial guess, we begin with the highest z-bin and choose the first (and thus, closest) sample from it. If no intersection could be found, we continue with the next sample in the bin, or the next lower bin if the current bin does not contain more samples, and so on.

The NURBS surface is then evaluated at the optimized parameter (u', v') to determine corresponding 3D point along with its normal. The cell value $c_{i,j}$ is obtained from the z-coordinate of the point quantized with respect to a height resolution ρ_z as in equation 2, i.e.

$$c_{i,j} = \lceil \frac{z}{\rho_z} \rceil \quad (2)$$

The x, y coordinate of each cell is given by

$$\begin{aligned} x &= j * \rho_r * \cos(-(i-1) * \rho_\theta)) \\ y &= j * \rho_r * \sin(-(i-1) * \rho_\theta)) \end{aligned} \quad (3)$$

As the descriptor embeds an environment of the *poi*, it represents a closed sequence such that the first and last row is considered adjacent and also the elements with the same column index corresponds to adjacent cell. Hence, the descriptor exhibits a cyclical property that is crucial for matching and determining the rotation parameter of the alignment transformation it represents (see section 3.4).

3.3 Similarity Measure

There exist multiple measure, for instance, correlation coefficient, mutual information, join entropy to obtain a quantitative value and are largely dependent on the proportion of overlap area to arrive at a score. To compare the source and target descriptors for each resolution, we rely on a specific similarity measure [Car09]. In point clouds with low overlap region, it is likely that the environment around the corresponding points of source and target scan appear similar resulting in selection of false pairs. To circumvent false matching that could be detrimental to registration accuracy, the similarity measure incorporates information from the non-overlapping region enlarging its descriptive capabilities. The objective of the algorithm is to maximize similarity $S(D_s, D_t)$ in equation 5 while comparing descriptors from two surfaces. $D(s,p)$ and $D(t,q)$ refers to a descriptor with respect to point p,q from two surfaces s, t respectively.

$$\underset{p,q}{argmax}\ S(D(s, p \in s), D(t, q \in t)) \tag{4}$$

$$S(D_s, D_t) = \frac{\varphi_{OL}(D_s,D_t)}{(\alpha * d_{OL}(D_s,D_t)+\beta')+\varphi_{OL}(D_s,D_t)(1-\beta')} \tag{5}$$

The similarity measure $S(D_s, D_t)$ in equation 5 relies on the percentage of overlap (φ_{OL}) between two surfaces and their proximity (d_{OL}), representing the average distance in overlapped area. The affect of overlap in the similarity measure can be modified using β' whereas α directly influences the average distance d_{OL}. The detailed derivation and additional concepts are provided in [Car12] and [Car09].

3.4 Multi-resolution Descriptor and Matching

The descriptor computation in section 3.2 and similarity measure in section 3.3 are core blocks of our hierarchical registration pipeline. The algorithm initially performs a descriptor matching at a coarse resolution of 16×16 with similarity measure described in section 3.3 between two sets of point-of-interest s_{poi} and t_{poi}. The pairs (s^i_{poi}, t^j_{poi}) are sorted in decreasing order of their similarity score at this coarse resolution to obtain a feasible set of starting points for the hierarchical matching of descriptors. We try the highest-ranked pairs first, avoiding initial match ups that are unlikely to yield good results.

The actual process of descriptor comparison then works on some (s^i_{poi}, t^j_{poi}) pair in isolation. We name the initial pair of descriptors at some resolution the *primary* descriptors for source and target, respectively. A search is performed over each cell and its associated point on the surface of the primary source descriptor to find a *secondary* source descriptor that gives the greatest similarity to the target at some row shift k. This row shift represents rotation around the normal of the keypoint that spawned the secondary descriptor, determining the remaining free parameter needed to form a rigid transformation. When the rows of the secondary source descriptor had to be shifted k times for highest similarity, then the rotation angle is $k\rho_\theta$, where ρ_θ is the angular step at the given resolution.

The 3D point and rotation associated with the secondary descriptor that gave the best match at the current resolution becomes the starting point for the primary descriptor at the next higher resolution. Figure 2 depicts a series of primary descriptors from coarse to fine resolution. The process terminates for a specific (s^i_{poi}, t^j_{poi}) pair once the specified maximum resolution is reached.

The final alignment transformation for a points-of-interest pair results from the best-matching secondary source descriptor at the highest resolution. Since we now assume both source and target descriptors to describe the same point on the surface, we can find this alignment by going from world space W into the local reference frame s of the final secondary source descriptor, applying the rotation resulting from the final row shift k, and finally concatenating the transformation back to W from the reference frame t of the target descriptor:

$$\mathrm{T}_{align} = {}^{W}\mathrm{T}_s^{-1} \cdot \mathrm{R}(k) \cdot {}^{W}\mathrm{T}_t \tag{6}$$

The transformation ${}^{W}\mathrm{T}_l$ for some descriptor reference frame l is given directly from the keypoint p that spawned the descriptor and its associated normal n (see section 3.2). The rotational part of the transformation from W to l is obtained as follows:

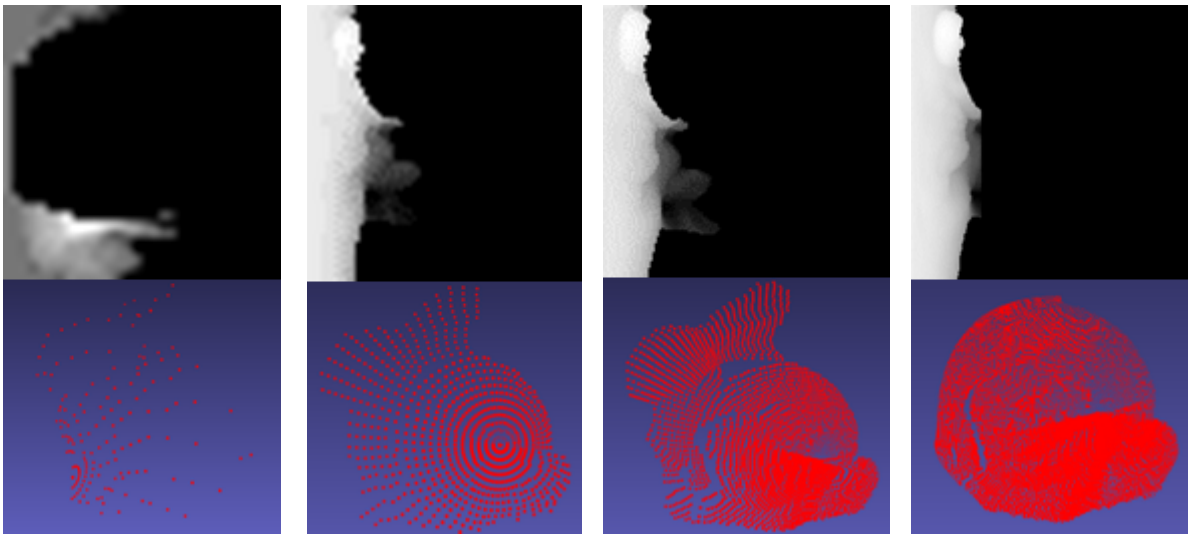
$${}^{W}\mathrm{R}_l = [\vec{x}_l \quad \vec{n} \times \vec{x}_l \quad \vec{n}]^T \tag{7}$$

The full transformation ${}^{W}\mathrm{T}_l$ then follows as

$${}^{W}\mathrm{T}_l = \begin{bmatrix} {}^{W}\mathrm{R}_l & -{}^{W}\mathrm{R}_l \cdot p \\ 0_{1\times 3} & 1 \end{bmatrix} \tag{8}$$

Finally, a rotation matrix parametrized by some row shift k is obtained as follows:

$$\mathrm{R}(k) = \begin{bmatrix} \cos(\rho_\theta \cdot k) & \sin(\rho_\theta \cdot k) & 0 & 0 \\ -\sin(\rho_\theta \cdot k) & \cos(\rho_\theta \cdot k & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$



(a) 32X32 (b) 64X64 (c) 128X128 (d) 256X256

Figure 2: Descriptor cell values mapped to an image at the top and corresponding point cloud represented by the descriptor at the bottom of *Bunny* model. The restriction of higher resolution descriptors to 64 columns is visible in image *(d)*.

Our key contribution of adapting the construction of Circon descriptors to continuous surfaces enables us to extend the hierarchy with much higher resolution descriptors, without being constrained by the original resolution of the point cloud. Like [Car12], we also reduce the search space further down in the hierarchy by halving the number of columns (i.e. radial steps away from the contour center) considered at each level. This is especially important for us since descriptor computation is significantly more expensive, and there is no need to consider points far away from the center of the radial contour at later stages. To mitigate the performance impact of our construction process even further, we also limit descriptor construction to 64 columns (see figure 2d). Every cell further away than 64 radial steps will thus be invalid, which the similarity measure is able to handle naturally as part of its down-weighting of non-overlapping regions (see section 3.3).

3.5 Stopping Criterion

The stopping criterion is essential for the iterative search to be convergent and importantly must embed characteristics of the descriptor. To this end, we choose three non-colinear points from the final source and target descriptors that each form a triangle around the respective Circon descriptor center. Such points can be easily obtained by selecting the same three equidistantly spaced cells from the first column of each descriptor and computing their 3D point equivalents. These triangles form a fictitious correspondence pair in accordance with Ferrero et al. [Car12]. The centroid of each triangle defines a local reference frame with fictitious transformation T_{fict} between them as represented by equation 6. T_{fict} is compared with rigid transformation T_{align} in terms of a delta transformation:

$$\Delta T = T_{align} \cdot T_{fict}^{-1} \tag{10}$$

This delta transformation is evaluated for rotational and translational difference separately (see **Appendix A** for details). If the rotational difference Δr and translation distance Δt_s extracted from ΔT are smaller than their respective thresholds, the algorithm terminates and T_{align} is returned as the final alignment transformation.

4 EVALUATION

To demonstrate the effectiveness of the proposed hierarchical registration method (HER), we perform our analysis on a publicly available dataset and compare quantitative measure with state-of-the art registration method and and also the hierarchical method proposed by Ferrero et al. [Car12] and for simplicity name it as ORI−HER. We ran all our experiments on an Intel Core i5-6500 clocked at 3.20GHz. The following section presents how we performed our evaluation including dataset description and the tested methods.

4.1 Dataset

We performed an evaluation on the comprehensive benchmark provided[2] by Petrelli et al. [Pet15], which we refer to as the LRF dataset. This dataset contains multiple objects captured by sensors of different quality, ranging from consumer-level depth cameras to professional laser scanners resulting in point clouds with varying point density. Moreover, the benchmark includes challenging low-overlap registration pairs.

[2] http://www.vision.deis.unibo.it/list-all-categories/78-cvlab/108-pairwiseregistrationbenchmark

(a) ORI−HER (b) FGR (c) HER

Figure 3: Registration of a view-pair From LRF *OilPump* dataset

(a) ORI−HER (b) FGR (c) HER

Figure 4: Registration of a view-pair From LRF *WoodChair* dataset

4.2 Methods

We base our comparisons with ORI−HER [Car12] that uses hierarchical descriptors with similarity measure for coarse point cloud registration and state-of-the-art global registration method FGR [Zho16]. This methods has been shown to rival the accuracy of local methods and therefore ideal for comparison to understand the difference of accuracy with our proposed approach. The parameters for each methods and their respective values adapted during the evaluation process are described in separate **Appendix-A**.

4.3 Metrics

After solving the various registration problems with different methods based on the dataset discussed in section 4.1, we evaluate how well the results match the ground truth transforms with a data-dependent metric. We use root mean square error (RMSE) as a metric to quantify registration accuracy. The RMSE measures the mean error over all source cloud point locations for some computed transformation T_{out} with respect to ground truth transformation T_g:

$$\text{RMSE}(T_{out}) = \frac{1}{\overline{d}} \sqrt{\frac{1}{n} \sum_{i=1}^{n} \|T_{out} p_i - T_g p_i\|^2} \qquad (11)$$

The normalization factor $\overline{d}$ represents the sampling distance. For the LRF dataset, where scans are represented as polygon meshes, we use the average mesh edge length.

5 RESULTS

The section elucidates on the results of our experiments that verify the proposed approach produces more accurate registration results than state-of-the art approach. We will also examine various characteristics of our algorithm.

5.1 Accuracy

The accuracy comparison is based on datasets introduced in section 4.1 with multiple registration problems and RMSE defined in section 4.3 on the registration output after convergence. We categorize RMSE results into *fine* ($< 20\overline{d}$), *coarse* ($[20\overline{d}, 50\overline{d}]$), *failed* (rest) and measure the number of view pairs (Table 1) that fall under these categories with respect to the total number of registration problems. To refine the comparison, we concentrate on the *fine* registration view-pairs and Figure 5 shows the accuracy results for the LRF datasets falling under this category. Figure 5 asserts that the highest accuracy are obtained by our method compared to FGR and ORI−HER. Furthermore, The statistical measures in Table 2 based on *fine* RMSE delineates lowest values for our method among others. The continuous representation of scans followed by descriptor computation with finer resolution attributes to the significant improvement in the accuracy. Figure 3 and Figure 4 depict particularly difficult registration problems from the LRF dataset that FGR and ORI−HER failed to align, while our method achieved *fine* alignment.

	No.Pairs	Fine	Coarse	Fail
ORI−HER	1840	57	427	1356
FGR	1840	279	642	919
HER	1840	355	711	774

Table 1: Categorization of registration view pairs from LRF datasets based on threshold as discussed in section 5.1

	Min	Max	Avg	Std.dev
ORI−HER	1.162	23.66	5.99	4.07
FGR	0.48	21.67	3.49	2.77
HER	0.34	20.24	2.58	2.25

Table 2: Statistical Comparison of RMSE based on Fine threshold as discussed in section 5.1

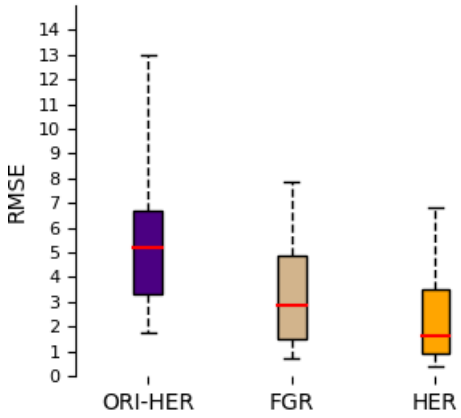


Figure 5: Accuracy comparison of pairwise registration methods for the LRF datasets. The box height represents the 25th and 75th percentile. The whiskers represent the 5th and 95th percentiles. The middle line represents the median. Better registration results are characterized by lower RMSE.

5.2 Runtime

We examined the computational time of ORI−HER, FGR and our approach taking into account the preprocessing steps and the core-algorithm. An analysis was performed on 10 randomly selected models from the LRF dataset. For our proposed approach, we consider three major components, i.e surface fitting, coarse resolution sorting and the iterative core algorithm. As depicted in the Figure 6, the runtime for sorting is negligible with respect to the other two components. The FGR algorithm relies on feature descriptors from a preprocessing step, which tend to be expensive to compute. Time computing the alignment is spend largely on matching these features for FGR and varies a lot depending on the point clouds. Nevertheless, it is the fastest method on average. ORI−HER exhibits the highest runtime across the comparison. We observed that it has to run through a large number of keypoint pairs (and thus, descriptor hierarchies) to determine an optimal transformation. While descriptors are significantly

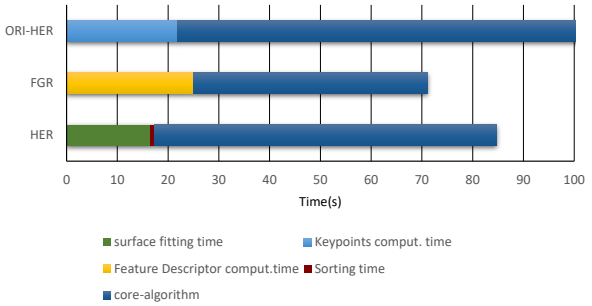


Figure 6: Runtime comparison of the proposed approach with ORI−HER, FGR including the preprocessing step for each methods. Our method exhibits higher runtime in the core algorithm part owing to its multiple descriptor computation compared to FGR but considerably lesser to ORI−HER

more expensive to compute for our method, we typically find an optimal solution within the first few keypoint pairs, as we can use higher resolution descriptors independent of the sampling situation, greatly speeding up convergence.

6 CONCLUSION AND OUTLOOK

In this paper, we have presented a hierarchical (coarse-to-fine) global registration method. The 2D descriptors built from a continuous representation of each point cloud enables us to embed higher accuracy into them. Our evaluation based on the diverse LRF dataset shows considerable improvement in the accuracy compared to the state-of-the-art FGR algorithm.

Nonetheless, a handful of improvements could be made, most obviously concerning speed. We chose NURBS surfaces for their versatility and ready availability of implementations, but the fitting process is expensive. Furthermore, as described in section 3.2, intersecting rays with a NURBS surface requires, in general, a numeric solution. A more specialized surface representation could accelerate both fitting and descriptor building, in turn making the use of higher resolution descriptors more feasible and thus improving accuracy even further.

Alternatively, leveraging GPU hardware to compute the descriptors promises immense speedups. This could be achieved using the programmable rasterization pipeline to generate orthographic depth maps of the continuous surfaces as viewed in the descriptor reference frame. The resolution of these depth maps can be chosen as high as needed to enable artifact-free sampling at the Circon cell centers. We expect this would bring down the computation time spend on building descriptors to a fraction of what it is in our current implementation.

7 REFERENCES

[Ale12] Alexandre L. A.,3d descriptors for object and category recognition: A comparative evaluation,

In Proceedings of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 1, Vilamoura, Portugal, 2012.

[And01] Anderson, R.E. Social impacts of computing: Codes of professional ethics. Social Science, pp.453-469, 2001.

[Bel14] Bellekens B., Spruyt V., Maarten Weyn R. B.: A survey of rigid 3d point cloud registration algorithms. In Fourth International Conference on Ambient Computing, Applications, Services and Technologies, Proceedings, IARA, pp.8-13, 2014.

[Bes92] Besl, P. J. and McKay, N. D. A method for registration of 3-d shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 1992, 14(2):239-256

[Chen91] Chen Y., Medioni G.: Object modeling by registration of multiple range images. In Proceedings. 1991 IEEE International Conference on Robotics and Automation (Apr 1991), pp. 2724-2729

[Car12] Torre-Ferrero Carlos, Garcia Jose R. Llata, Alonso Luciano, Robla Sandra, Sarabia Esther G. :3D point cloud registration based on a purpose-designed similarity measure. EURASIP J. Adv. Signal Process. 2012: 57 (2012)

[Car09] Torre-Ferrero Carlos, Garcia Jose R. Llata, Robla Sandra, Sarabia Esther G., A similarity measure for 3D rigid registration of point clouds using image-based descriptors with low overlap. S3DV09, in IEEE 12th International Conference on Computer Vision, ICCV Workshops 2009, Kyoto, Japan, pp. 71-78 (2009)

[Goj09] Gojcic, Z., Zhou, C., Wegner, J.D., Guibas, L.J. and Birdal, T., 2020. Learning multiview 3d point cloud registration. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 1759-1769).

[Goj19] Gojcic Zan, Zhou Caifa, Wegner Jan D, and Wieser Andreas. The perfect match: 3d point cloud matching with smoothed densities. In 16 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5545-5554, 2019.

[Guo16] Guo Y., Bennamoun M., Sohel F., Lu M., Wan J.,and Kwok N. M., A comprehensive performance evaluation of 3d local feature descriptors, International Journal of Computer Vision, vol. 116, no. 1, pp. 66-89, 2016

[Had14] Hadji I.and De G. N., Local-to-global signature descriptor for 3d object recognition, In Proceedings of the Asian Conference on Computer Vision, pp. 570-584, Springer, 2014.

[Hao18] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 195-205, 2018.

[Han18] Hana, X.-F.; Jin, J.S.; Xie, J.; Wang, M.-J.; Jiang, W. A Comprehensive Review of 3d Point Cloud Descriptors.arXiv 2018, arXiv:1802.02297.

[Han18a] Han X.-F., Sun S.-J., Song X.-Y., and Xiao G.-Q., 3d point cloud descriptors in hand-crafted and deep learning age:state-of-the-art, 2018, http://arxiv.org/abs/1404.3978.

[Hua21] Huang X., Mei G., Zhang J., and Abbas R. A comprehensive survey on point cloud registration. arXiv preprint arXiv:2103.02690, 2021.

[Hua11] Huang Y, Z. L., Tan Z, W. Q., and L., C. Adaptive moving least-squares surfaces for multiple point clouds registration. ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference 2011, pages 105-113.

[Hua09] Huang, H., Li, D., Zhang, H., Ascher, U., and Cohen-Or, D. Consolidation of unorganized point clouds for surface reconstruction. ACM transactions on graphics (TOG) (2009), 28(5):176.

[Hua22] Huang, J., Birdal, T., Gojcic, Z., Guibas, L.J. and Hu, S.M., 2022. Multiway Non-rigid Point Cloud Registration via Learned Functional Map Synchronization. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[Joh99] Johnson, Andrew E., and Hebert Martial. Using spin images for efficient object recognition in cluttered 3D scenes. Pattern Analysis and Machine Intelligence, IEEE Transactions on 21, no. 5 1999: 433-449

[Kub12] Kubacki, D., Bui, H., Babacan, S., and Do, M. Registration and integration of multiple depth images using signed distance function. In Computational Imaging X 2012, volume 8296,

[Lip07] Lipman, Y., Cohen-Or, D., Levin, D., and Tal-Ezer, H .Parameterization-free projection for geometry reconstruction. ACM Transactions on Graphics (TOG) (2007), 26(3):22.

[Liu06] Liu, Y.-S., Paul, J.-C., Yong, J.-H., Yu, P.-Q., Zhang, H.,Sun, J.-G., and Ramani, K. Automatic least squares projection of points onto point clouds with applications in reverse engineering. Computer-Aided Design 2006, 38(12):1251-1263.

[MPD06] Makadia A., Patterson A., Daniilidis K.: Fully automatic registration of 3D point clouds. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2006)

[Mit04] Mitra, N. J., Gelfand, N., Pottmann, H., and Guibas, L. Registration of point cloud data from a geometric optimization perspective. In Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing, pages 22-31. ACM.

[Moe13] Mörwald T. and Vincze M., Object Modelling for Cognitive Robotics, PhD Thesis, Vienna University of Technology, Austria, 2013

[Mun07] El Munim, H. A. and Farag, A. A. Shape representation and registration using vector distance functions. In Computer Vision and Pattern Recognition, 2007. CVPR 2007. IEEE Conference on, pages 1-8.

[Nel65] Nelder, J.A. and Mead, R. (1965), A simplex method for function minimization, Comput. J., 7, pp. 308-313.

[Par03] Paragios, N., Rousson, M., and Ramesh, V. Nonrigid registration using distance functions. Computer Vision and Image Understanding 2003., 89(2-3):142-165.

[Pet15] Petrelli A., DI Stefano L.: Pairwise registration by local orientation cues. Computer Graphics Forum 35, 2015, 59-72.

[Pot06] Pottmann, H., Huang, Q.-X., Yang, Y.-L., and Hu, S.-M. Geometry and convergence analysis of algorithms for registration of 3d shapes. International Journal of Computer Vision 2006, 67(3).

[Ran20] Ran Y. and Xu X., Point cloud registration method based on sift and geometry feature, Optik, vol. 203, Article ID 163902, 2020.

[Rad09] Rusu Radu Bogdan ,Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments, PhDthesis, Computer Science department, Technische Universitaet Muenchen, Germany,2009

[Rus01] Rusinkiewicz S., Levoy M. Efficient variants of the icp algorithm. In Proceedings Third International Conference on 3-D Digital Imaging and Modeling, pp.145-152, 2001.

[Rus11] Rusu R. B. and Cousins S., 3D is here: Point Cloud Library (PCL), IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 2011

[Sal07] Salvi J., Matabosch C., Fofi D., Forest J.: A review of recent range image registration methods with accuracy evaluation. Image and Vision Computing 25,pp 578-596, 2007.

[Tam13] Tam, G. K. L., Cheng, Z. Q., Lai, Y. K., Langbein, F. C.,Liu, Y., Marshall, D., Martin, R. R., Sun, X. F., and Rosin, P. L. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. IEEE Transactions on Visualization and Computer Graphics 2013, 19(7):1199-1217.

[Tom10] Tombari F., Salti S.,DI Stefano L.Unique shape context for 3D data description. In Proceedings of ACM Eurographics Workshop on 3D Object Retrieval 2010, pp. 57-62.

[Wan19] Wang Lingjing, Jianchun Chen, Xiang Li, and Yi Fang. Non-rigid point set registration networks. arXiv preprint arXiv:1904.01428, 2019.

[Yua20] Wentao Yuan, Eckart Benjamin, Kihwan Kim, Varun Jampani, Fox Dieter, and Kautz Jan. Deepgmr: Learning latent gaussian mixture models for registration. In European Conference on Computer Vision, pages 733-750. Springer, 2020.

[Yan19] Zhenpei Yang, Jeffrey Z. Pan, Linjie Luo, Zhou Xiaowei ,Grauman Kristen, and Qixing Huang. Extreme relative pose estimation for rgbd scans via scene completion. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

[Yan06] Yang, Y.-L., Lai, Y.-K., Hu, S.-M., and Pottmann, H.Robust principal curvatures on multiple scales. In Proceedings of the Fourth Eurographics Symposium on Geometry Processing, SGP 2006, pages 223-226, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

[Zah12] Zaharescu A., Boyer E., Horaud R.: Keypoints and local descriptors of scalar functions on 2D manifolds. International Journal of Computer Vision 100, 1 2012, 78-98.

[Zen17] Zeng Andy, Shuran Song, Niessner Matthias, Fisher Matthew, Xiao Jianxiong, and Funkhouser Thomas. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1802-1811, 2017.

[Zho16] Zhou, Q.-Y., Park, J., and Koltun, V. Fast global registration. In Computer Vision-ECCV 2016, Leibe B., Matas J., Sebe N.,Welling M., (Eds.), Springer International Publishing pages 766-782.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Spatiotemporal redundancy removal in immersive video coding

Adrian Dziembowski [1]
adrian.dziembowski@put.poznan.pl
Dawid Mieloch [1]
Marek Domański [1]
Gwangsoon Lee [2]
Jun Young Jeong [2]

[1] Institute of Multimedia Telecommunications, Poznań University of Technology
Polanka 3, 61-131 Poznań, Poland

[2] Electronics and Telecommunications Research Institute
Daejeon, Republic of Korea

ABSTRACT

In this paper, the authors describe two methods designed for reducing the spatiotemporal redundancy of the video within the MPEG Immersive video (MIV) encoder: patch occupation modification and cluster splitting. These methods allow optimizing two important parameters of the immersive video: bitrate and pixelrate. The patch occupation modification method significantly decreases the number of active pixels within texture and depth video produced by the MIV encoder. Cluster splitting decreases the total area needed for storing the texture and depth information from multiple input views, decreasing the pixelrate. Both methods proposed by the authors of this paper were appreciated by the experts of the ISO/IEC JTC1/SC29/WG11 MPEG and are included in the Test Model for MPEG Immersive video (TMIV), which is the reference software implementation of the MIV standard.

Keywords

Immersive video coding, multiview compression, virtual reality.

1. INTRODUCTION

Recently, there is a common interest in immersive video [Isg14] and virtual reality systems, where a user virtually immerses into the scene. Such systems are an evolution of previous free-viewpoint television and free navigation systems [Tan12], [Sta18], where a user may virtually navigate around the scene.

In the immersive video system, a scene is acquired by a set of multiple precisely calibrated [Tao21] cameras. The number of cameras may vary, depending on the system, from less than ten [Mie20b] to even hundreds of cameras [Fuj06].

However, even in the most expensive systems equipped with dozens of cameras, the user should not be limited to watch the videos explicitly captured by the cameras. In order to provide smooth virtual navigation, the user should be able to choose his or her own viewport, which has to be rendered [Ceu18], [Fac18], [Zhu19] using data from input cameras.

Such a rendering requires the creation of the 3D model of the scene, i.e., calculation of the exact position of each captured object. The 3D scene model can be stored using various representations, e.g., meshes, voxels, or point clouds [Cui19], [Zha20], but the most commonly used representation is the MVD (multiview video plus depth) [Mül11]. In the MVD representation each input view is complemented by the corresponding depth map (either captured by time-of-flight cameras or estimated based on input views [Mie20a]).

Obviously, the user of the immersive video system has to receive the multiview content, i.e., multiple views, corresponding depth maps, and exact camera parameters. Without efficient compression, such content would require hundreds of megabits per second of the video, making the system highly impractical. The most straightforward method of the compression of the multiview content is performing the simulcast encoding, i.e., using separate instances of the video encoder (e.g., the newest VVC [Bro21]) for each input view and depth map. However, such an approach does not reduce the inter-view redundancy of input videos, thus wastes bits for coding of unnecessary information.

A better solution is to use dedicated video encoders, which utilize the similarity between several input views, e.g., MVC [Nem10], MV-HEVC, or 3D-HEVC [Tec15], which are the multiview extensions of the AVC [Sul05] and HEVC [Sul12] encoders. However, these techniques either restrict the camera arrangement (3D-HEVC, which allows compression of multiview video captured by linear camera systems) or do not efficiently use the information about the 3D

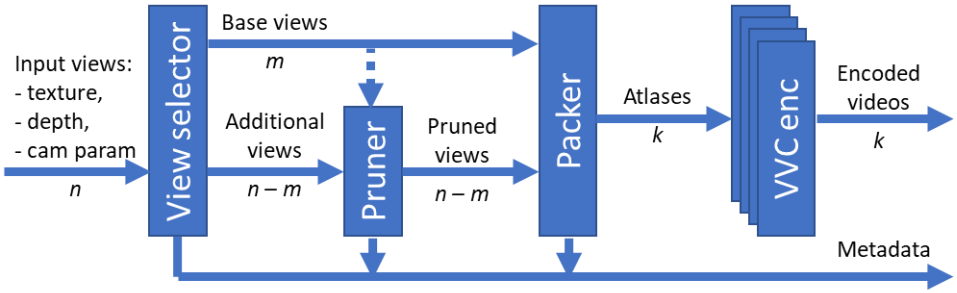


Figure 1. Simplified scheme of the MIV encoder.

scene model (MVC and MV-HEVC, which do not use depth maps for inter-view redundancy removal).

Mentioned flaws of existing multiview coding techniques motivated the development of the new technique – MPEG Immersive video (MIV) [Boy21], dedicated for any type of immersive video, including simple free navigation, free-viewpoint television, and virtual reality systems, where a user immerses into the 3D scene using the head-mounted device (HMD). The MPEG Immersive video is being developed by the ISO/IEC JTC1/SC29 WG04 MPEG VC group since 2019 and became a standard this year [ISO22].

2. MPEG IMMERSIVE VIDEO

The purpose of the MPEG Immersive video standard is to remove the inter-view consistency of the multiview video. As presented in Fig. 1, MIV is designed to be a preprocessing step before the actual video compression, which is performed using a typical video encoding algorithm, e.g., VVC. However, MIV is codec agnostic, so another video encoders (HEVC, AVC, or even M-JPEG) may be used as well.

The input data for the MIV encoder are n input views (including texture, depth map, and camera parameters for each input view).

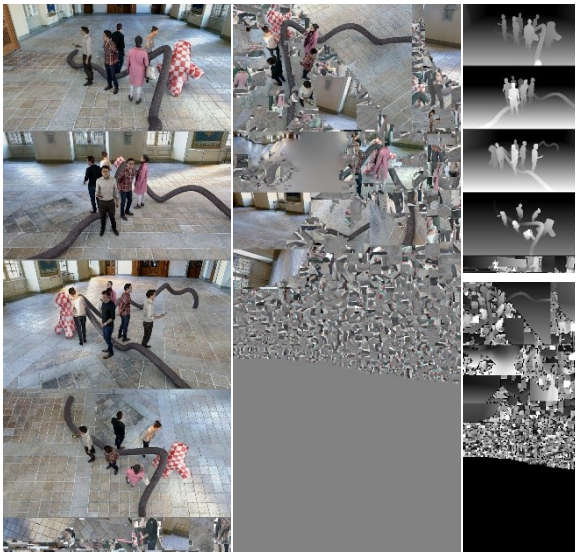


Figure 2. Atlases for sequence Group: two texture atlases and two depth atlases (reduced resolution).

Based on these data, MIV creates k atlases – videos containing information from n input views. An example of atlases is presented in Fig. 2.

At the first step, the MIV encoder chooses views, which will be sent without inter-view redundancy removal. These views are called "base views" and are pasted into the first atlas as full views. The base views are selected automatically based on camera parameters to cover the possibly largest part of the scene. Any other view ("additional view") is pruned in order to reduce the inter-view redundancy.

The pruning operation is performed by reprojecting pixels between views. Any pixel of an additional view is removed (pruned) if its depth and color are similar to the depth and color of the pixel reprojected to its position from base views and other (already pruned) additional views.

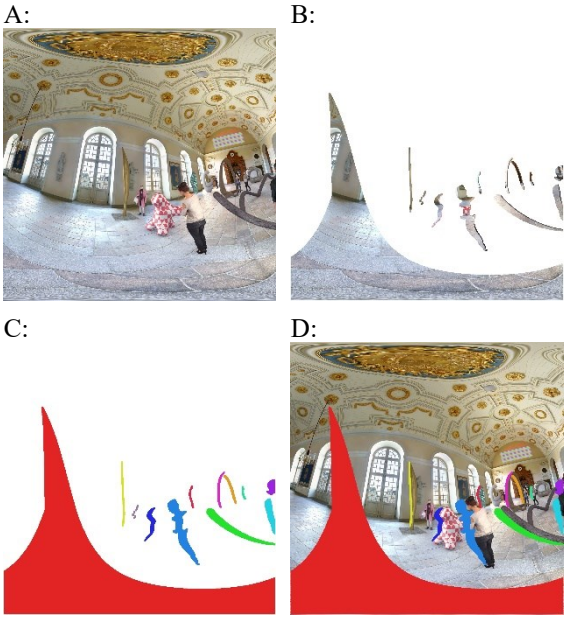


Figure 3. Pruning and clustering; A: input view (sequence Museum), B: view after pruning, C: preserved pixels after clustering, D: preview of clusters within input view.

All the preserved (non-pruned) pixels of each view are then merged into consistent clusters containing mutually connected pixels. An example of pruning

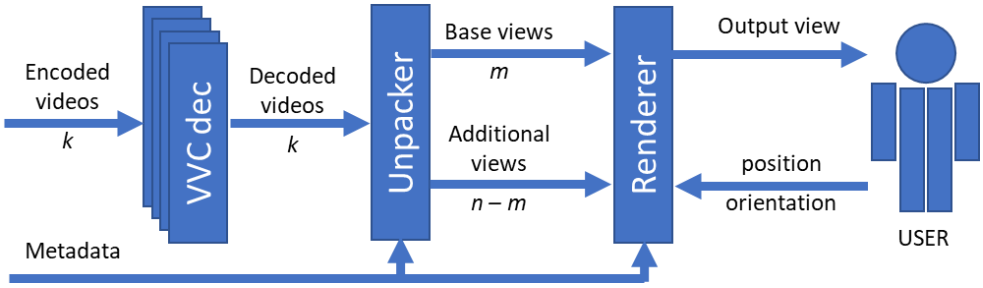


Figure 4. Simplified scheme of the MIV decoder.

and clustering is presented in Fig. 3. In Fig. 3A, an additional view is shown. All inter-view redundant pixels were pruned and only areas non-visible in other views were preserved (Fig. 3B). Fig. 3C presents the effect of pixel clustering, where each cluster was colored differently. In Fig. 3D, colored clusters were pasted into the input view to highlight, which areas of the view were preserved (disocclusions behind foreground objects and the bottom part of a view, which was out of field of view in other cameras).

In the next step, all the clusters are packed into atlases as "patches", containing a cluster together with its bounding box.

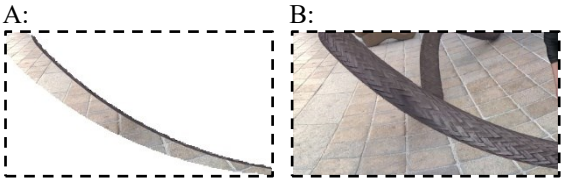


Figure 5. Cluster vs. patch; A: green cluster from Fig. 3, B: patch containing texture information for the cluster and its entire bounding box.

The packing process tries to efficiently fit non-pruned information from all input views in atlases, significantly reducing the total pixelrate (total number of pixels that have to be processed by the decoder) of the video (compared to the pixelrate of input video).

Of course, the packing operation has to be reversible in order to allow the unpacking of the atlas at the decoder side (Fig. 4). Reversibility of the packing process is provided by sending additional metadata for each patch, including its size, position within the input view, position within the atlas, and input view number.

In the last step, each atlas is separately encoded by the typical video encoder, e.g., VVC. The MIV standard [ISO22] describes also the process of multiplexing video bitstreams with metadata, as well as many other minor video processing techniques providing more efficient encoding of immersive video. However, this paper does not focus on them. The detailed description of the MIV encoder can be found in [Boy21] or [MPEG21].

On the decoder side, *k* video bitstreams are decoded using a typical video encoder (e.g., VVC). After video decoding, the input views are restored by unpacking patches from the atlases. Base views are restored completely. Restored additional views have many unoccupied areas, which were pruned in the encoder.

The last step of the decoding is the rendering of the view being watched by the user of the immersive video system. The user provides his or her position and orientation, and the renderer creates demanded virtual view.

3. PATCH OCCUPATION MODIFICATION

As presented in Fig. 5, a patch is a rectangular fragment of the input view, containing a cluster of non-pruned pixels together with its entire bounding box (Fig. 6A).

Such an approach has a major flaw: when the video encoder processes an entire patch, it wastes many bits for encoding useless texture information (pixels qualified as inter-view redundant by the pruner).

On the other hand, patches could contain only the non-redundant pixels (Fig. 6B), i.e., any pixel outside of the cluster could be greyed out and signaled as unoccupied by setting its depth value to a restricted level [MPEG21].

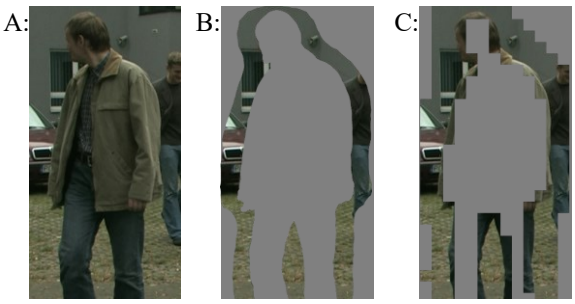


Figure 6. Various approaches to patch occupation; A: fully occupied patch, B: patch containing only non-pruned pixels, C: patch with modified occupation; sequence Carpark.

However, clusters have irregular shapes and thus are more difficult to efficiently encode by the video encoder, which has to handle many irregular edges between preserved and pruned (greyed out) areas. Moreover, the shape of a cluster changes in time because of the movement of objects in the scene, additionally reducing the efficiency of the inter-frame prediction.

We proposed an encoder-oriented solution, which adapts to the grid of the coding tree in the video encoder. In the proposed approach, the cluster is divided into blocks (e.g., blocks of size 16×16 pixels), and the entire block is greyed out if it does not contain any preserved pixels. Otherwise, all pixels within the block have a texture (Fig. 6C).

Figs. 7 and 8 compare both texture atlases created using two approaches: default with fully occupied patches (Fig. 7) and the proposed one (Fig. 8).

Figure 7. Texture atlases without patch occupation modification (sequence Frog).

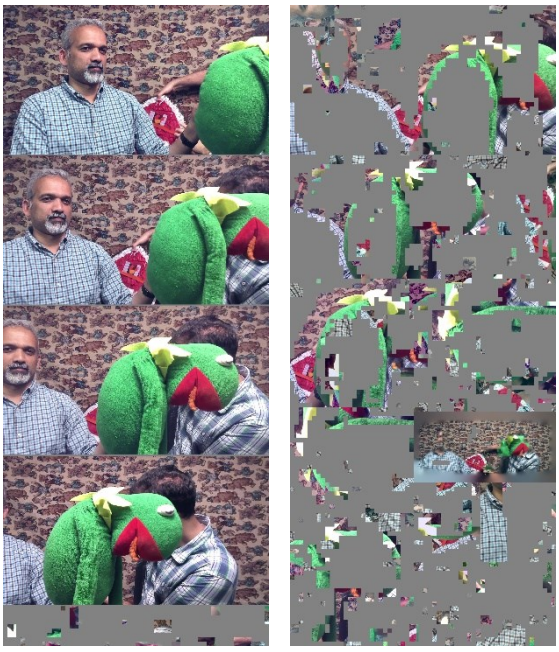


Figure 8. Texture atlases with patch occupation modification (sequence Frog).

As presented, the proposed modification significantly reduces the number of non-grey pixels within the second atlas. It does not change base views in the first atlas, as they are not pruned (cf. Fig. 1).

Regarding the temporal domain, the active blocks within atlases change over time, slightly decreasing the inter-frame prediction efficiency, but due to the fact, that the patch position does not change in consecutive frames, active blocks still have a similar texture and the decrease is very slight (Fig. 9).

Figure 9. Fragment of the second atlas from Figs. 7 and 8, frames 0 and 10; top: with fully occupied patches, bottom: with proposed patch occupation modification.

The method proposed by the authors of this paper was appreciated by the ISO/IEC MPEG VC experts [Dzi20b] and is included in the Test Model for MPEG Immersive video (TMIV) [MPEG21], which is the reference software implementation of MIV.

4. CLUSTER SPLITTING

The method of changing patch occupancy can decrease the bitrate needed for encoding of the atlases (especially the second one, as it does not contain base views), but it does not change the second crucial parameter of the practical immersive video system – the pixelrate, which defines the total number of pixels that have to be decoded. Therefore, we proposed a second technique, which allows reducing this parameter by allowing the splitting of large irregular clusters, e.g., the big red cluster presented in Fig. 3.

If a cluster is L-shaped (Fig. 10A), the patch containing this cluster has many unoccupied pixels (red area in Fig. 10B). If such a cluster will be split into two smaller clusters, the total area of two patches may be significantly smaller (red and blue areas in Fig. 10D).

The split line is parallel to the shorter side of the patch (Fig. 10C) and is placed in a position, which minimizes the total area of patches after the split. If the total area of patches after the split is similar to the area before splitting, the cluster is not split.

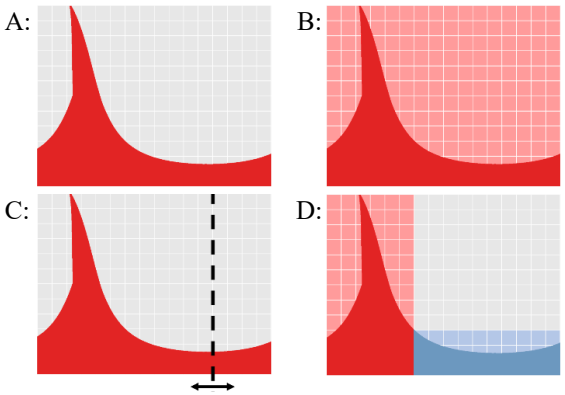


Figure 10. Splitting of the L-shaped cluster; A: initial cluster, B: patch for cluster A, C: the splitting of cluster A, D: cluster A split into two smaller clusters.

If the cluster has irregular contour but is not L-shaped (Fig. 11A), a different splitting algorithm is performed. For such a patch, occupied and non-occupied areas are being compared. If most pixels within the patch are non-occupied, the cluster is split into two halves, along the line parallel to the shorter side of the patch (Fig. 11B), resulting in two smaller clusters (Fig. 11C).

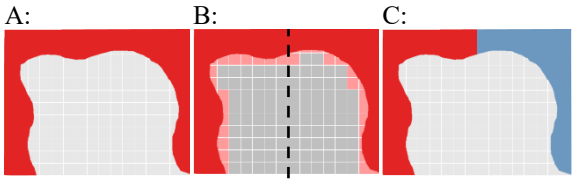


Figure 11. Splitting of the C-shaped cluster; A: initial cluster, B: the splitting of cluster A, C: cluster A split into two smaller clusters.

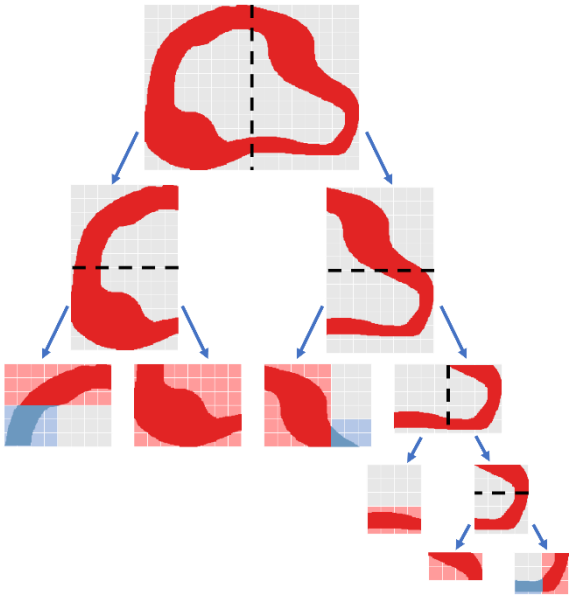


Figure 12. Recursive splitting of irregular cluster.

As presented in Fig. 11, the result of the splitting of the C-shaped cluster is two L-shaped clusters. To provide a reduction of the total area of patches, such clusters have to be split again, as shown in Fig. 10. Multiple splitting of a cluster is possible due to the recursiveness of the proposed method – each cluster is split until the splitting significantly minimizes the total area of the patches (Fig. 12).

As presented in Figs. 13 and 14, the proposed method of cluster splitting allows to significantly decrease the total occupied area of the atlas (the non-occupied, grey area in the second atlas in Fig. 14 is much bigger than the non-occupied area in Fig. 13).

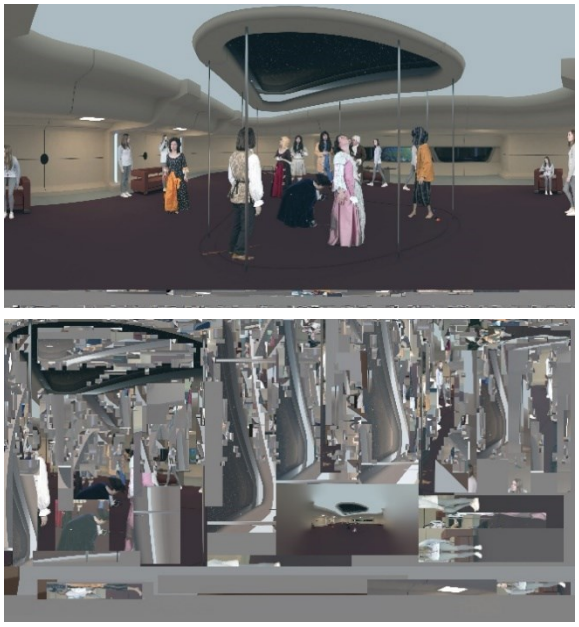


Figure 13. Two atlases without cluster splitting (sequence Hijack).

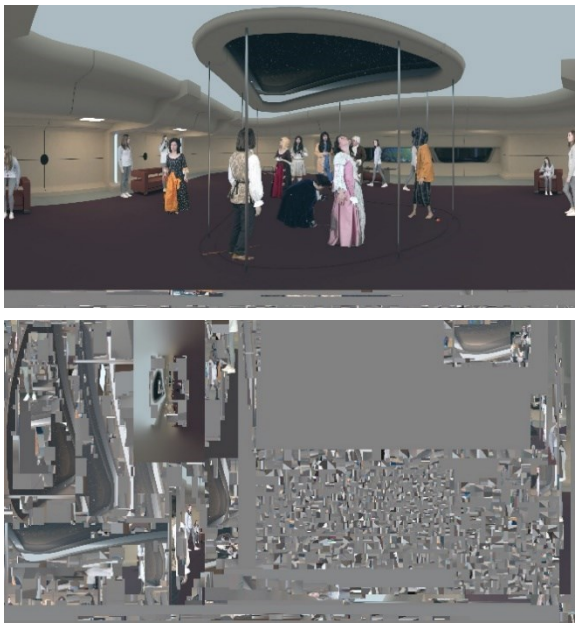


Figure 14. Two atlases with cluster splitting (sequence Hijack).

Similarly to the method presented in the previous section, also the cluster splitting method proposed by the authors of this paper was appreciated by the ISO/IEC MPEG VC experts [Dzi20a] and is included in the Test Model for MPEG Immersive video (TMIV) [MPEG21].

5. EXPERIMENTAL RESULTS

Both techniques presented in this paper were tested under the common test conditions for MPEG Immersive video (MIV CTC) [MPEG22], on 16 miscellaneous test sequences, including both natural content (NC) and computer-generated (CG) sequences of different resolutions, the number of cameras, and camera types (perspective and omnidirectional, represented in equirectangular projection – ERP). Key parameters of the test set are presented in Table 1.

Sequence name	Views	Type	Resolution	Source
Cadillac	15	NC/Persp.	1920×1080	[Dor21a]
Carpark	9	NC/Persp.	1920×1088	[Mie20b]
Chess	10	CG/Omni	2048×2048	[Ilo19]
ChessPieces	10	CG/Omni	2048×2048	[Ilo20]
ClassroomVideo	16	CG/Omni	1920×1080	[Kro18]
Fan	15	NC/Persp.	1920×1080	[Dor20a]
Fencing	10	NC/Persp.	1920×1080	[Dom16]
Frog	13	NC/Persp.	1920×1080	[Sal18]
Group	21	NC/Persp.	1920×1080	[Dor20b]
Hall	9	NC/Persp	1920×1088	[Mie20b]
Hijack	10	CG/Omni	4096×2048	[Dor18]
Kitchen	25	CG/Persp.	1920×1080	[Boi18]
Mirror	15	CG/Persp.	1920×1080	[Dor21b]
Museum	24	CG/Omni	2048×2048	[Dor18]
Painter	16	NC/Persp	2048×1088	[Doy18]
Street	9	NC/Persp	1920×1088	[Mie20b]

Table 1. Test sequences used in experiments.

To present a variety of content within the test set, Figs. 15 and 16 contain a single frame from each sequence.

Figure 15. Natural sequences. Left column: Carpark, Street, Frog; right column: Hall, Fencing, Painter.

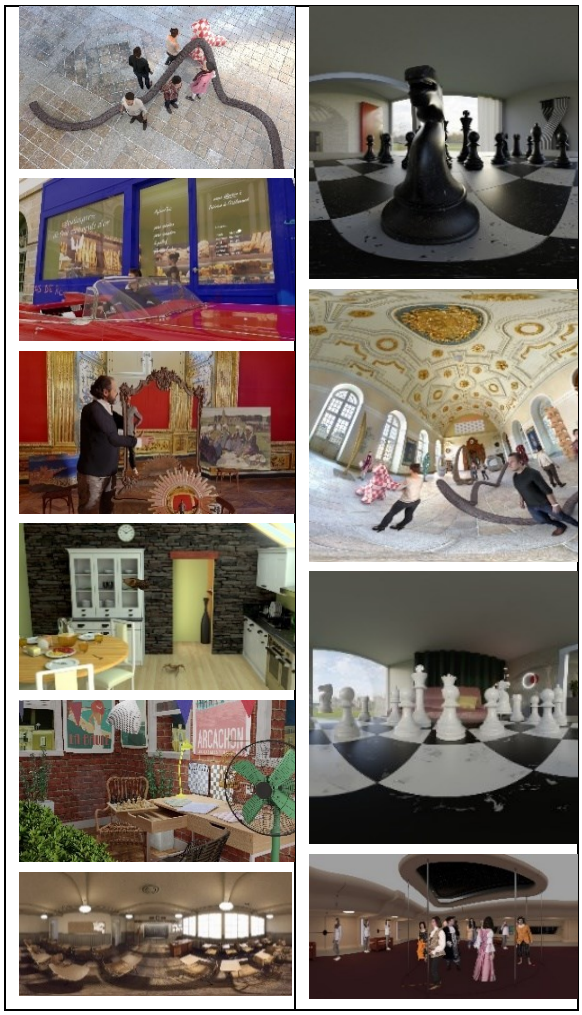


Figure 16. Computer-generated sequences. Left column: Group, Cadillac, Mirror, Kitchen, Fan, and ClassroomVideo; right column: Chess, Museum, ChessPieces, and Hijack.

Table 2 shows the gain of the proposed patch occupation modification method presented as bitrate reduction (compared to an atlas with fully occupied patches), separately for texture atlases, depth atlases, and total reduction for all video bitstreams.

As presented, the proposed method allows to significantly decrease the total bitrate needed for the representation of encoded video bitstreams, especially for higher bitrates (25 Mbps, on average). For low bitrates, the reduction is lower but noticeable.

Different efficiency between low and high bitrate was expected, as for higher bitrates, the encoder tries to encode all the high-frequency details of the video (and fully occupied patches have much more details than plain grey area), while for lower bitrates the details are destroyed (so the number of bits needed for encoding of grey area and highly compressed texture and depth is more similar).

Differences between various sequences, which can be spotted for low bitrates are caused by the sequence

characteristics. For example, texture of the Fan sequence is very detailed and has many areas with high frequencies which are very hard to be encoded at low bitrates. Therefore, when these areas are greyed out, the encoding is more efficient. On the other hand, for less-detailed sequences (e.g., ChessPieces) proposed technique increases a number of high frequencies (by adding edges between occupied and non-occupied regions), decreasing the efficiency of the VVC at higher QPs.

Test sequence	Bitrate reduction (patch occupation modification vs. fully occupied patches)					
	High bitrate (~25 Mbps)			Low bitrate (~7 Mbps)		
	Texture	Depth	All	Texture	Depth	All
Carpark	5.0%	5.0%	5.1%	0.9%	3.8%	3.0%
Fencing	4.6%	3.5%	4.3%	-6.1%	-1.3%	-2.6%
Frog	26.4%	14.1%	22.2%	24.2%	9.5%	17.4%
Hall	4.7%	9.5%	8.5%	-8.2%	7.1%	5.1%
Painter	10.5%	9.6%	10.1%	5.6%	5.6%	5.6%
Street	7.0%	8.4%	7.4%	6.0%	8.1%	7.0%
NC: Average	**9.7%**	**8.3%**	**9.6%**	**3.7%**	**5.5%**	**5.9%**
Cadillac	6.5%	4.8%	6.3%	-7.0%	-15.6%	-13.1%
Fan	25.0%	35.6%	32.5%	17.5%	34.2%	32.1%
Group	7.6%	3.3%	6.8%	-4.0%	-1.9%	-2.4%
Kitchen	14.6%	12.8%	14.3%	8.4%	9.0%	8.7%
Mirror	-2.1%	1.8%	-0.6%	-6.2%	0.1%	-1.7%
CG-P: Average	**10.3%**	**11.7%**	**11.9%**	**1.7%**	**5.2%**	**4.7%**
Chess	14.8%	4.3%	12.7%	2.8%	-6.5%	-1.0%
ChessPieces	9.6%	-2.5%	6.4%	-6.8%	-19.3%	-13.3%
ClassroomVideo	15.3%	10.7%	14.4%	1.4%	3.4%	2.6%
Hijack	32.3%	9.9%	27.6%	19.8%	6.8%	12.9%
Museum	20.3%	7.5%	18.0%	10.5%	-0.3%	5.3%
CG-O: Average	**18.5%**	**6.0%**	**15.8%**	**5.5%**	**-3.2%**	**1.3%**
All: Average	**12.6%**	**8.6%**	**12.3%**	**3.7%**	**2.7%**	**4.1%**

Table 2. Bitrate reduction caused by the proposed patch occupation modification method; NC: natural content, CG-O: computer-generated omnidirectional video, CG-P: computer-generated perspective video.

In Table 3, the influence of the second proposed method – cluster splitting – is presented. The cluster splitting purpose is to decrease the pixelrate of the immersive video. However, in the MIV CTC [MPEG22], the pixelrates are explicitly set to the limit defined for HEVC Level 5.2: 1,069,547,520 luma samples.

Therefore, to be compliant with the MIV CTC, we did not change the atlas size (thus pixelrate), but we have calculated the total area occupied by patches. This approach allows to estimate the possible pixelrate reduction without modifying the CTC.

As presented in Table 3, proposed cluster splitting allows to significantly reduce the total occupied area of the second atlas. The first atlas is practically unchanged, as it contains mostly the base views.

Test sequence	Occupied area in second atlas		
	No cluster splitting	Cluster splitting	Difference
Carpark	25.41%	25.62%	0.21%
Fencing	47.54%	42.61%	– 4.94%
Frog	15.63%	11.98%	– 3.65%
Hall	43.62%	41.66%	– 1.97%
Painter	20.98%	21.23%	0.25%
Street	8.99%	5.89%	– 3.10%
NC: Average	**27.03%**	**24.83%**	**– 2.20%**
Cadillac	96.47%	94.75%	– 1.72%
Fan	43.13%	38.65%	– 4.49%
Group	89.38%	79.44%	– 9.95%
Kitchen	39.87%	40.32%	0.45%
Mirror	92.60%	79.97%	– 12.63%
CG-P: Average	**72.29%**	**66.63%**	**– 5.67%**
Chess	97.70%	85.10%	– 12.60%
ChessPieces	98.66%	84.81%	– 13.85%
ClassroomVideo	21.62%	21.82%	0.20%
Hijack	93.27%	79.43%	– 13.84%
Museum	69.12%	40.26%	– 28.86%
CG-O: Average	**76.07%**	**62.28%**	– 13.79%
All: Average	**56.50%**	**49.60%**	**– 6.91%**

Table 3. Area occupied by patches in the second atlas with and without the proposed cluster splitting method.

The possible pixelrate for both approaches can be calculated as follows:

$$P\left[\frac{pix}{s}\right] = (1 + O) \cdot W_A \cdot H_A \cdot FPS \cdot 1.25$$

where: O is the occupied area percentage presented in Table 3, W_A and H_A are atlas width and height (defined in the MIV CTC [MPEG22]), FPS is the frame rate of the sequence (25 for Carpark, Fencing, Hall, and Street; 30 for other sequences). Multiplier 1.25 allows to include both texture and geometry atlas (1 for texture with full resolution and 0.25 for depth atlas, decimated by 2 in both directions [MPEG21]).

Figs. 17 and 18 present the influence of both proposed methods, in terms of both bitrate and pixelrate.

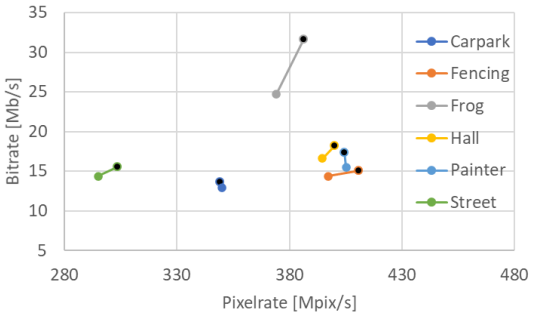


Figure 17. Bitrate vs. pixelrate for natural content; black dot: without proposed methods, color dot: with proposed modifications.

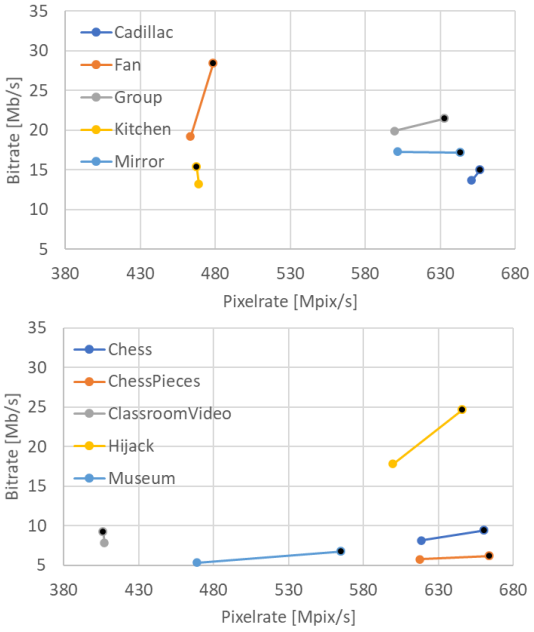


Figure 18. Bitrate vs. pixelrate for computer-generated sequences; black dot: without proposed methods, color dot: with proposed modifications.

As presented, the combination of patch occupation modification and cluster splitting allows to significantly decrease bitrate and pixelrate, irrespectively to the type of content.

Moreover, both proposed methods do not affect the rendering quality, as they do not modify the non-pruned pixels, which are used for rendering the virtual view watched by user of the immersive video system.

Regarding the computational time, video encoding (i.e., VVC) is much faster than without proposed techniques, while time needed for MIV encoding (i.e., atlas creation) is not influenced (Table 4).

Test sequence	MIV encoding time change			VVC encoding time change		
	A	B	C	A	B	C
Carpark	101.63%	113.66%	118.18%	93.16%	96.77%	84.80%
Fencing	90.76%	131.41%	99.88%	96.77%	95.49%	101.99%
Frog	111.34%	98.40%	116.30%	77.28%	106.93%	74.89%
Hall	86.38%	86.51%	88.32%	87.95%	98.06%	83.99%
Painter	116.85%	101.57%	107.15%	97.84%	91.31%	83.03%
Street	149.52%	101.37%	101.24%	92.34%	96.48%	84.43%
NC: Average	**109.41%**	**105.49%**	**105.18%**	**90.89%**	**97.51%**	**85.52%**
Cadillac	75.70%	75.40%	67.69%	84.10%	93.91%	85.65%
Fan	93.26%	139.61%	104.55%	74.90%	92.97%	76.62%
Group	98.46%	100.46%	105.20%	93.31%	90.58%	90.49%
Kitchen	99.52%	97.69%	117.75%	83.96%	97.52%	92.46%
Mirror	112.80%	113.05%	114.48%	111.43%	99.21%	97.06%
CG-P: Average	**95.95%**	**105.24%**	**101.93%**	**89.54%**	**94.84%**	**88.46%**
Chess	90.07%	86.19%	91.43%	96.11%	89.98%	90.54%
ChessPieces	97.00%	115.62%	102.29%	97.91%	92.34%	93.50%
ClassroomVideo	95.72%	104.38%	89.26%	80.23%	110.26%	89.30%
Hijack	85.47%	97.65%	94.85%	79.31%	90.07%	62.65%
Museum	100.37%	84.47%	101.29%	82.41%	89.66%	72.81%
CG-O: Average	**93.73%**	**97.66%**	**95.82%**	**87.19%**	**94.46%**	**81.76%**
All: Average	**100.30%**	**102.96%**	**101.24%**	**89.31%**	**95.72%**	**85.26%**

Table 4. Encoding time change (compared to the approach without proposed techniques); A: patch occupation modification, B: cluster splitting, C: both proposed techniques enabled.

6. CONCLUSIONS

The paper presents two techniques which allow to reduce the spatiotemporal redundancy of video within the MPEG Immersive video (MIV) encoder.

The first method is the patch occupation modification, which decreases the total bitrate of the immersive video encoded by MIV by decreasing the number of occupied pixels within texture and depth atlases.

The second method – cluster splitting allows to split large irregular clusters in order to decrease the total area of patches thus the pixelrate of the video.

Both ideas proposed by the authors of this paper were evaluated by experts of the ISO/IEC JTC1/SC29/WG 11 MPEG and are included in the Test Model for MPEG Immersive video (TMIV), which is the reference software implementation of the MIV.

7. ACKNOWLEDGMENTS

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00207, Immersive Media Research Laboratory).

8. REFERENCES

[Bro21] B. Bross et al. Overview of the Versatle Video Coding (VVC) standard and its applications. IEEE Tr. on Circ. and Syst. for Vid. Tech., 2021.

[Boi18] Boissonade P., and Jung J. Proposition of new sequences for Windowed-6DoF experiments on compression, synthesis, and depth estimation. Document ISO/IEC JTC1/SC29/WG11 MPEG/M43318, Ljubljana, Slovenia, Jul. 2018.

[Boy21] Boyce J., Doré R., Dziembowski A., Fleureau J., Jung J., Kroon B., Salahieh B., Vadakital V.K.M., and Yu L. MPEG Immersive Video Coding Standard. Proceedings of the IEEE, vol. 109, no. 9, pp. 1521-1536, Sep. 2021.

[Ceu18] B. Ceulemans et al. Robust Multiview Synthesis for Wide-Baseline Camera Arrays. IEEE Tr. on Multimedia, 2018.

[Cui19] L. Cui et al. Point-Cloud Compression: Moving Picture Experts Group's New Standard in 2020. IEEE Consumer Electronics Mag., 2019.

[Dor18] Doré R. Technicolor 3DoF+ Test Materials. Doc. ISO/IEC JTC1/SC29/WG11 MPEG/ M42349, San Diego, CA, USA, Apr. 2018.

[Dor20a] Doré R. et al. InterdigitalFan0 content proposal for MIV. Doc. ISO/IEC JTC1/SC29/ WG04 MPEG VC/ M54732, Online, Jul. 2020.

[Dor20b] Doré R. et al. InterdigitalGroup content proposal for MIV. Doc. ISO/IEC JTC1/SC29/ WG04 MPEG VC/ M54731, Online, Jul. 2020.

[Dor21a] Doré R. et al. Interdigital Mirror Content Proposal for advanced MIV investigations on reflection. Doc. ISO/IEC JTC1/SC29/WG04 MPEG VC/ M55710, Online, Jul. 2021.

[Dor21b] Doré R. et al. New Cadillac content proposal for advanced MIV v2 investigations. Doc. ISO/IEC JTC1/SC29/WG04 MPEG VC/ M57186, Online, Jan. 2021.

[Dom16] Domański M. et al. Multiview test video sequences for free navigation exploration obtained using pairs of cameras. Doc. ISO/IEC JTC1/SC29/WG11, MPEG M38247, 2016.

[Doy18] Doyen D. et al. [MPEG-I Visual] New Version of the Pseudo-Rectified Technicolor painter Content. Doc. ISO/IEC JTC1/SC29/ WG11 MPEG/M43366, Ljublana, 2018.

[Dzi20a] Dziembowski A. et al. Immersive Video CE3.1: Patch splitting. Doc. ISO/IEC JTC1/SC29/ WG11 MPEG/M51602, Brussels, Jan. 2020.

[Dzi20b] Dziembowski A. et al. Immersive Video CE3.2: Temporal patch redundancy removal. Doc. ISO/IEC JTC1/SC29/WG11 MPEG/M51603, Brussels, Belgium, Jan. 2020.

[Fac18] Fachada S. et al. Depth image based view synthesis with multiple reference views for virtual reality. 3DTV-Conf, Helsinki, Finland, Jun. 2018.

[Fuj06] Fujii T. et al. Multipoint measuring system for video and sound – 100-camera and microphone system, IEEE Int. Conf. on Mult. and Expo, 2006.

[Ilo19] Ilola L. et al. New test content for Immersive Video – Nokia Chess. Doc. ISO/IEC JTC1/SC29/ WG11 MPEG/M50787, Geneva, Oct. 2019.

[Ilo20] Ilola L. et al. Improved NokiaChess sequence. ISO/IEC JTC1/SC29/WG04 MPEG VC/ M54382, Online, Jul. 2020.

[Isg14] F. Isgro et al. Three-dimensional image processing in the future of immersive media. IEEE Tr. on Circuits and Systems for Video Tech., 2014.

[ISO22] Standard ISO/IEC FDIS 23090-12. Information technology – Coded representation of immersive media – Part 12: MPEG Immersive video. 2022.

[Kro18] Kroon B. Test sequence ClassroomVideo. Document ISO/IEC JTC1/SC29/WG11 MPEG/M42415, San Diego, CA, USA, Apr. 2018.

[Mie20a] Mieloch D., Stankiewicz O., and Domański M. Depth Map Estimation for Free-Viewpoint Television and Virtual Navigation. 2020 IEEE Access, vol. 8, pp. 5760-5776, 2020.

[Mie20b] Mieloch D. et al. [MPEG-I Visual] Natural Outdoor Test Sequences. Doc. ISO/IEC JTC1/ SC29/WG11 MPEG/M51598, Brussels, Jan. 2020.

[MPEG21] Test Model 11 for MPEG Immersive video. Document ISO/IEC JTC1/SC29/WG04 MPEG VC, N0142, Online, Oct. 2021.

[MPEG22] Common Test Conditions for MPEG Immersive video. Document ISO/IEC JTC1/SC29/ WG04 MPEG VC, N0169, Online, Jan. 2022.

[Mül11] Müller K. et al. 3-D Video Representation Using Depth Maps. 2011 Proceedings of the IEEE, vol. 99, no. 4, pp. 643-656, 2011.

[Nem10] O. Nemcic et al. Multiview Video Coding extension of the H.264/AVC standard. ELMAR, Zadar, Croatia, Sep. 2010.

[Sal18] Salahieh B. et al. Kermit test sequence for Windowed 6DoF Activities. Doc. ISO/IEC JTC1/ SC29/WG11 MPEG/M43748, Ljublana, Jul. 2018.

[Sta18] O. Stankiewicz et al. A free-viewpoint television system for horizontal virtual navigation. IEEE Tr. on Multimedia, 2018.

[Sul05] G. Sullivan and T. Wiegand. Video compression – from concepts to the H.264/AVC standard. Proceedings of the IEEE, vol. 93, 2005.

[Sul12] G. Sullivan et al. Overview of the High Efficiency Video Coding (HEVC) standard. IEEE Tr. on Circ. and Syst. for Vid. Tech., vol. 22, 2012.

[Tan12] Tanimoto M. et al. FTV for 3-D Spatial Communication. 2012 Proceedings of the IEEE, vol. 100, no. 4, pp. 905-917, 2012.

[Tao21] L. Tao et al. A Convenient and High-Accuracy Multicamera Calibration Method Based on Imperfect Spherical Objects. IEEE Tr. on Instrumentation and Measurement, vol. 70, 2021.

[Tec15] G. Tech et al. Overview of the Multiview and 3D Extensions of High Efficiency Video Coding. IEEE Tr. on Circ. and Syst. for Vid. Tech. 2016.

[Zha20] J. Zhang et al. Point Cloud Normal Estimation by Fast Guided Least Squares Representation. IEEE Access, 2020.

[Zhu19] S. Zhu et al. An improved depth image based virtual view synthesis method for interactive 3D video. IEEE Access, 2019.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Uncertainty-aware Evaluation of Machine Learning Performance in binary Classification Tasks

Leo Sperling
University of Kaiserslautern
Gottlieb-Daimler-Strasse 47
Germany, 67663
Kaiserslautern, RLP
sperling@rhrk.uni-kl.de

Simon Lämmer
Leipzig University
Ritterstr. 26
Germany, 04109
Leipzig, Saxony
simon.laemmer@outlook.de

Hans Hagen
University of Kaiserslautern
Gottlieb-Daimler-Strasse 47
Germany, 67663
Kaiserslautern, RLP
hagen@cs.uni-kl.de

Gerik Scheuermann
Leipzig University
Ritterstr. 26
Germany, 04109
Leipzig, Saxony
scheuermann@informatik.uni-leipzig.de

Christina Gillmann
Leipzig University
Ritterstr. 26
Germany, 04109
Leipzig, Saxony
gillmann@informatik.uni-leipzig.de

ABSTRACT

Machine learning has become a standard tool in computer vision. Nowadays, neural networks are one of the most prominent representatives in this class of algorithms that usually require training and evaluation to work as desired. There exist a variety of evaluation metrics to determine the quality of a trained neural network, which are usually threshold dependent. This results in massive changes in the resulting evaluation when the threshold is changed slightly. Further, measurements of uncertainty such as resulting from Bayesian approaches, are not considered in this analysis. In this paper, we present evaluation metrics for machine learning approaches that are able to attach a probability distribution to the utilized threshold and include uncertainty measures. We demonstrate the applicability of our approach by applying the defined metrics to a real-world example where a Bayesian neural network has been used to predict stroke lesions.

Keywords

Evaluation Measures, Uncertainty-awareness, Machine Learning

1 INTRODUCTION

Machine learning approaches become increasingly important in the area of computer vision [1]. Especially in classification tasks, machine learning approaches have developed into a standard tool, massively reshaping the respective area. In this process, the evaluation of machine learning approaches is a crucial factor. Here, a variety of measures exist that aim to examine the performance using different assumptions and focus points.

As input data, models, and the use of visualization usually include uncertainty [2], the evaluation of machine learning approaches can be affected. Sacha et al. [3] proposed, that uncertainty has a crucial impact on the decision-making process. Unfortunately, the existing measures do not include uncertainty in their computation.

Evaluation measures such as DICE-coefficient (=F1 score) and accuracy for machine learning approaches, usually do not consider the uncertainty inherent in the machine learning process. Instead, they consider a pre-selected threshold and build their computation based on true positive (TP), true negative (TN), false positive (FP) and false negative (FN). Unfortunately, the selection of this threshold holds a large potential of uncertainty. Slight changes in the choice of the threshold can have a massive impact on the resulting evaluation.

In addition, fuzzy machine learning approaches, such as Bayesian Neural Networks [4], output a measure of uncertainty in addition to the classification prediction, which is usually not considered in the evaluation of machine learning approaches. This results in an evaluation that is equally balanced along all classifications made in

a machine learning model, independent of how certain the prediction is (see Section 2).

In this work, we aim to revisit popular evaluation measures for machine learning approaches that target binary classification tasks (see Section 3). To achieve this, we first rephrase the terms TP, TN, FP, FN, such that we use an uncertainty-aware threshold. Based on this, we can rebuild popular machine learning evaluation measures to include the uncertainty-aware threshold. Further, we include a damping factor that allows adjusting the importance of predicted classifications based on measured uncertainty.

Therefore, this paper contributes:

- Uncertainty-aware classification of machine learning results
- A mechanism to include uncertain classification results in machine learning evaluation
- Uncertainty-aware evaluation measures for machine learning approaches

We show how the defined uncertainty-aware measures can be used in machine learning performance evaluation using varying examples as shown in Section 4. Our results will be discussed in Section 5.

2 RELATED WORK

In the context of the presented approach, we aim to analyze previous work conducted in the area of uncertainty-aware machine learning and the evaluation of these approaches.

2.1 Uncertainty-aware Machine Learning

The importance of uncertainty analysis in the area of machine learning has been highlighted by Klaes et al. [5]. In their work, they summarize potential sources of uncertainty in the respective area. The presented taxonomy holds a valuable starting point in the presented area of research. This approach was also refined for machine learning approaches in medical imaging [6]. The described sources of uncertainty are manifold and therefore various approaches exist that aim to target one or multiple sources of uncertainty.

Sluijterman et al. [7] provided an adapted approach of regression, that aims to include uncertainty quantification during the computation. Nieradzik et al. [8] exchanged the output activation function which is usually set to the sigmoid function with further functions and examined their suitability regarding the resulting prediction and their uncertainty. Ding et al. [9] proposed an uncertainty-aware training, where training data is adapted such that more reliable data points become more important in the training process. Eldesokey et al. [10] aimed for a holistic uncertainty-aware machine learning approach that includes multiple sources of uncertainty. Here, uncertainty arising from the data as well as the uncertainty of the model is included throughout the entire computation of the machine learning approach. Although these approaches all target the incorporation of uncertainty into the training process, they rely on the classic evaluation approaches for machine learning approaches, which are threshold-based. In this work, we aim to extend these approaches such that the threshold holds a probability distribution function to indicate its potential uncertainty.

Recently, the number of machine learning approaches that explicitly work with mathematical concepts that directly include uncertainty increased. Here, approaches such as fuzzy deep networks [11] or Bayesian neural networks [4] that can output epistemic and aleatoric uncertainty [12] in their prediction have been developed. Epistemic uncertainty refers to uncertainty inherent in a model, as models are always making assumptions. On the other hand, aleatoric uncertainty refers to uncertainty inherent in captured data due to random effects and measurement imprecision.

Also, Sacco et al. [13] proposed a neural network approach that builds a second neural network to predict the uncertainty inherent in the computational process. All these approaches are able to attach an uncertainty to the made prediction. Still, these values are usually only reviewed visually but are not considered in the evaluation of the proposed approach. In this work, we aim to provide a mechanism to include this knowledge.

2.2 Uncertainty-aware evaluation of Machine Learning

The evaluation of machine learning approaches is a key point while using them. There exist a variety of surveys and books that summarize and categorize them [14, 15]. These measures include DICE-coefficient, accuracy, recall, and precision and are used for benchmarking [16]. Their selection is dependent on the underlying problem and type of used machine learning approach [17]. All these measures are based on the separation of predicted values into TP, TN, FP, and FN. Here, a threshold is selected to achieve this separation. The choice of this threshold can have a massive influence on the evaluation of the machine learning approach and needs to be adjusted in each case.

Gao et al. [18], presented an approach that aims to generate a self-adapting threshold for the evaluation of a neural network. The method is built on an analysis of the imbalance of classes that are predicted. Thada et al. [19] adapted the threshold for evaluation based on the underlying scale of predicted classifications. Here, different scales obtain different thresholds. Li et al. [20] provided a machine learning approach that aims

to guess a proper threshold based on the underlying dataset. Here, different thresholds are examined to understand the resulting classification. Although these approaches aim to select the threshold that is used for evaluation, the choice may still remain uncertain. Therefore, our approach aims to add a probability distribution function to the selected threshold to express the uncertainty in this decision.

Taha et al. [21] presented a set of evaluation metrics that are based on fuzzy theory. Here, the prediction and the ground truths are considered as fuzzy sets, and metrics are presented that compare them. Although this gives a valuable starting point for the presented work, the approach is not able to indicate an uncertainty-aware threshold. In addition, the inclusion of uncertainty that can result from a neural network cannot be included in this approach.

Psaros et al. [22] provided evaluation metrics that aim to include the uncertainty that can be outputted by machine learning approaches. Here, prominent metrics are adapted individually to include uncertainty information. Still, this approach is based on a fixed threshold. In the presented approach we aim to present a generalized way to include an uncertainty-aware threshold as well as a damping factor that adjusts made classifications based on the underlying uncertainty.

3 METHODS

To achieve uncertainty-aware evaluation measures for machine learning, we first aim to extend prominent measures of neural network performance to include a probability distribution to the user-defined threshold. Based on this, we will further include potential uncertainty measures that can be outputted by Bayesian Network approaches.

3.1 Uncertainty-aware classification

Neural Networks aim to learn from existing datasets. To test the performance of the neural network, the predicted results are compared to a ground truth. In general, the closer both are to each other, the better the performance. Here, each datapoint i, and its classification $c(i)$ is compared to the prediction $p(i)$. Note that we restrict the range of $c(i)$ to 1 and 0, while the range of $p(i)$ is the interval $[0,1]$, as most machine learning approaches output probabilities instead of fixed class assignments.

Most evaluation measures for neural networks work based on a classification of values into TP, FP, TN, FN, which are based on a pre-selected threshold t. Therefore, the following definitions are known:

$$TP_i(t) = \begin{cases} \overbrace{1}^{A} & \overbrace{c(i)}^{B} \wedge \overbrace{[p(i) \geq t]}^{C} \\ 0 & else \end{cases} \tag{1}$$

$$TN_i(t) = \begin{cases} 1 & !c(i) \wedge [p(i) \leq t] \\ 0 & else \end{cases} \tag{2}$$

$$FP_i(t) = \begin{cases} 1 & !c(i) \wedge [p(i) > t] \\ 0 & else \end{cases} \tag{3}$$

$$FN_i(t) = \begin{cases} 1 & c(i) \wedge [p(i) < t] \\ 0 & else \end{cases} \tag{4}$$

with respect to the threshold t. We define subequations for future reference in this manuscript to allow easy to follow changes that we make. A is defined as the function output of the classification functions. B represents the classification that was made by a neural network and C represents the groundtruth that is is used for the comparison.

By definition, the result of these functions can only be 0 or 1. A is either 0 or 1 in a fixed case. In our approach, we also consider uncertain ground truths. Although most of the available training databases provide a fixed classification, the number of ground truths that hold fuzzy values increases. Therefore, we redefine $c(i)$ and allow it to lie in the range of $[0,1]$. Now, we also have to adjust the decision to which class a point belongs. Here, $B = [c(i) > t]$ holds. Still, we need to find a mechanism that allows rating the certainty of this decision.

Considering that we no longer work with fixed values of 1 and 0, we need to make an adaptation. We aim to define a general way to compare values to a threshold that has a probability distribution attached.

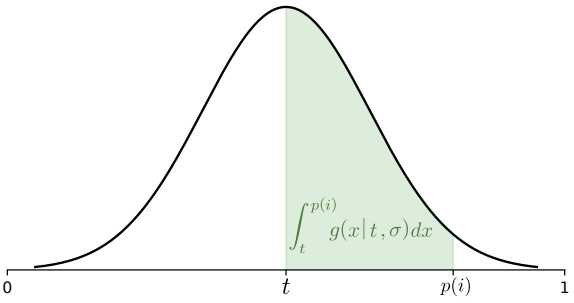


Figure 1: Schematic description of incorporation of Gaussian distribution of the threshold t.

As mentioned, the decision based on a fixed threshold results in fixed classifications. The decision to choose a threshold can be very hard as it is usually dependent on the underlying application. In addition, slight changes in the choice of the threshold can have a massive influence on the quality measures. Here, we use a Gaussian distribution function that is normalized:

$$g(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5}$$

A normalized Gaussian distribution function has beneficial attributes in the presented case. As the area under the curve is always 1, we can use this measure to adapt our previous classifications. Here, σ decides how sharp or flat the resulting Gaussian distribution is. We allow users to set the standard deviation σ in conjunction with the threshold t.

Here, a probabilistic measure if a threshold is exceeded can be expressed as:

$$\overline{A_{x_1}} = 2\int_{x_1}^{t} g(x|t,\sigma)dx \tag{6}$$

As the peak of the Gaussian distribution function is located at the threshold t, the maximum size of the area under the curve can be 0.5. As we aim for a measure that is located in the range of [0,1], we need to double this area. If t and x_1 are located close to each other, the resulting area under the curve converges to 0. A close location means a high uncertainty, which means that under this condition we aim to tone down the result of the classification. On the other hand, if the points are not close to each other, the area under the curve converges to 1. This results in low uncertainty. Resulting from this consideration, the classification scheme A can be rephrased as:

$$TP_i(t,\sigma)\begin{cases}\overline{A_{p(i)}}\cdot\overline{A_{c(i)}} & [c(i)>t]\wedge[p(i)>t]\\ 0 & else\end{cases} \tag{7}$$

The values of A computed in the measures $TN_i(t,\sigma)$, $FP_i(t,\sigma)$ and $FN_i(t,\sigma)$ are computed like this as well. Based on these extended definitions of TP, TN, FP, and FN, we further aim to include uncertainty that is captured in predictions by machine learning approaches.

3.2 Inclusion of predicted uncertainty

Recently, a variety of machine learning approaches is able to output uncertainty measures related to the made prediction. Here, especially Bayesian neural networks are able to output aleatoric as well as epistemic uncertainty measures. In this work, we aim to include these measures into the classifications. Here, we aim to achieve a weighting of the classification according to the outputted uncertainty measures. In particular, we aim for a classification scheme, that extends the existing scheme in the following manner $TP_i(t,\sigma,d) = TP_i(t,\sigma)\cdot C_d$, where C_d is supposed to work as a damping factor.

The goal of this factor is to tone down prediction values that are considered uncertain in the measures that can be outputted by uncertainty-aware machine learning approaches. We consider $u(i)$ as the uncertainty attached to the prediction value $p(i)$. Here, the uncertainty can be located in the range of $[0,\infty)$.

To define C_d, we aim for a function that outputs 1, if the uncertainty predicted by a machine learning approach is 0. In this case, the made classification will remain the same. In contrast, if a data point is classified as uncertain, we aim to let the damping function converge to 0. Here, we utilize the function:

$$C_d = e^{-(u(i)\cdot d)}, \tag{8}$$

where d works as an additional damping factor, that can be located in the range $[0,\infty)$.

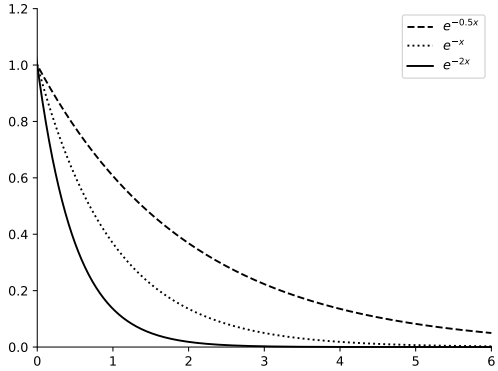


Figure 2: Different damping functions, based on the damping factor d. Examples are shown for $d = \frac{1}{2}$, $d = 1$, $d = 2$.

To indicate the effect of d, Figure 2 shows examples of interesting classes of the damping factor. The higher the damping factor, the more pronounced is the influence of the damping on the result. This gives the user a further input parameter with which to control the damping of the classification based on uncertainty quantification of the predictions made by machine learning approaches. If this quantification does not exist or cannot be achieved, C_d can be set to 1 and therefore does not change the made classifications.

Based on the made extensions of the classification schemes, we can extend the formulas from equations 1, 2, 4 and 3 using the scheme as explained:

$$TP_i(t,\sigma,d) = \begin{cases}TP_i(t,\sigma)\cdot C_d & [c(i)>t]\wedge[p(i)\geq t]\\ 0 & else\end{cases} \tag{9}$$

$$TN_i(t,\sigma,d) = \begin{cases}TN_i(t,\sigma)\cdot C_d & [c(i)\leq t]\wedge[p(i)\leq t]\\ 0 & else\end{cases} \tag{10}$$

$$FP_i(t,\sigma,d) = \begin{cases}FP_i(t,\sigma)\cdot C_d & [c(i)\leq t]\wedge[p(i)>t]\\ 0 & else\end{cases} \tag{11}$$

$$FN_i(t,\sigma,d) = \begin{cases} FN_i(t,\sigma)\cdot C_d & [c(i) > t] \wedge [p(i) < t] \\ 0 & else \end{cases} \tag{12}$$

Based on these definitions, we are able to extend well-known evaluation metrics for machine learning approaches.

3.3 Uncertainty-aware evaluation measures

In the following, we will summarize potential evaluation measures that are based on the prior classifications. The measures have been chosen as they are popular choices for machine learning evaluation [23]. Here, we need to sum all values that will be outputted when considering all n datapoints. Therefore we define $TP(t,\sigma,d) := \Sigma_{i=0}^{n} TP_i(t,\sigma,d)$. Respectively, $TN(t,\sigma,d)$, $FP(t,\sigma,d)$ and $FN(t,\sigma,d)$ can be defined.

$$\overline{Accuracy} := \frac{TP(t,\sigma,d)+TN(t,\sigma,d)}{TP(t,\sigma,d)+TN(t,\sigma,d)+FP(t,\sigma,d)+FN(t,\sigma,d)} \tag{13}$$

$$\overline{Precision} := \frac{TP(t,\sigma,d)}{TP(t,\sigma,d)+FP(t,\sigma,d)} \tag{14}$$

$$\overline{Recall} := \frac{TP(t,\sigma,d)}{TP(t,\sigma,d)+FN(t,\sigma,d)} \tag{15}$$

$$\overline{FalsePositiveRate} := \frac{FP(t,\sigma,d)}{FP(t,\sigma,d)+TN(t,\sigma,d)} \tag{16}$$

$$\overline{F1} := \frac{2\cdot\overline{Precision}\cdot\overline{Recall}}{\overline{Precision}+\overline{Recall}} \tag{17}$$

4 CASE STUDY

In this section, we aim to apply the developed uncertainty-aware evaluation metrics to a trained Bayesian U-Net (BNN) [24] for stroke lesion prediction [25]. We aim to show how the defined metrics can be used and how the defined parameters influence the computation.

4.1 Use Case Description

The provided BNN generates lesion maps from stroke patients that predict their final formation. Here, a multimodal input is used to predict a lesion map that can be found in the work of Gillmann et al. [26]. In addition, it predicts voxel-wise epistemic and heteroscedastic aleatoric uncertainty alongside [27]. The epistemic uncertainty stems from Monte Carlo dropout and is a property of the model used to describe the real-world process. It expresses not knowing exactly, which model generated the data in the real world. The heteroscedastic aleatoric uncertainty was trained as an unsupervised parameter in the loss function. We will use this model to demonstrate the applicability of the presented approach.

(a) (b) (c) (d) (e)

Figure 3: Cross section of lesion map prediction (a) and epistemic uncertainties (b) from BNN, thresholded at $t = 0.8$ (c) and compared to the ground truth (d). (e) shows the groundtruth (in red) overlayed on top of the CT Angiography, which is one of the inputs to the BNN.

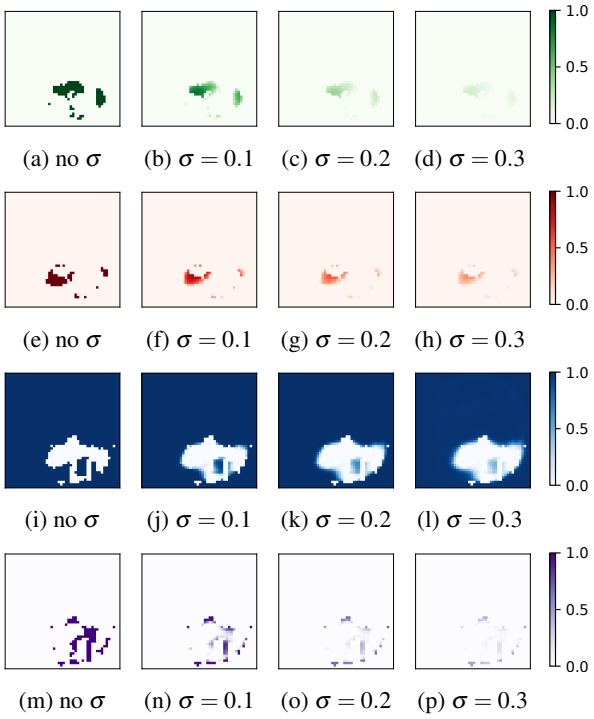
In this use case, we consider a particular cross-section of the 3D volume of a patient. Figure 3(a) shows the prediction made by the BNN, whereas 3(b) shows the predicted epistemic uncertainty. The prediction holds values between 0 (no lesion predicted) and 1 (lesion predicted). The ground truth that was labeled by medical experts is shown in Figure 3(d). Usually, performance measures are computed based on the thresholded prediction (Fig. 3(c)) and the pre-labeled ground truth. In this case, the groundtruth was created by medical experts that reviewed each patient individually and marked areas in the image that show a stroke lesion. For this example we show how the presented uncertainty-aware measures can be applied.

4.2 Results

In the following we aim to discuss the influence of the user-selected values σ and d to the classification values as well as the resulting metrics that can be computed based on these classifications. We define a consistent colorscheme for the four classes, i.e. TP (green), FP (red), TN (blue) and FN (purple).

Influence of σ to classifications As mentioned, the user-defined variability of the selected threshold shapes the sharpness of the classification result. The resulting classification of the presented prediction of stroke lesion into TP, FP, TN and FN with varying σ can be seen in Figure 4. 4(a), 4(e), 4(i) and 4(m) show the original computation of the classification. Here, clear boundaries can be identified due to the strict separation of classifiers. This coincides with an application of our evaluation approach when $\sigma \to 0$. Therefore, the presented measures extend the existing ones.

When increasing σ, the crisp boundaries of the classifications vanish and the separation into the classes is



(a) no σ (b) $\sigma = 0.1$ (c) $\sigma = 0.2$ (d) $\sigma = 0.3$

(e) no σ (f) $\sigma = 0.1$ (g) $\sigma = 0.2$ (h) $\sigma = 0.3$

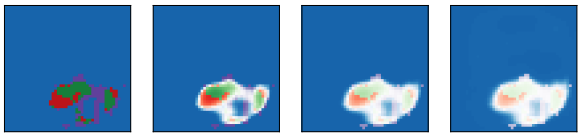
(i) no σ (j) $\sigma = 0.1$ (k) $\sigma = 0.2$ (l) $\sigma = 0.3$

(m) no σ (n) $\sigma = 0.1$ (o) $\sigma = 0.2$ (p) $\sigma = 0.3$

Figure 4: Classification of TP, FP, TN and FN (rows) with varying σ (columns). With larger σ, the score for classifications decreases according to the closeness to the threshold.

less clear. This matches with the intuition that a high σ represents a large uncertainty of the selected threshold.

Figure 5 shows the merged visualization of the made classification with varying σ. Figure 5(a) show the strict separation of the made prediction into the four classes. For increasing σ, an area with uncertain values is visible that indicates values of the prediction that are close to the threshold but would be considered as certain as the other predictions if the threshold is set fixed.



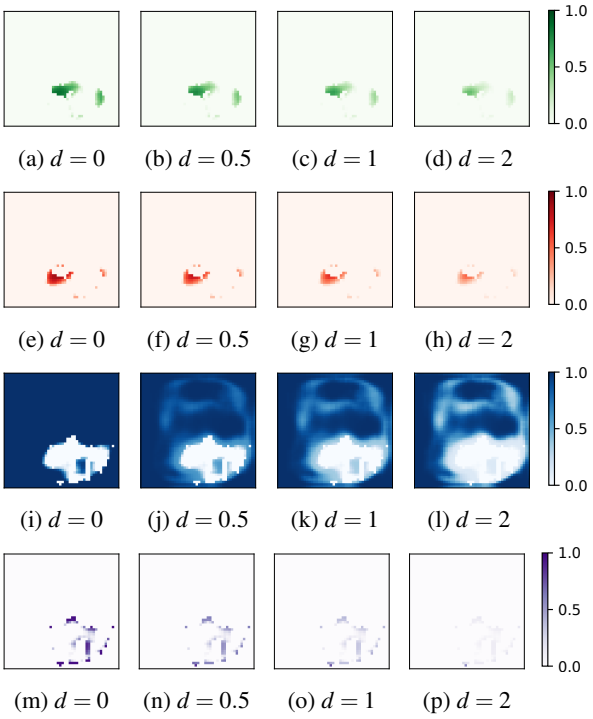
(a) no σ (b) $\sigma = 0.1$ (c) $\sigma = 0.2$ (d) $\sigma = 0.3$

Figure 5: Classification of TP, FP, TN and FN with different values for σ and fixed $d = 0$.

Influence of d The damping factor d has a large influence where the uncertainty of the BNN is high.

This effect to the classification metrics can be seen in Figure 6. Here, the value of σ is set to 0.1 in all cases. d is altered with $0, 0.5, 1$ and 2.

The damping factor controls the influence of the uncertainty on the made classifications. With increasing d, values that contain a high uncertainty will result in



(a) $d = 0$ (b) $d = 0.5$ (c) $d = 1$ (d) $d = 2$

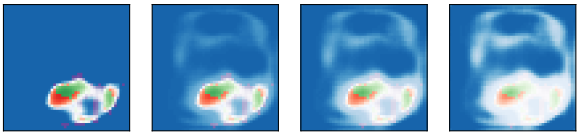
(e) $d = 0$ (f) $d = 0.5$ (g) $d = 1$ (h) $d = 2$

(i) $d = 0$ (j) $d = 0.5$ (k) $d = 1$ (l) $d = 2$

(m) $d = 0$ (n) $d = 0.5$ (o) $d = 1$ (p) $d = 2$

Figure 6: Classification of TP, FP, TN and FN (rows) for a fixed $\sigma = 0.1$ and varying d (columns).

a less strong classification. This effect can be seen very clearly when considering Figure 6(i) 6(j), 6(k) and 6(l). When setting d to 0, the result is almost binary. While increasing d, values with a high uncertainty get a lower classification score. When comparing the result of Figure 6(l) with Figure 3(b) we can identify the large influence of uncertain values in the prediction. Here, the uncertainty results in areas that cannot be separated clearly. Further, areas that do not contain a high uncertainty will not be affected by the application of the damping factor.



(a) $d = 0$ (b) $d = 0.5$ (c) $d = 1$ (d) $d = 2$

Figure 7: Classification of TP, FP, TN and FN with fixed $\sigma = 0.1$ and varying d.

The effect of varying the damping factor d on the combined image of the classifications into TP, FP, TN and FN is shown in Figure 7. Here, we can identify that a high uncertainty lowers the overall classification score of datapoints. When increasing d, uncertain areas will result in unclassified data values.

Influence on evaluation metrics Based on the made classifications, we adapted prominent examples of evaluation metrics.

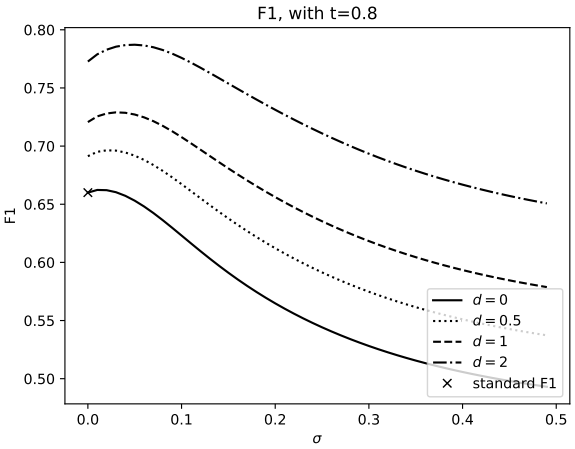


Figure 8: Results of adapted F1 metric. By incorporating the epistemic uncertainty information we get higher scores from the metric. The $\overline{F1}$ score increases for small σ, but for large σ we get lower scores.

Figure 8 shows the results of the adapted F1 metric (also known as DICE-coefficient) with a threshold of $t = 0.8$, variable σ and selected values for the damping factor d (0, 0.5, 1 and 2).

The unmodified F1 score is also highlighted in the graph (indicated with a black x). Incorporating the epistemic uncertainty into the classification by using a damping factor improved the final F1 score significantly while increasing the σ reduced the score overall. This results from the fact, that an increased σ removes confidence in the classifications and therefore lowers the result of the measurement output.

On the other hand, the damping factor removes uncertain values from the computation. Usually, these values are located around the boundary of areas in the ground truth which turn out to be classified wrongly in many cases. The damping factor removes these areas and therefore increases the overall performance result. At this point, we want to highlight that this effect might be reversed when uncertain areas are located within correctly predicted regions.

Interestingly, the best output of the evaluation metrics can be achieved with a sigma slightly lower than 0.1 and a damping factor of 2. In the given case this means that a consideration of an uncertainty-aware threshold leads to a better rating of the network performance. This fits with the intention of this work which aims to remove the fixed thresholding.

We also applied our adapted measures to further evaluation measures as shown in Figure 9. Here, we examined the measures $\overline{Accuracy}$, $\overline{Precision}$, $\overline{Recall}$ and $\overline{FPR}$.

When considering $\overline{Accuracy}$ (Figure 9(a)), we can identify that the unmodified accuracy metric shows a good result for the network (0.97). In the presented case, this is not surprising as the network predicts a high amount of TN correctly. Increasing σ results in even better ratings for the network as uncertain classifications are weighted less than certain classifications. In addition, an increased d further improves the network performance.

Figure 9(b) shows the results of the measure $\overline{Precision}$, when varying σ and d. Here, we can observe that the best choice of σ in the presented case is 0.1. Interestingly a further increase of σ leads to a dramatic loss in precision. This matches with the observation that can be made in Figure 4(d) and 4(h). Increasing σ results in a slow vanishing of FP and a faster vanishing of TP. Resulting from this, the output of precision decreases as well. Overall the effect of d is low in the considered case.

A similar effect can be seen when considering $\overline{Recall}$. Again the best results are achieved when using σ at around 0.1. The measure is computed using TP and TN. In Figure 4(l), we can observe that increasing σ results in less vanished values for TN. Therefore, this effects the $\overline{Recall}$ metric similarly to the $\overline{Precision}$ metric. In contrast to precision, for recall, the effect of d is high in the given case.

The effect of σ and d for $\overline{FPR}$ can be seen in Figure 9(d). When increasing σ, the result improves. This also holds for an increased d.

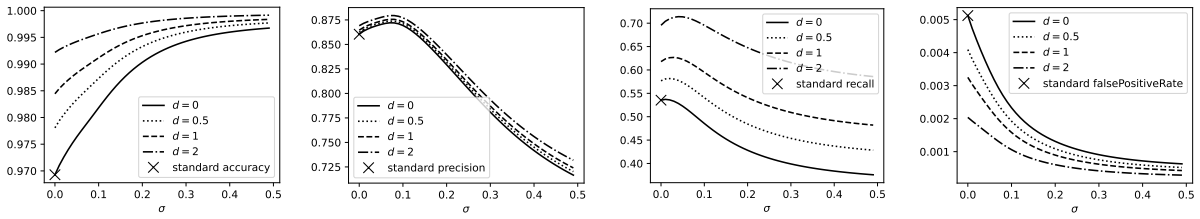
5 DISCUSSION

General Observations The presented metrics allow generalizing original machine learning performance metrics. When setting σ and d to 0, the resulting values are equal to the original computations.

The classifications that we proposed are based on a Gaussian distribution function but can be exchanged with any distribution function that holds an overall integral of 1. Also, the used damping function could be adapted if required. Here, functions that output 1, when a damping factor of 0 is used can be considered.

By using the adapted definitions of TP, FP, TN, and FN with a $\sigma > 0$ we can basically encode how far away the predictions are from the threshold. The benefit of this can be seen in Figure 5, wherewith increasing σ one can easily assess the quality of the threshold. For this particular example, it seems like the threshold is well chosen to classify the TN while keeping FN to a minimum. With increasing σ the TN stays the same, except at the boundaries, while the FN quickly fades away, which means that they are close to the threshold – they are classified with high uncertainty. The classifications of TP and FP also fade away relatively quickly. They are also relatively close to the threshold, and thus also relatively uncertain.

Using the damping factor allows to include the uncertainty captured in the made prediction of a machine



(a) $\overline{Accuracy}$ (b) $\overline{Precision}$ (c) $\overline{Recall}$ (d) $\overline{FPR}$

Figure 9: Results of the adapted metrics. Accuracy, Precision, Recall and FPR are affected by the choice of σ and d. The unmodified values of these metrics are indicated by a black x.

learning approach into the classification scheme. In Figure 7 it can be clearly seen that the classification around the general area of the lesion in the ground truth data is very uncertain. The same effect happens at the boundaries of the brain. Interestingly, the BNN predicts with high certainty FP. This is of course not desirable and such errors can be easily spotted with the method presented in this paper.

As a rule of thumb, it holds, that if for high σ and d the classifications of TP and TN are high and for the FP and negatives it is low, the model is trustworthy. This is also reflected in the adapted metrics, for example in the adapted accuracy metric depicted in Figure 9(a). The values are monotonically increasing with increasing σ and overall higher with higher d. One could infer, that our model is generally trustworthy where the uncertainty is low. If the graph in Figure 9(a) was monotonically decreasing, it would mean that the model predicts with high certainty wrong results, i.e. it is not that trustworthy. Care has to be taken for very unbalanced datasets, or datasets where one class can be much more easily identified than the other. This is the case for our model because a brain lesion can only occur in the brain, therefore a significant portion of the head scan can be easily classified as a TN with very high certainty. These problems can be alleviated by also considering the other metrics, like the F1-score.

Limitations Although the presented approach provides large flexibility, it also results in more input parameters. In this work, we showed that the influence of the input parameters can be inspected visually. Here, contrary to the original measurements, a visual inspection of the parameters is required.

In the presented work we showed that the provided measures are applicable for a BNN. We do not see limitations in the application to further networks, but we have not proven this statement.

6 CONCLUSION

This paper introduced adaptations to existing metrics for evaluating a binary classifier, that can incorporate uncertainty information from the model itself and uncertainty regarding the exact location of the threshold. For that, we use a Gaussian distribution function attached to the threshold and allow a damping factor for uncertainty-aware machine learning outputs. These metrics were applied to a real-world example of a Bayesian neural network to prove applicability.

As future work, we aim to use the measures in the back-propagation in the learning phase of neural networks. In addition, we further research the visual inspection of the chosen parameters of the presented measures.

7 REFERENCES

[1] A. I. Khan and S. Al-Habsi, "Machine learning in computer vision," *Procedia Computer Science*, vol. 167, pp. 1444–1451, 2020. International Conference on Computational Intelligence and Data Science.

[2] R. G. C. Maack, G. Scheuermann, H. Hagen, J. T. H. Penaloza, and C. Gillmann, "Uncertainty-aware Visual Analytics-Scope, Oppertunities and Challenges," PREPRINT (Version 1) available at Research Square, 2021.

[3] D. Sacha, H. Senaratne, B. C. Kwon, G. Ellis, and D. A. Keim, "The role of uncertainty, awareness, and trust in visual analytics," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 1, pp. 240–249, 2015.

[4] E. Goan and C. Fookes, "Bayesian neural networks: An introduction and survey," *Lecture Notes in Mathematics*, p. 45–87, 2020.

[5] M. Kläs and A. M. Vollmer, *Uncertainty in Machine Learning Applications: A Practice-Driven Classification of Uncertainty: SAFECOMP 2018 Workshops, ASSURE, DECSoS, SASSUR, STRIVE, and WAISE, Västerås, Sweden, September 18, 2018, Proceedings*, pp. 431–438. 01 2018.

[6] C. Gillmann, D. Saur, and G. Scheuermann, "How to deal with uncertainty in machine learning for medical imaging?," in *2021 IEEE Workshop on TRust and EXpertise in Visual Analytics (TREX)*, pp. 52–58, 2021.

[7] L. Sluijterman, E. Cator, and T. Heskes, "How to evaluate uncertainty estimates in machine learning for regression?," 2021.

[8] L. Nieradzik, G. Scheuermann, D. Saur, and C. Gillmann, "Effect of the output activation function on the probabilities and errors in medical image segmentation," 2021.

[9] Y. Ding, J. Liu, X. Xu, M. Huang, J. Zhuang, J. Xiong, and Y. Shi, "Uncertainty-aware training of neural networks for selective medical image segmentation," in *Proceedings of the Third Conference on Medical Imaging with Deep Learning* (T. Arbel, I. Ben Ayed, M. de Bruijne, M. Descoteaux, H. Lombaert, and C. Pal, eds.), vol. 121 of *Proceedings of Machine Learning Research*, pp. 156–173, PMLR, 06–08 Jul 2020.

[10] A. Eldesokey, M. Felsberg, K. Holmquist, and M. Persson, "Uncertainty-aware cnns for depth completion: Uncertainty from beginning to end," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12011–12020, 2020.

[11] R. Das, S. Sen, and U. Maulik, "A survey on fuzzy deep neural networks," *ACM Comput. Surv.*, vol. 53, may 2020.

[12] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods," *Machine Learning*, vol. 110, p. 457–506, Mar 2021.

[13] M. Sacco, J. Ruiz, M. Pulido, and P. Tandeo, "Evaluation of machine learning techniques for forecast uncertainty quantification," 11 2021.

[14] J. M. Twomey and A. E. Smith, "Performance measures, consistency, and power for artificial neural network models," *Mathematical and Computer Modelling*, vol. 21, pp. 243–258, 1995.

[15] O. Schoppe, N. S. Harper, B. D. B. Willmore, A. J. King, and J. W. H. Schnupp, "Measuring the performance of neural models," *Frontiers in Computational Neuroscience*, vol. 10, 2016.

[16] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark analysis of representative deep neural network architectures," 10 2018.

[17] W. Imtiaz, H. Ghafoor, R. Sehar, T. Mahboob, and M. Khanam, "Evaluating the performance estimators via machine learning supervised learning algorithms for dataset threshold," *International Journal of Computer Applications*, vol. 119, pp. 1–6, 06 2015.

[18] S. Gao, W. Dong, K. Cheng, X. Yang, S. Zheng, and H. Yu, "Adaptive decision threshold-based extreme learning machine for classifying imbalanced multi-label data," *Neural Processing Letters*, vol. 52, pp. 1–23, 12 2020.

[19] V. Thada and V. Jaglan, "Comparison of jaccard, dice, cosine similarity coefficient to find best fitness value for web retrieved documents using genetic algorithm," *International Journal of Innovations in Engineering and Technology*, vol. 2, pp. 202–205, 08 2013.

[20] S. Li, Y. Xie, and L. Song, "Data-driven threshold machine: Scan statistics, change-point detection, and extreme bandits," 2016.

[21] A. A. Taha and A. Hanbury, "Metrics for evaluating 3d medical image segmentation: Analysis, selection, and tool," *BMC Medical Imaging*, vol. 15, 08 2015.

[22] A. F. Psaros, X. Meng, Z. Zou, L. Guo, and G. E. Karniadakis, "Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons," 2022.

[23] D. Conway and J. White, *Machine Learning for Hackers: Case Studies and Algorithms to Get You Started*. O'Reilly Media, 2012.

[24] C. K. Jones, G. Wang, V. S. Yedavalli, and H. I. Sair, "Quantifying epistemic and aleatoric uncertainty in 3d u-net segmentation," in *medRxiv*, 2021.

[25] K. Ramamurthy, R. Menaka, A. Johnson, and S. Anand, "Neuroimaging and deep learning for brain stroke detection - a review of recent advancements and future prospects," *Computer Methods and Programs in Biomedicine*, vol. 197, p. 105728, 08 2020.

[26] C. Gillmann, L. Peter, C. Schmidt, D. Saur, and G. Scheuermann, "Visualizing multimodal deep learning for lesion prediction," *IEEE Computer Graphics and Applications*, vol. 41, no. 5, pp. 90–98, 2021.

[27] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?," 2017.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Interactive Editing of Voxel-Based Signed Distance Fields

Ole Wegen
Hasso Plattner Institute,
Digital Engineering Faculty,
University of Potsdam, Germany
ole.wegen@hpi.uni-potsdam.de

Jürgen Döllner
Hasso Plattner Institute,
Digital Engineering Faculty,
University of Potsdam, Germany
juergen.doellner@hpi.uni-potsdam.de

Matthias Trapp
Hasso Plattner Institute,
Digital Engineering Faculty,
University of Potsdam, Germany
matthias.trapp@hpi.uni-potsdam.de

Figure 1: Signed Distance Function (SDF) reconstructed from ScanNet [Dai17] scan. Left image: Original SDF. The ceiling is shaded very dark as the virtual light source is located inside the room. Right image: Cleaned SDF using the implemented manipulation possibilities. The editing time amounted to 5 minutes.

ABSTRACT

Signed distance functions computed in discrete form from given RGB-D data as regular voxel grids can represent manifold shapes as the zero crossing of a trivariate function; the corresponding meshes can be derived by the Marching Cubes algorithm. However, 3D models automatically reconstructed in this way often contain irrelevant objects or artifacts, such as holes or noise, due to erroneous scan data and error-prone reconstruction processes. This paper presents an approach for interactive editing of signed distance functions, derived from RGB-D data in the form of regular voxel grids, that enables the manual refinement and enhancement of reconstructed 3D geometry. To this end, we combine concepts known from constructive solid geometry, where complex models are created from simple base shapes, with the voxel-based representation of geometry reconstructed from real-world scans. Our approach can be implemented entirely on GPU to enable real-time interaction. Further, we present how to implement high-level operators, such as copy, move, and unification.

Keywords: Signed Distance Fields, Interactive Editing, GPU, CUDA

1 INTRODUCTION

Automated 3D reconstruction is a key functionality required in a growing number of application fields, such as robotics, autonomous driving, manufacturing, and spatial digital twins [Kha19]. Volumetric methods for 3D reconstruction are based on computing Signed Distance Functions (SDFs) for regular voxel grids, which encode manifold surfaces as zero-sets of a trivariate implicit function, allowing the acquisition of a large class of objects [Ber02]. Among the most popular raw data formats for volumetric 3D reconstruction methods are colored point clouds or depth-sensitive image data (RGB-D) [Keh14], generated by various scanning and acquisition technologies such as Light Detection and Ranging (LiDAR) sensors, which recently have even become built-in features in mobile devices (e.g., iOS TrueDepth camera); an overview of general 3D reconstruction methods based on RGB-D data is given

in [Zol18]. However, "completely digitizing an object or even an entire scene at high-quality is a tedious and time consuming process" [Zol18]. 3D reconstruction results are almost always not *perfect*. For example, a reconstructed 3D model can contain holes if parts of the scene were occluded during scanning, it can show artifacts that result from noise due to reflection in the scan data, or it can contain objects that are irrelevant for the concrete task.

In this paper, we investigate editing techniques that enable the manual refinement of geometry reconstructed based on RGB-D data from real-world indoor and outdoor scenes. To this end, we first convert RGB-D data into a regular voxel-based SDF representation. By combining the voxel-based SDF and procedural shapes into a hybrid scene representation, we enable interactive editing of reconstructed 3D geometry, e.g., for refinement and correction purposes, as shown in Figure 1.

1.1 Problem Statement

The use of SDFs for representing real-world scenes as voxel grids is common since 3D reconstruction by means of volumetric fusion of range data results in such a voxel grid [Cur96]. Furthermore, the creation and manipulation of SDF scenes using Constructive Solid Geometry (CSG) to combine multiple simple shapes into complex objects is also well known; tools such as MagicaCSG enable the creation of SDF scenes in exactly this manner. However, approaches are missing that combine the voxel-grid based SDF representation of real-world scenes reconstructed from scan data with the editing capabilities of CSG, i.e., the combination of procedural geometric shapes by means of set operations [Har95]. With respect to this, the following main challenges have to be considered:

C1: Handling Hybrid Scenes for SDFs: Editing techniques have to handle the hybrid character of SDF scenes, which consist of both procedural primitives (e.g., spheres, boxes) and voxel grids storing the SDF values.

C2: Real-Time Rendering for SDFs: Real-time editing needs interactive frame rates. Rendering voxel-based SDFs requires trilinear interpolation, resulting in many memory reads. If reasonably detailed resolutions for scenes should be achieved and additional (procedural) objects are present in the scene, a Graphics Processing Unit (GPU)-based implementation strategy has to be taken.

1.2 Approach & Contributions

We use a GPU-based approach for creating an SDF from RGB-D scans of real-world objects and scenes. To enable editing and refinement of such scenes at interactive frame rates, we present a new approach that can be implemented entirely on GPUs. It combines the representation of an SDF using a voxel grid with procedural shapes that can be integrated into the grid using set operations. Further, it enables the duplication as well as the translation of scene parts. Additionally, a unification approach enables the merging of the voxel-based and procedural-based SDF representation into a single representation, to increase rendering performance after editing. The presented manipulation techniques could also be transferred to neural geometry representations.

To summarize, this paper presents

1. interaction and manipulation techniques for voxel-based SDFs using procedural shapes,
2. an approach for unification of the scene with subsequent SDF recalculation for persisting changes after editing and thus increasing rendering performance, and
3. a GPU-based implementation of the presented techniques.

The remainder of this work is structured as follows: Section 2 reviews related work with respect to synthesis, rendering, and manipulation of SDF-encoded scenes. Section 3 presents a conceptual overview of the proposed approach. Section 4 details implementation aspects of our Compute Unified Device Architecture (CUDA)-based system. Section 5 evaluates the system's performance and discusses results and limitations. Finally, Section 6 concludes this work and presents ideas for future research.

2 BACKGROUND & RELATED WORK

In the following, we review related work regarding SDF representation, synthesis, rendering, and manipulation.

2.1 SDF Synthesis & Representation

SDFs can be represented by a combination of procedurally defined shapes, in form of voxel volumes, or as weights of a neural network. These representations differ with respect to use-cases and applications.

The procedural representation is mostly used in the context of CSG to build complex objects from simple base shapes. With respect to polygon-based geometry, Willis *et al*. presented PSML (Procedural Shape Modeling Language), which combines shape grammars with sequential statements and can be used to model complex models from 19 simple, predefined base shapes in a hierarchical manner [Wil21]. With respect to SDFs, Reiner *et al*. presented a modeling system that also uses a hierarchical scene graph structure and a CSG-based approach [Rei11]. The advantages of procedural approaches are the low memory consumption and, if visual editing is provided, the user-friendly creation process. The main disadvantage is the tedious nature of

the creation process for very complex objects or real-world-based scenes.

The voxel-based representation is commonly used for representing real-world scenes, reconstructed from scan data. Curless and Levoy proposed volumetric fusion for creating such voxel-based SDFs from range images [Cur96]. The range images are fused iteratively into an SDF voxel volume, utilizing the camera extrinsics and intrinsics. Based on that, Izadi *et al*. proposed KinectFusion [Iza11]; camera poses are estimated and utilized for fusing depth images GPU-based in real-time into a global implicit surface model. Such fusion approaches result in Truncated Signed Distance Field (TSDF) volumes that contain distance values only in vicinity to the geometry's surface.

Apart from automatic reconstruction from scan data, voxel-based SDFs can also be created from polygonal meshes or point clouds using distance transform algorithms. An example is the Jump Flooding Algorithm (JFA) [Ron06] that derives a Voronoi tessellation of a voxel grid, with respect to starting seed points. Based on this tessellation, the distance to the nearest seed can be computed per voxel cell. Using the points of a point cloud as starting seeds, the JFA can be used to compute an SDF approximation of the input geometry. Other algorithms utilize hierarchical data structures to directly compute mesh-to-voxel distances, e.g., MeshSweeper [Gue01]. The automatic creation from RGB-D data is one of the main advantages of voxel-based SDF representations. However, the manipulation of such geometry can be challenging due to the fine-grained nature of voxel-grids.

Recent neural approaches store the distance information in the weights of a neural net, which leads to a compact representation with respect to memory consumption [Tak21; Wan21], but increases rendering time, compared to classical representations. Additionally, the editing of such neural representations is challenging and an area of active research.

Our approach combines the procedural and voxel-based representation in order to be able to use automatic reconstruction from RGB-D data together with the easy manipulation known from CSG. The approach can be also adapted to neural representations.

2.2 SDF Rendering

SDFs are typically rendered using ray-marching, where for each pixel of the result image, a ray is emitted into the scene. The ray is advanced, using a fixed step size, until the distance to the surface at the ray's endpoint lies below a certain threshold. Subsequently, the final position of the ray can be used for determining color and normal information for shading.

Hart presented sphere tracing as a faster alternative to the classical ray marching [Har95]. The distance to the surface at a point in space corresponds to the radius of a sphere, in which no geometry is present. Therefore, in each iteration during ray-marching, a ray can be advanced by the distance retrieved from the SDF at the ray's current position, without intersecting geometry. This leads to shorter rendering times compared to classical ray marching. Keinert *et al*. proposed several techniques to enhance sphere tracing, for example an over-relaxation approach for faster tracing [Kei14]. While classical ray-marching can also be used for TSDFs, sphere tracing requires a full SDF. For rendering our scene representation, we rely on sphere tracing, as proposed by Hart.

2.3 SDF Manipulation

In his work on sphere tracing, Hart proved that set operations known from Boolean algebra can be implemented for SDFs using the min/max operators [Har95]. CSG approaches for SDFs, such as the one presented by Reiner [Rei10], utilize this insight for manipulation of procedural-based SDF representations. Zhang presented a GPU-accelerated system for voxel-based SDF modeling, also utilizing these set operations, as well as skeleton-based animation approaches [Zha16]. The focus of his work was, however, on manipulating single objects in rather small voxel grids that could be used, for example, for 3D printing. Our work also utilizes set operations for SDFs, but for real-world-based scenes stored in voxel grids. Additionally, we propose high-level manipulation techniques, such as copy and move functionalities.

3 METHOD

In the following, we give an overview of our SDF creation and rendering process, present our hybrid scene representation, and describe the user interaction concept for SDF scene editing.

3.1 Process Overview

For SDF creation, we iteratively fuse captured RGB-D data into a TSDF volume, as described in [Cur96]. To enable shpere tracing of the reconstructed geometry, we then convert the TSDF into a full SDF. For this we use the JFA, as described in Section 2.1. First, starting seed points are extracted from the TSDF volume by converting to a surface mesh, using Marching Cubes [Lor87], and subsequently sampling the mesh triangles uniformly. Then the JFA is used on these seed points to compute the full SDF. A volume containing color information can be created in the same way.

Rendering the SDF is achieved by means of sphere tracing with additional soft shadow rendering. Subsequently, the SDF can be manipulated, utilizing our heterogeneous SDF representation (Section 1.1, C1).

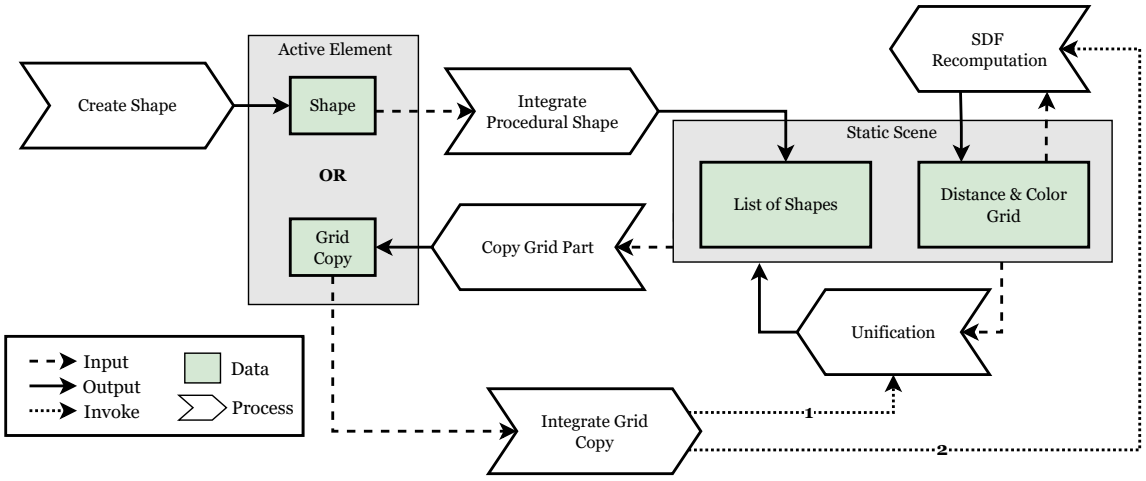


Figure 2: Overview of the processes and data in our approach: Green elements represent data and white arrows represent processes that can be initiated by the user. The static scene consists of two voxel grids, storing distance and color information, as well as a list of shapes that were added to the scene using set operations. The active element can be moved and placed in the scene and is either the currently selected, procedural shape, or a copy of a part of the static scene.

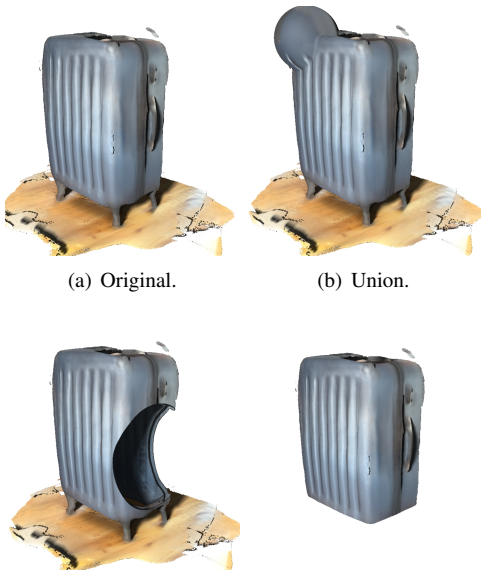
3.2 SDF Scene Representation

The hybrid SDF scene representation, we present, consists of the following components:

Distance & Color Grid: These voxel grids are reconstructed from RGB-D data, as described in Section 3.1.

List of Shapes: A list of procedural shapes that were added to the scene using set operations.

Active Element: The currently selected scene element (shape or voxel grid copy).



(a) Original. (b) Union.

(c) Subtraction. (d) Intersection.

Figure 3: Set operations of an SDF voxel grid (a) with a sphere shape in (b) and (c) and a box shape in (d).

The voxel grids and the list of shapes form the static scene, while the active element represents the dynamic part of the scene that is currently manipulated by the user.

3.3 User Interaction Concept

Figure 2 shows an overview of our system for interactive editing of SDFs. An SDF can be manipulated by creating new, procedural shapes ("Create Shape") and integrating them into the static scene ("Integrate Procedural Shape"). Additionally, parts of the static scene can be copied ("Copy Grid Part") and integrated into the scene at another position ("Integrate Grid Copy"). This copy functionality depends on a unification operation that can also be used for increasing rendering performance. All of the mentioned interaction techniques are described in the following.

Basic Operations. In our approach, geometry is manipulated by geometric shapes (such as spheres or boxes), introduced to a scene. Each shape possesses attributes, such as the position, orientation, size, color, and the scene integration type, all of which can be set by the user, using a Graphical User Interface (GUI). A pointer device (e.g., a mouse) is used to move and place the different shapes in the scene. A once added shape can be selected again by the user to edit its attributes or remove it from the scene.

With respect to the integration type of a shape, three types are supported, corresponding to set operations known from Boolean algebra (Figure 3). They can be implemented for SDFs, using the minimum and maximum operators, as described by Hart [Har95]:

Given two SDFs a and b; an SDF c, resulting from a set operation on the implicit surfaces defined by a

(a) Selecting part of the scene. (b) Placing the copy in the scene.

Figure 4: Example of a copy operation.

and b, can be computed as: $c = \min(a,b)$ (Union), or $\max(-a,b)$ (Subtraction), or $\max(a,b)$ (Intersection).

High-Level Operations. High-level manipulation features include copy and move functionalities, enabling the user to duplicate or move parts of the scene, such as single pieces of furniture in a room. First, the user selects the part of the scene to copy, using a "rubber-band" selection box. The selected part of the static scene is then copied and can be translated and rotated within the scene. Subsequently, the copied part is placed and integrated again into the scene. Figure 4 shows an example for a copy operation In the case of a move operation, a box with the size and position of the selection box is subtracted from the scene after copying.

Unification Operation. With an increasing number of shapes in the scene, the performance of sphere tracing decreases, due to more intricate distance queries. Therefore, the implemented system provides a unification functionality to convert a scene, consisting of a voxel grid and a list of procedural shapes, into a unified voxel grid to improve rendering performance (Section 1.1, C2). This functionality is also used when a copied part of the scene should be integrated back into the scene. After unification, the list of shapes is cleared and the new, unified voxel grid replaces the old one.

4 IMPLEMENTATION ASPECTS

The SDF creation from RGB-D data (TSDF fusion and JFA) was implemented using Python and C++ together with CUDA kernels for hardware acceleration. The SDF rendering and editing was implemented using C++ and CUDA version 11.3 and will be detailed in the following.

4.1 Scene Rendering

Listing 1 shows example code for the `querySDF` function that is invoked during sphere tracing to retrieve the distance to the surface for any point in space. Instead of only trilinearly interpolating in the voxel grid, the list of added shapes, as well as the active element have to be considered.

```
float querySDF(float3 p, float *voxelGrid, /*...*/) {
 float result = FLT_MAX;
 // Handling voxel grid SDF
 if(bboxHit[0]) { result = queryVoxelGrid(p, voxelGrid); }
 // Handling procedural shapes
 for(int i = 0; i < numberOfShapes; i++) {
  if(bool(bboxHit[i+1])){
   float3 queryPoint = mul(sceneShapes[i].rotation,
                           p-sceneShapes[i].position);
   auto combinationFunc = sdfCombinationFunc[sceneShapes[i].
         integrationType];
   auto shapeFunc = sdfShapeFunc[sceneShapes[i].type];
   result = combinationFunc(result, shapeFunc(queryPoint, sceneShapes
         [i].size));
  }
 }
 // Handling active element
 if(bool(bboxHit[numberOfShapes+1])) {
  float3 queryPoint = mul(activeElement.rotation,
                          p-activeElement.position);
  auto combinationFunc = sdfCombinationFunc[activeElement.
        integrationType];
  result = combinationFunc(result, activeElementSdf(queryPoint,
        activeElement));
 }
 return result;
}
```

Listing 1: CUDA code for querying the distance in an SDF scene.

Before sphere tracing, for each ray, the intersection of the ray with the bounding box of each shape and the voxel grid is tested. The results of these bounding box tests are stored into an array (`bboxHit`). This array can then be used to skip all shapes that the ray cannot hit. Apart from that, the following variables and functions are used in the code:

`sceneShapes` is the array of shapes that were already added to the scene.

`activeElement` is the currently selected shape or grid copy.

`sdfCombinationFunc` is an array of functions, implementing different set operations.

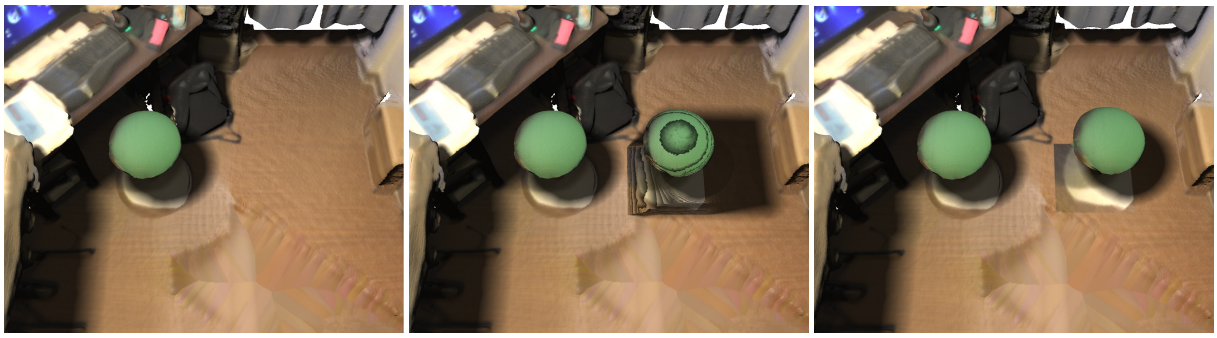
`sdfShapeFunc` is an array of signed distance functions for different shapes. By indexing into this array (and the `sdfCombinationFunc` array) with the corresponding shape type, unnecessary branching is avoided.

`queryVoxelGrid()` performs trilinear interpolation in a voxel grid.

`mul()` applies a transformation matrix to a point and is used for realizing rotation of objects.

`activeElementSdf()` returns the signed distance for the active element by either calling the corresponding `sdfShapeFunc` in the case of a procedural shape or `queryVoxelGrid` in the case of a voxel grid copy.

After a ray has terminated, the ray's endpoint is used to retrieve the surface color from the color volume. Additionally, synthetic soft shadows can be rendered by including an additional tracing step from the surface point to a light source to check for occluding geometry in between.



(a) (b) (c)

Figure 5: Copying parts of the input scene (a) can lead to shadowing artefacts (b). An SDF recalculation step resolves the problem (c).

4.2 Scene Manipulation

As described in Section 3.3, procedural shapes can be added to the scene using set operations implemented by means of min/max operators. Whenever a user adds a shape, it is set as the `activeElement` that can be moved and rotated. On integration (e.g., triggered by clicking the left mouse button), the currently active shape gets appended to the `sceneShapes` array. During scene rendering, for each ray, the shape ID of the shape with the minimal distance to the ray's endpoint is stored. This allows for shape selection by point-and-click, as each pixel of the rendered frame can be matched to the corresponding shape by means of the stored shape ID. This ID is also used during shading to retrieve the respective material.

For the copy functionality, the `querySDF` function is used to fill a voxel grid of the size of the selection box with signed distance values. Negative values at border voxels are set to zero to avoid "leaking" at cut borders. This grid copy is then set as the active element. However, for sphere tracing the SDF has to be defined at any point in space, which is not the case for the grid copy, which is only defined within its bounds. This leads to incorrect rendering results. To mitigate this problem, the grid copy therefore returns the distance to its bounding box for points outside this bounding box and the real distances for points inside the bounding box (as suggested by [Rei10]). However, to ensure that the rays during sphere tracing do not stop at the bounding box, the distance to a slightly shrinked bounding box is returned. When the user places the grid copy within the scene, a unification step is performed to integrate the copied geometry.

4.3 Unification & Recalculation

The unification is implemented as a CUDA kernel that executes the `querySDF` function (Listing 1) for each voxel, storing the retrieved distances in a new voxel grid. However, if integrating copied parts into the scene in this manner, problems can arise during shadow computation. As these copied parts return the distance to their bounding box for query points located outside of it, the otherwise invisible bounding boxes result in unwanted shadows and shadowing artefacts (Figure 5(b)). Additionally, the rendering performance can decrease, as a ray first has to approach the bounding box of a copied part and only inside this bounding box can retrieve the actual distances, increasing the number of sphere tracing steps. Further, the number of tracing steps also increases after subtraction and intersection operations, as the computation using minimum and maximum operators only results in a lower bound to the actual distance, as Hart notes in [Har95]. An SDF recalculation step was implemented to mitigate the described problems.

It first converts the SDF into a voxel grid, containing seed points for a JFA pass. For each voxel that implicitly contains parts of the geometry's surface, a surface point is required. Since, SDFs store only distances not the surface points itself, we compute the surface point as follows (Figure 6). The surface normal $\vec{n}$, which corresponds to the gradient at this point, is computed using central differences. Subsequently, the center point C and the distance d to the surface S are used for computing surface point $P = C - \vec{n} \cdot d$.

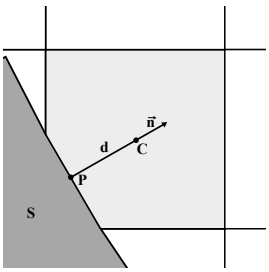
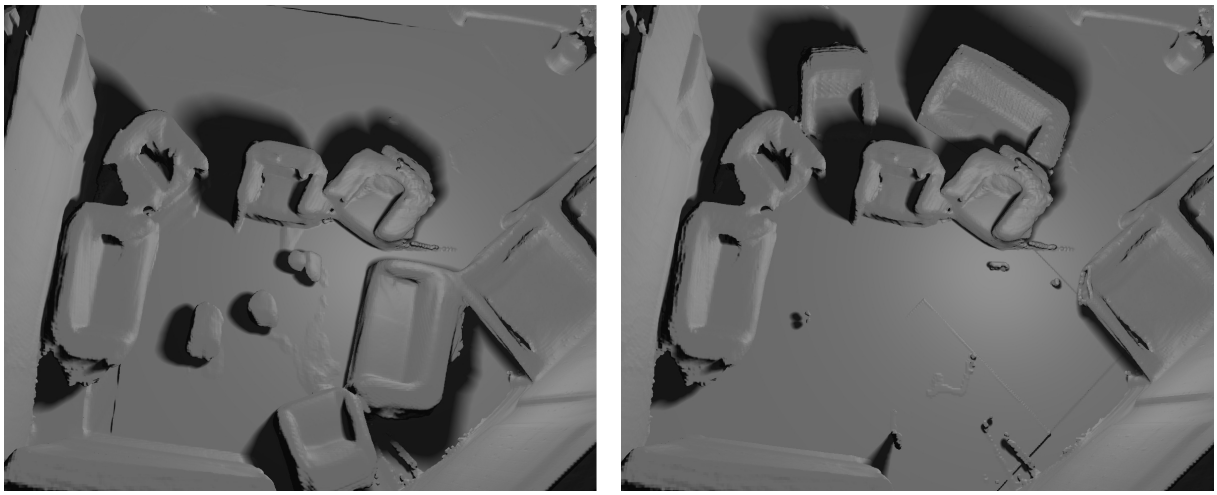


Figure 6: Computation of surface point P.

After the voxel grid has been filled with seed points, the JFA is executed to obtain the exact distances to the surface for every voxel (see Section 2.1). For the copied parts, the bounding box is omitted before the JFA is applied. Figure 5(c) shows that the shadows are rendered correctly after the recalculation step.

The JFA only results in an approximation of the surface represented by the SDF. To reduce accumulation of errors, when the recalculation is executed several



(a) Original SDF from IPad scan (b) SDF after moving objects

Figure 9: Manipulation of an SDF using the move operator. The SDF was reconstructed from an RGB-D scan obtained by an IPad.

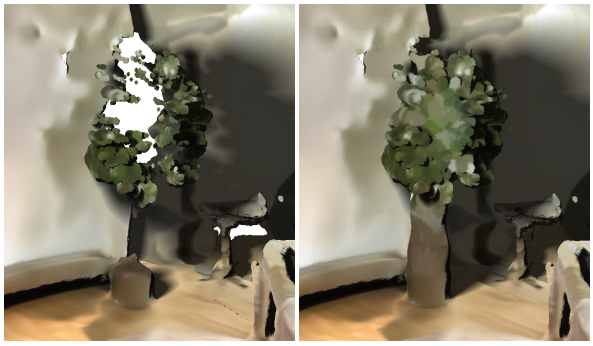
times, we use an additional pass at the start of the JFA (the so called 1+JFA variant). Using this JFA variant lead to no observable errors in the geometry, even after repeated SDF recalculation.

5 RESULTS & DISCUSSION

The following section presents exemplary results, created with the implemented manipulation techniques. Subsequently, the run-time performance is evaluated and current limitations are discussed.

5.1 Exemplary Results

Figure 1 (on the first page) shows, how the presented manipulation techniques can be used to refine the 3D geometry of a scene reconstructed from RGB-D data.



(a) Original (b) Edited

Figure 7: Low-quality SDF of a pot tree, reconstructed from a real-world scan with an IPad. (a) shows the original, noisy, and incomplete SDF. (b) shows the edited SDF, where the tree stump was connected to the tree crown by merging spheres in the scene's color. The editing time amounted to 1-2 minutes.

Holes are filled with shapes in the scene's color, noise in the scene is removed, and unwanted parts, such as the ceiling are cut away. This takes only a few minutes and improves the reconstructed geometry noticeably. Figure 7 shows a similar scenario, where the geometry of a tree, reconstructed from a low-resolution scan, is completed, using the implemented set operations. The common problem of faulty reconstructions from real-world data can therefore be countered, using the proposed manipulation techniques.

Apart from countering problems, such as holes and noise in the scene, the proposed techniques can also be used to add new objects to the scene or alter existing ones, e.g., for artistic purposes. Figure 8 shows an example of this. Additionally, the move functionality can be used for arranging furniture in a reconstructed room anew (Figure 9). This can be useful for interior design and room planning. First, a furnished room

(a) Input SDF (b) Edited SDF

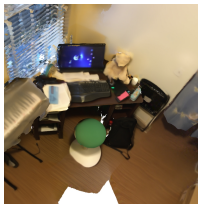
Figure 8: Example of structure and appearance editing. The original 3D reconstruction (a) was edited by filling holes in the scene, changing the color of the stool by merging a red sphere into it's upper part, and adding ears and eyes to the teddy bear by merging spheres of the scene's color (b).

is scanned and reconstructed. Subsequently, new furniture arrangements can be tested, without actually having to move the furniture in the real world. All these manipulation techniques are easy and fast to use, due to the GPU-based implementation and the presented user interaction concept.

5.2 Run-Time Performance Evaluation

In the following, we show an evaluation of the run-time performance of the system. The performance was measured on an AMD Ryzen 5 5600x (6 cores, 3.7 GHz) Central Processing Unit (CPU) with 32 GB DDR4 RAM and an NVIDIA GeForce RTX 3090 GPU with 24 GB VRAM. The application runs on a Windows 10 operating system at a viewport resolution of 1200×960 pixels.

For the measurements, reconstructions from a ScanNet scene with different voxel grid resolutions were used, rendered from three different virtual camera views (Figure 10). The rendering time per frame was measured over 12 seconds and the results averaged.



(a) Camera 1

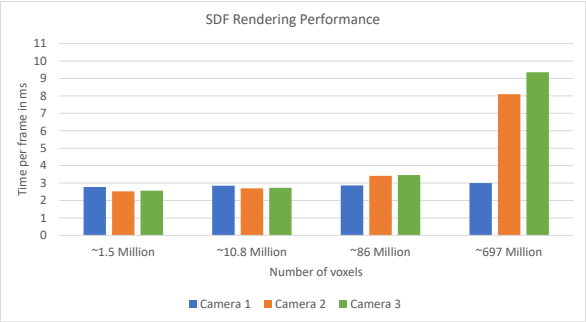
(b) Camera 2

(c) Camera 3

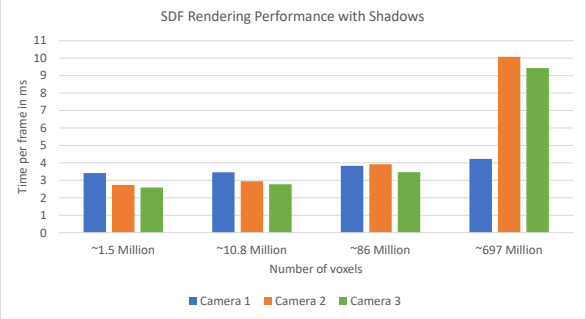
Figure 10: The three different camera configurations used for acquiring performance measurements. Inside the room (a), above (b), and below (c).

Figure 11(a) shows the measurements for rendering the test scene without soft shadows or any additional objects. Even when the full scene is in view (which is the case for camera 2 and 3), it can be rendered in real-time. Figure 11(b) shows measurements for the same set-up, but with soft shadows activated. An increase of up to 2ms rendering time per frame can be observed, especially for the largest scene. Rendering is still possible in real-time.

Figure 12(b) shows how the rendering timings change if additional objects are present in a scene with approx. 697 million voxels. A number of spheres were placed randomly in the scene (Figure 12(a)). The rendering time increases with an increasing amount of objects. Up to 30 objects can be rendered together with the voxel grid at a maximum frame time of around 16ms. For more than 30 objects, the higher frame times result in less than 60 frames per second during rendering. After a unification step is applied, the rendering time always decreases again to approximately the time measured for zero objects in the scene.

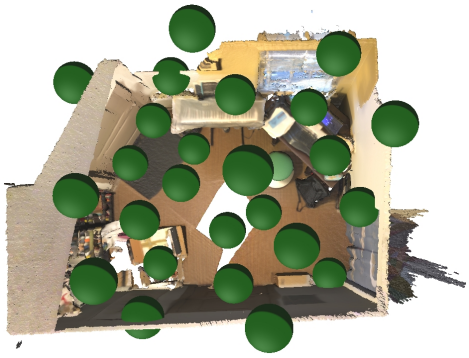


(a) Without shadows

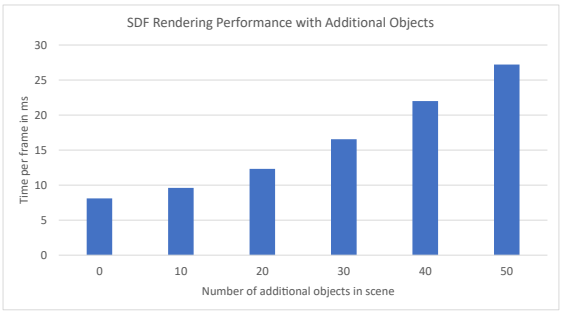


(b) With soft shadows

Figure 11: Performance results for rendering SDFs of different sizes.



(a) Example scene for measuring the performance of the SDF renderer when additional objects are present.



(b) Performance for rendering SDF voxel grids with additional objects added to the scene.

Figure 12: Rendering an SDF voxel grid ($917 \times 1044 \times 728$) with additional objects.

Voxel Grid Resolution	#Voxels	Unification	SDF Recalculation
$119 \times 134 \times 95$	$\sim 1.5 \cdot 10^6$	3 ms	24 ms
$229 \times 260 \times 181$	$\sim 10.8 \cdot 10^6$	17 ms	204 ms
$457 \times 520 \times 362$	$\sim 86 \cdot 10^6$	146 ms	1024 ms
$917 \times 1044 \times 728$	$\sim 697 \cdot 10^6$	903 ms	8362 ms

Table 1: Runtime for executing the unification and SDF recalculation steps for different voxel grid resolutions.

Table 1 shows the processing times for the unification and SDF recalculation for voxel grids of different size. These steps can be used for increasing rendering performance, fixing shadow computations after manipulation, and integrating copied or moved parts of the scene. For large voxel grids, the SDF recomputation can require several seconds.

5.3 Limitations

While the feasibility of real-time rendering and editing of real-world scenes represented as SDFs was demonstrated in this work, there are still some open questions and constraints that need to be addressed. A major limitation is currently the memory consumption as full SDF voxel volumes are used. With several hundred million voxels, we are able to represent single rooms with sufficient detail, but larger scenes are still difficult to reconstruct and render. Further, while the unification step increases rendering performance, it is irreversible and makes the subsequent editing of already added shapes impossible. A snapshot-based approach could be used to implement an undo operation. Additionally, while the unification can be executed in less than a second, the SDF recalculation step requires more time. If soft shadows should be rendered, the recalculation step is necessary after every copy/move operation, which can interrupt the work flow if the voxel grid is large, as the execution time of the recalculation step then amounts to several seconds. With regard to user interaction, it is possible to select and edit shapes that were added to the scene using a union operator. Shapes that were integrated into the scene using subtraction or intersection can not be selected for further editing, as during sphere tracing only the shape IDs of shapes where the ray terminates are retrieved. Suitable selection mechanisms have to be developed in the future.

6 CONCLUSIONS & FUTURE WORK

This paper presented an approach for interactive editing of voxel-based signed distance fields. It builds on a hybrid representation consisting of a voxel grid and a number of procedural shapes (Section 1.1, C1), which enables easy manipulation for refining geometry (e.g., closing holes or removing artefacts). Thus, this work is a further building block for creation of high-quality 3D models from real-world scenes, by enabling manual refinement of 3D reconstruction results. High-level manipulation methods, such as copy and move functionality, provide suitable techniques for altering real-world-based geometry and therefore facilitate design and planning processes. The GPU-based implementation of rendering and manipulation enables real-time interaction (Section 1.1, C2).

By exchanging the voxel grid with a neural representation for rendering, the proposed techniques could also be used for altering neural geometry representations. Nevertheless, an additional training phase would be required to transfer the changes back into the neural representation.

The SDF manipulation still has some limitations and extension possibilities. In the future, we will address the problem of memory consumption, by evaluating sparse data structures to reduce memory constraints and allow for larger scenes. Additionally, appearance manipulation (e.g., altering the color volume through a painting technique) would be a useful extension to the already implemented manipulation methods.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable feedback. This work was partially funded by the German Federal Ministry of Education and Research (BMBF) through grants 01IS18092 (“mdViPro”) and 01IS19006 (“KI-LAB-ITSE”).

REFERENCES

[Ber02] Fausto Bernardini and Holly Rushmeier. “The 3D Model Acquisition Pipeline”. In: *Computer Graphics Forum* 21.2 (2002), pp. 149–172. DOI: https://doi.org/10.1111/1467-8659.00574. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/1467-8659.00574.

[Cur96] Brian Curless and Marc Levoy. “A Volumetric Method for Building Complex Models from Range Images”. In: *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’96. New York, NY, USA: Association for Computing Machinery, 1996,

pp. 303–312. ISBN: 0897917464. DOI: 10.1145/237170.237269.

[Dai17] A. Dai et al. “ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, July 2017, pp. 2432–2443. DOI: 10.1109/CVPR.2017.261.

[Gue01] A. Gueziec. “"Meshsweeper": dynamic point-to-polygonal mesh distance and applications”. In: *IEEE Transactions on Visualization and Computer Graphics* 7.1 (2001), pp. 47–61. DOI: 10.1109/2945.910820.

[Har95] John Hart. “Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces”. In: *The Visual Computer* 12 (June 1995), pp. 527–545. DOI: 10.1007/s003710050084.

[Iza11] Shahram Izadi et al. “KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera”. In: *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. Ed. by Jeffrey S. Pierce, Maneesh Agrawala, and Scott R. Klemmer. ACM, 2011, pp. 559–568. DOI: 10.1145/2047196.2047270.

[Keh14] Wadim Kehl, Nassir Navab, and Slobodan Ilic. “Coloured signed distance fields for full 3D object reconstruction”. In: *British Machine Vision Conference, BMVC 2014, Nottingham, UK, September 1-5, 2014*. Ed. by Michel François Valstar, Andrew P. French, and Tony P. Pridmore. BMVA Press, 2014.

[Kei14] Benjamin Keinert et al. “Enhanced Sphere Tracing”. In: *Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference*. Ed. by Andrea Giachetti. The Eurographics Association, 2014. ISBN: 978-3-905674-72-9. DOI: 10.2312/stag.20141233.

[Kha19] Siavash H. Khajavi et al. “Digital Twin: Vision, Benefits, Boundaries, and Creation for Buildings”. In: *IEEE Access* 7 (2019), pp. 147406–147419. DOI: 10.1109/ACCESS.2019.2946515.

[Lor87] William E. Lorensen and Harvey E. Cline. “Marching cubes: A high resolution 3D surface construction algorithm”. In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1987, Anaheim, California, USA, July 27-31, 1987*. Ed. by Maureen C. Stone. ACM, 1987, pp. 163–169. DOI: 10.1145/37401.37422.

[Rei10] Tim-Christopher Reiner. “Interactive Modeling with Distance Fields”. MA thesis. University of Stuttgart - Institute for Visualization and Interactive Systems, Feb. 2010.

[Rei11] Tim Reiner, Gregor Mückl, and Carsten Dachsbacher. “Interactive modeling of implicit surfaces using a direct visualization approach with signed distance functions”. In: *Computers & Graphics* 35 (June 2011), pp. 596–603. DOI: 10.1016/j.cag.2011.03.010.

[Ron06] Guodong Rong and Tiow-Seng Tan. “Jump Flooding in GPU with Applications to Voronoi Diagram and Distance Transform”. In: *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*. I3D ’06. Redwood City, California: Association for Computing Machinery, 2006, pp. 109–116. ISBN: 159593295X. DOI: 10.1145/1111411.1111431.

[Tak21] Towaki Takikawa et al. “Neural Geometric Level of Detail: Real-Time Rendering With Implicit 3D Shapes”. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 2021, pp. 11358–11367.

[Wan21] Yifan Wang, Lukas Rahmann, and Olga Sorkine-Hornung. “Geometry-Consistent Neural Shape Representation with Implicit Displacement Fields”. In: *CoRR* abs/2106.05187 (2021). arXiv: 2106.05187.

[Wil21] Andrew R. Willis et al. “Volumetric procedural models for shape representation”. In: *Graph. Vis. Comput.* 4 (2021), p. 200018. DOI: 10.1016/j.gvc.2021.200018.

[Zha16] Di Zhang. “A GPU Accelerated Signed Distance Voxel Modeling System”. PhD thesis. University of Washington, 2016.

[Zol18] Michael Zollhöfer et al. “State of the Art on 3D Reconstruction with RGB-D Cameras”. In: *Computer Graphics Forum* 37 (May 2018), pp. 625–652. DOI: 10.1111/cgf.13386.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Magnitude of Semicircle Tiles in Fourier-space - A Handcrafted Feature Descriptor for Word Recognition using Embedded Prototype Subspace Classifiers

Anders Hast[1,2]
[1] Department of Information Technology
Uppsala University
Centre for Image Analysis
SE-751 05 Uppsala, Sweden
anders.hast@it.uu.se
[2] The Swedish Institute for Children's Books
SE-113 22 Stockholm, Sweden

ABSTRACT

The purpose of this paper is to in detail describe and analyse a Fourier based handcrafted descriptor for word recognition. Especially, it is discussed how the Variability in the results can be analysed and visualised. This efficiency of the descriptor is evaluated for the use with embedded prototype subspace classifiers for handwritten word recognition. Nonetheless, it can be used with any classifier for any purpose. An hierarchical composition of discrete semicircles in the Fourier-space is proposed and it will will be show how this compares to Gabor filters, which can be used to extract edges in an image. In comparison to Histogram of Oriented Gradients, the proposed feature descriptor performs better in this scenario. Compression using PCA turns out to be able to increase both the F_1-score as well as decreasing the Variability.

Keywords

Discrete Fourier Transform, Gabor Filters, Subspaces, Embedded Prototypes, Clustering, F_1 score, Variability, Deep Learning, t-SNE.

1 INTRODUCTION

In recent times, *Embedded Prototype Subspace Classification* (EPSC) [HV21, HLV19, HL20, HV21] has proven to be able to classify datasets of various kinds, containing everything from single digits, characters to whole words, and even objects. Datasets used have been the MNIST dataset of handwritten digits [LCB10], E-MNIST containing letters [CATvS17], the Kuzushiji-MNIST dataset containing Japanese handwritten characters [CBK*18], the Fashion MNIST (F-MNIST) [XRV17] containing small images of clothes and accessories and a recently published dataset [HV21] based on the Esposalles dataset [RFS*13], where 30 different words were extracted to create an imbalanced dataset with a total of 16354 word images. This latter one will be used for the subsequent analysis in this paper.

The main contributions of this paper are as follows. First of all the Fourier based handcrafted feature descriptor (previously called mFFT) used in [HLV19, HL20, HV21] will for the first time be described and analysed in detail. Furthermore, an hierarchical composition of discrete semicircles in the Fourier-space is proposed and it will be shown how this compares to Gabor filters, which can be used to extract edges of different orientations and sizes in an image [MNR92]. This more elaborate feature descriptor will subsequently be called: *Magnitude of Semicircle Tiles in Fourier-space*, or MoSTiF for short.

Moreover, it will be shown how the EPSC can be optimised to use the proposed feature descriptor for fast matching. And last but not least, it will be shown that the when performing bootstrapping on a dataset, varying the size of the split between learning and validation partitions, the standard deviation (SD) of F_1 score can be a useful for understanding the behaviour of the classifier. The word *Variability* is often used as a synonym to SD, but here it will subsequently be used to denote the SD of the F_1 score in particular.

2 BACKGROUND

The main advantage of EPSC compared to many deep learning based methods [Sha18] for handwritten text recognition [KDJ18, DKMJ18, SF16] is that EPSC is shallow to its nature, with no hidden layers, and therefore does not require powerful GPU resources in the training process. In general, EPSC learns from the embedding of feature vectors, using dimensionality reduction techniques like t-SNE, UMAP or SOM [HV21] and then creates so-called subspaces from each cluster [KLR*77], which are a set of neurons specialised on identifying the class variation captured in that cluster. Obviously, EPSC does not always outperform the state-of-the-art deep learning approaches when it comes to accuracy. However, both learning and inference will generally be much faster due to its simplicity and compactness. Moreover, both the learning and classification processes are inherently easy to interpret [Kri19, CPC19], explain [ADRS*19, GSC*19, CPC19], and visualise.

The EPSC uses handcrafted features as input, while deep learning approaches such as Convolutional Neural Networks (CNN) have been efficiently used to extract learned features from images [SSTF*15, ZK15]. One drawback with learned features is the time consuming learning process. Since it has been noticed that CNN's produce Gabor-like features some efforts have been done to replace the CNN with Gabor filters [LCZ*18, JJL07]. It will be shown that this idea can be utilised by creating a hierarchical composition of discrete semicircles in the Fourier-space.

Several handcrafted features have been proposed in the literature, and some popular methods include Scale Invariant Feature Transform (SIFT) [Low04], Speeded Up Robust Features (SURF) [BETVG08] and Histograms of Oriented Gradients (HOG) [DT05], where some have been used in recognition systems [GDDM14]. Fourier based detectors can be constructed by taking the magnitude of the lowest frequency elements of signals [HV18, HSSK18]. Matuszewski et al. compute the magnitude from a few elements close the the centre of the shifted Fourier transform [MHWS17] while Buchholz and Jug [BJ21], create what they call a Fourier Domain Encoding (FDE) by computing normalised amplitude and phase of half the concentric Fourier rings. Herein, a mix of these two methods is proposed, by computing the magnitude of neighbouring Fourier semicircles. This is basically what was used previously for mFFT in [HLV19, HV21] together with HOG. However, here is proposed the extension of a hierarchy of image partitions making the feature vector more effective, which can be used without HOG. Since HOG has proven to be both simple and effective, the proposed descriptor will be compared to HOG as a reference.

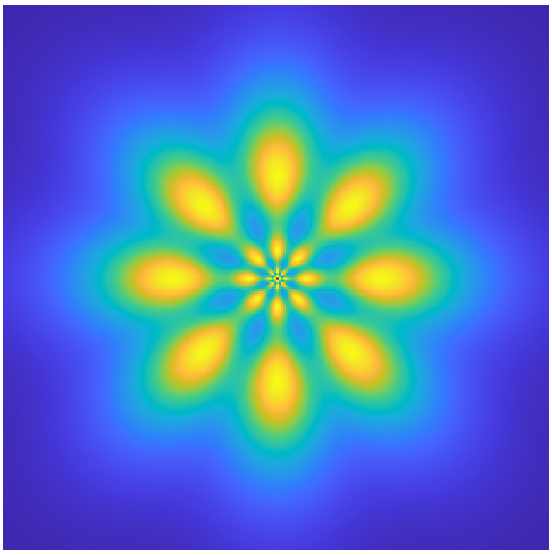


Figure 1: The Gabor filter banks in Fourier-space. The symmetric Gabor filter bank is distributed in four orientations and three frequency bands in this example.

3 THE MAGNITUDE OF SEMICIRCLE TILES IN FOURIER-SPACE

In this section the Magnitude of Semicircle Tiles in Fourier-space (MoSTiF) feature descriptor is introduced. The idea comes from the fact that CNN's create something similar to what Gabor filters does, which can be used to detect lines in different directions and frequencies. These are then combined in subsequent levels to more complicated features. In the EPSC this is done using the subspaces. Subspace classification is done by computing the norm of the projected feature vector to be classified into each subspace. The subspace of a certain class yielding the largest norm will tell what class the feature vector most likely belongs to. Subspace classification will be explained more in detail later.

As shown in Figure 1, a Gabor filter bank can be created by using Gaussians in a symmetric fashion. Each pairwise filter (placed diametrically with respect to the centre) corresponds to an orientation of the features. The further away from the centre the pairs are placed, the higher the frequency. Hence, only the innermost filters are of interest since they detect shape, while higher frequencies corresponds to very fine details or noise. Since subspaces are used to determine the more complicated features, elements are simply picked in a space filling [BHB16] semicircle and the magnitude of the complex Fourier value of the Discrete Fourier Transform (DFT) is computed as:

$$|\mathscr{F}[f(n)]| = \sqrt{\Re(\mathscr{F}[f(n)])^2 + \Im(\mathscr{F}[f(n)])^2}. \quad (1)$$

where $f(n)$ is the image f at point n.

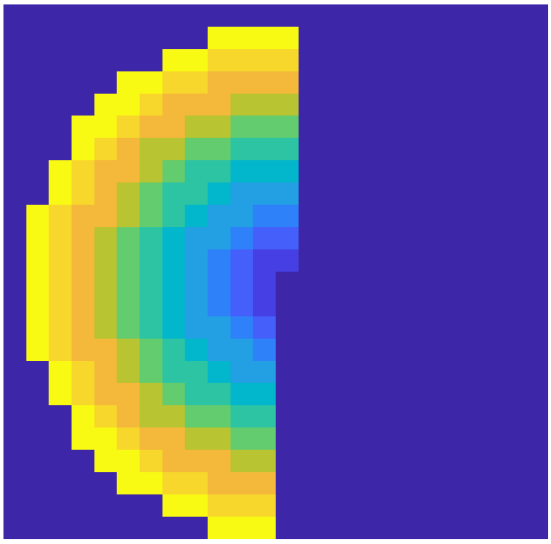


Figure 2: The different space filling semi-circles are shown in different colours.

Remember that the elements are diametrically distributed and therefore only half of the space filling circle is needed to construct a feature vector. Hence, a semicircle can be regarded as sampling the filter banks for a thin band of frequencies in all possible directions.

Figure 2 shows the possible space filling semi-circles for a 24×24 image patch. Note that the central pixel is not used since it corresponds to the DC content of the image, i.e. the overall brightness.

Computing one set of semi-circles on the whole image would not be efficient enough. Therefore the semicircles are also computed on different partitionings of the image in a hierarchical manner and then combined into a longer feature vector. Figure 3 shows three different combinations of partitions. The number above each sub-partition shows the number of semicircles used. The number to the right shows the total feature vector length.

By choosing the size 120×120 the image width and/or height can be evenly divided by $2, 3, 4, 5$ and 6. This gives 36 different possible block partitions of the input image. Each semicircle is concatenated in order to construct each part of the feature vector, which is subsequently normalised for each block in the partition. When all blocks and partitions are computed the final feature vector is also normalised.

3.1 Subspace Classification

Since it was first proposed by Watanabe et al. [WP73] in 1967, Subspaces have been used for classification in pattern recognition. This approach was later further developed by Kohonen and others [WLK*67, KLR*77, KO76, KRMV76, OK88]. In general, subspace classification can be regarded as a two layer neural network [HLV19, OK88, Laa07], where the weights are not learned using time consuming backpropagation. Instead weights are mathematically defined through Principal Component Analysis (PCA) [Laa07]. (Note that PCANet [CWW15] also set weights using PCA. However, this is done for features in a sliding window manner and is therefore fundamentally different from Subspace classification.) Last but not least, one important advantage is that the whole learning process can easily be visualised, since it is based on visualisation techniques (e.g. t-SNE), which makes it easy to both understand, interpret and explain, as compared to most of the state-of-the-art deep learning approaches, which are often regarded as black boxes.

Every image to be classified is represented by a feature vector $\mathbf{x}$ with m real-valued elements $\mathbf{x}_j = \{x_1, z_2 ... x_m\}, \in \mathbb{R}$, such that the operations take place in a m-dimensional vector space $\mathbb{R}^m$. Any set of n linearly independent basis vectors $\{\mathbf{u}_1, \mathbf{u}_2, ... \mathbf{u}_n\}$, where $\mathbf{u}_i = \{w_{1,j}, w_{2,j} ... w_{m,j}\}, w_{i,j} \in \mathbb{R}$, which can be combined into an $m \times n$ matrix $\mathbf{U} \in \mathbb{R}^{m \times n}$, span a subspace $\mathscr{L}_\mathbf{U}$

$$\mathscr{L}_\mathbf{U} = \{\mathbf{x} | \mathbf{x} = \sum_{i=1}^{n} \rho_i \mathbf{u}_i, \rho_i \in \mathbb{R}\} \tag{2}$$

where,

$$\rho_i = \mathbf{x}^T \mathbf{u}_i = \sum_{j=1}^{m} x_j w_{i,j} \tag{3}$$

Classification of a feature vector can be performed by projecting $\mathbf{x}$ onto each and every subspace $\mathscr{L}_{\mathbf{U}_k}$. The vector $\hat{\mathbf{x}}$ will in this way be a reconstruction of $\mathbf{x}$, using n vectors in the subspace through

$$\hat{\mathbf{x}} = \sum_{i=1}^{n} (\mathbf{x}^T \mathbf{u}_i) \mathbf{u}_i \tag{4}$$

$$= \sum_{i=1}^{n} \rho_i \mathbf{u}_i \tag{5}$$

$$= \mathbf{U}^T \mathbf{U} x^T \tag{6}$$

By normalising all the vectors in $\mathbf{U}$, the norm of the projected vector can be simplified as

$$||\hat{\mathbf{x}}||^2 = (\mathbf{U}x^T) \cdot (\mathbf{U}x^T) \tag{7}$$

$$= (\mathbf{U}x^T)^2 \tag{8}$$

$$= \sum_{i=1}^{n} {\rho_i}^2 \tag{9}$$

Therefore, the feature vector $\mathbf{x}$, which is most similar to the feature vectors that were used to construct the subspace in question $\mathscr{L}_{\mathbf{U}_k}$, will therefore have the largest norm $||\hat{\mathbf{x}}||^2$.

3.2 Parameters to learn

As mentioned earlier, subspaces do not require learning through backpropagation, since the learning itself is

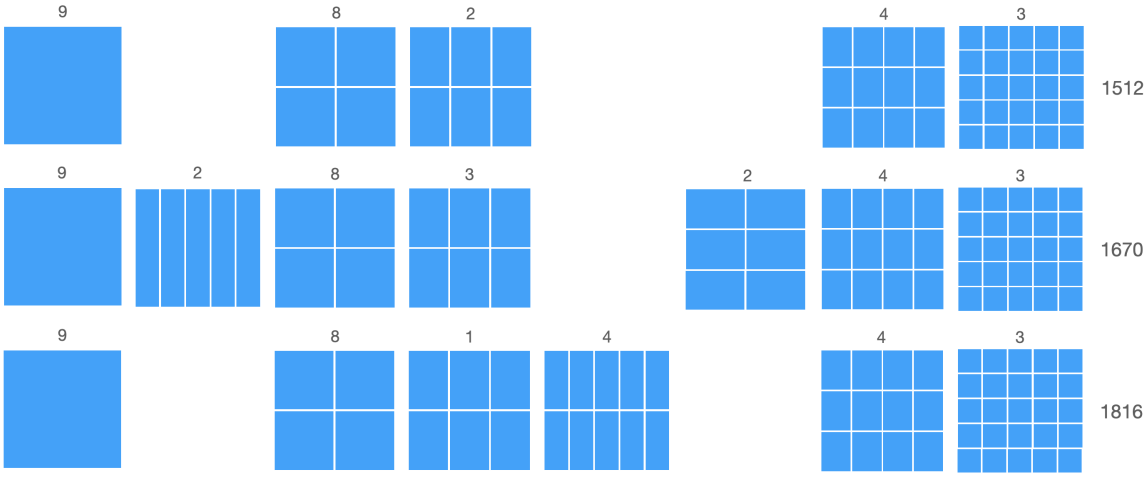


Figure 3: Three different feature vectors and their different partitionings are shown, one per row. The number of semicircles are denoted above each partition and the feature length is reported to the right.

done by the embedding obtained from some dimensionality reduction method such as t-SNE [MH08], UMAP [MH18] or SOM [Koh82]. Moreover, all the weights in the resulting neural network are set mathematically by PCA.

Nevertheless, bootstrapping can be used to evaluate and set parameters that are required for the overall performance. Such parameters are the feature detector itself, i.e. how many semicircles are to be used for different partitions. So far we have not devised a technique for doing that. Instead a Monte Carlo sampling approach was used to find the best parameters. The draw back is of course that this can be time consuming. Nonetheless, we think that the proposed partitionings shown in Figure 3 can be used for any dataset of handwritten words.

The subspace projection itself is done using only 6 dimensions, and it has been noted that this can be varied to improve performance when the size of the dataset to learn from is changed. Moreover, the effectiveness of PCA compression of the feature vector can also be evaluated as will be shown later herein.

3.3 Improved Embedding and Clustering

The idea of EPSC [HV21, HLV19] is to use some embedding technique to obtain prototypes for the construction of each subspace. In this work t-SNE [MH08], was used to reduce the number of dimensions of high dimensional data down to 2 dimensions. In this process, clusters are formed since t-SNE strives to move similar features (represented by their projected points) closer to each other and dissimilar points are kept further away from each other.

Previously, Hast et al. [HLV19] used kernel density estimation (KDE) [CHTT96] and watershed transform on the inverse image to find clusters in a two-dimensional image space, which is basically the same as performing

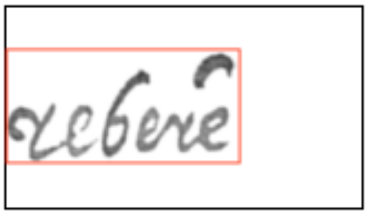


Figure 4: The placement of each word in the 90×160 rectangular bounding box. The background is removed but the word is not binarised. And the red rectangle shows how the word is cut out and then resampled to 120×120.

the Mean-Shift [CM02, FH75]. However, it was shown that other algorithms that requires specifying the exact number of clusters, such as K-means [HW79] could also be used [HV21]. Moreover, it was shown that instead of using a certain bandwidth for clustering, better performance was achieved by computing k clusters, striving for these clusters to contain a certain predefined number of features n_f. Herein it was chosen to use a fixed k instead. In fact $k = 2$ was chosen as it gave the best overall classification for this dataset (these initial experiments are not reported herein, since there are reasons to believe that k depends on the data at hand and possibly also the amount of data). Furthermore, the process was simplified by replacing the inverse watershed with a gradient ascend method working on each data point instead of each pixel in the KDE image. Both these changes speeded up the learning process noticeably.

Another improvement, which gave an overall higher F_1 score, is shown in Figure 4, where each word image is cut out (red bounding box) and extracted from the image and resampled to the size 120×120..

4 EVALUATION AND METHOD

Opitz and Burst [OB21] show that the best choice to compute the F_1 scores for imbalanced datasets, is the arithmetic mean over harmonic means. I.e. F_1 scores are computed for each class and then averaged via arithmetic mean, such that

$$\mathbb{F} = \frac{1}{n}\sum_{i=1}^{n} F_1 = \frac{1}{n}\sum_{i=1}^{n} \frac{2P_iR_i}{P_i+R_i} \tag{10}$$

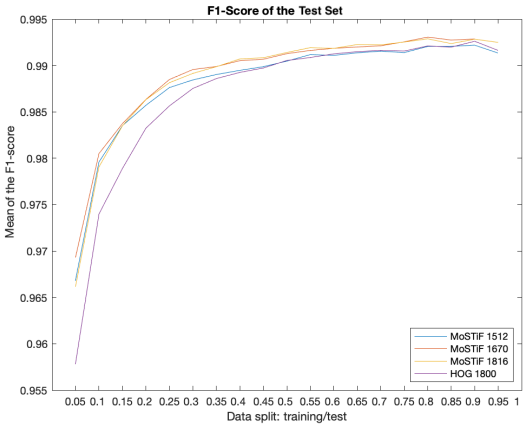
where P_i and R_i are precision and recall respectively for each class i.

It was chosen to compute the F_1 scores using the Bootstrapping method [Koh95, KW96], with stratification because the data is imbalanced. This means that the bootstrap sample is taken from the original set by using *sampling with replacement*, and that both the learning, test and validation sets are forced, as much as possible, to contain a certain percentage of each class. The validation set was kept the same throughout the experiments, using 50% of the available data. The remaining 50% was split into a set for training and a set for testing. The experiments where conducted 200 times for each data split, varying the permutations randomly, in order to be able to analyse the impact of what data are in each split, i.e the training and test set. By varying the percentage used for training, the change in F_1 scores could be analysed and parameters could be learned and set accordingly, as previously discussed. Moreover the SD of the F_1 score for each run was computed and is being called *Variability* in the subsequent presentation of the results. In any case, varying the split gives a chance to analyse the impact on the overall performance of the feature vectors in combination with EPSC.

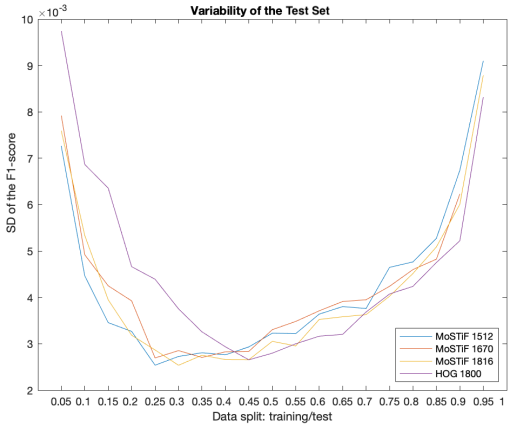
5 RESULTS

Figure 5 shows the F_1 score 5a and Variability 5b, respectively for the training set. It can be noted that HOG performs much better when the training set becomes larger. The U-shape of the Variability can be explained by the fact that when few data is used for training the classification will heavily depend on how well those data represents the test set as a whole. Similarly, when the test set is small, the classification will heavily depend on whether if it is difficult or not. In any case, the MoSTiF outperforms HOG and generally have lesser Variability.

Finally the Figure 6 and shows the F_1-score 6a and Variability 6b, respectively for the validation set. This time MoSTiF generally outperforms HOG and the U-shape is not visible in the Variability graph since the validation set is kept fixed. Using all learning data gives a validation F_1 score of 0.995 and a Variability of $9.116 \cdot 10^{-4}$.



(a) F_1-score

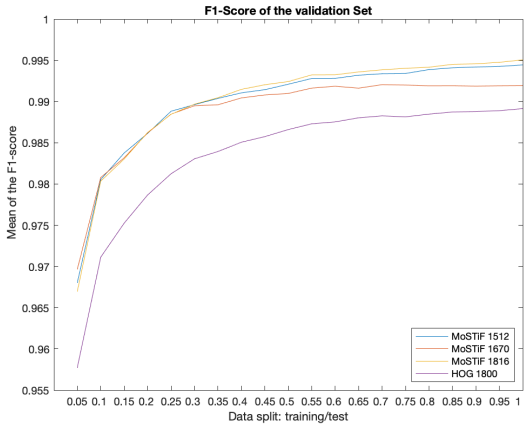


(b) Variability

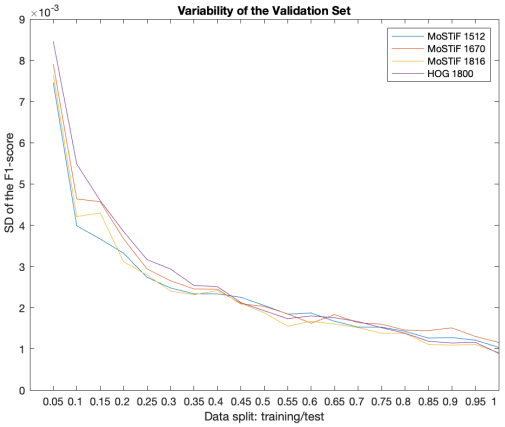
Figure 5: F_1-score and Variability (y-axis) for the test set, for 200 random runs by varying the split of data into a training and test set with varying sizes.

5.1 PCA Compression

In this section we analyse how to make faster training and classification by training on a fixed number of principal components obtained from the features in the training dataset, instead of using the full length of the training features [HM18]. This lossy compression is achieved by applying PCA on the matrix of features in order to reduce the dimensionality into a smaller number of principal components. The first Principal Component will capture the maximum variance in the features. The second will find variance that is incremental to the first, while still being orthogonal to the first. This process is repeated to find all the principal components. In fact, PCA can only produce as many principal components as there are features in the training dataset. Since each successive principal component captures the variance that is left after its preceding component, the components will be less discriminative and at some point they can be discarded without lowering

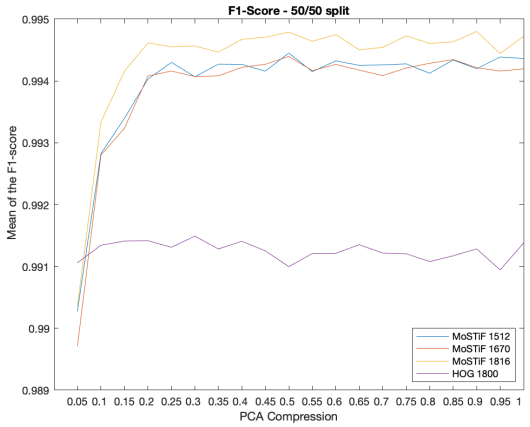


(a) F_1-score

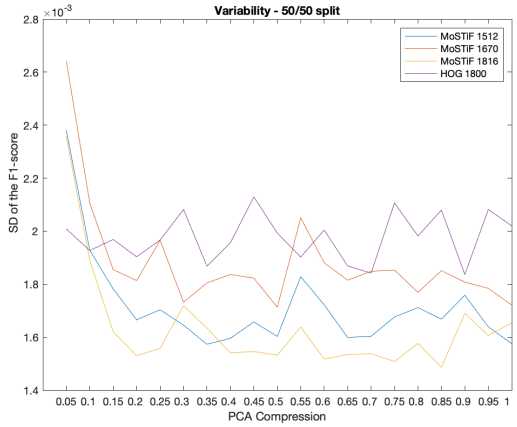


(b) Variability

Figure 6: F_1-score and Variability for the validation set, for 200 random runs by varying the split of data into a training and test set with varying sizes.



(a) F_1-score (y-axis) for different amounts of compression (x-axis).



(b) variability (y-axis) for different amounts of compression (x-axis).

Figure 7: F_1-score and Variability for different amounts of compression for 200 random 50-50 splits of the data.

the F_1-score. In fact it might even make it higher since this procedure will denoise the data, by capturing the main signal in the data and hereby omitting the noise.

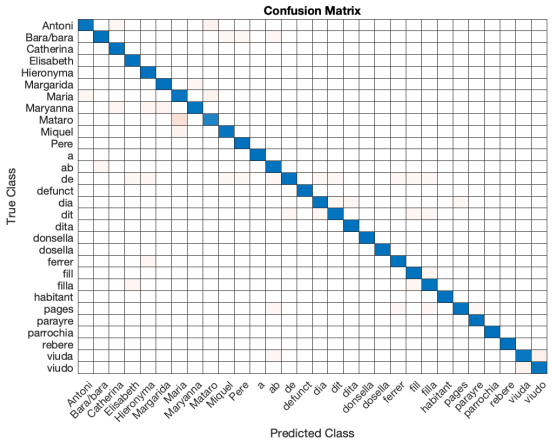
In practice, a number of the first components from the matrix obtained by PCA are kept and the rest are discarded. This matrix is subsequently multiplied to all data (training, validation and test) in order to reduce the dimensionality. This can actually be seen as a neural net applied to the feature vectors, which extracts the most important features in the data, since matrix multiplication is exactly what neurons do. The resulting features are then normalised before being used for training and inference.

Figure 7 shows both the F_1-score 7a and the Variability 7b for different amounts of compression for 200 random 50-50 splits of the data, i.e. while keeping the same validation set as before, all the remaining data is used for the learning. Interestingly all MoSTiF features perform slightly better when compressed down to 50% of its original size than using the full length. The MoSTiF 1816 even performs about just as well (99.99%) for only 20% of its original length, i.e. 334 elements long, and the Variability is even becoming lower.

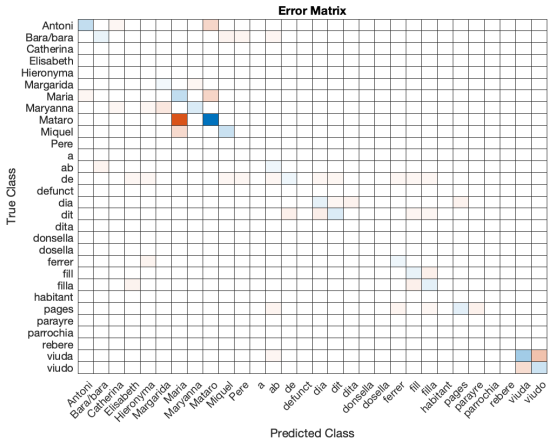
6 DISCUSSION

Figure 8 shows the confusion matrix in 8a, which is not so informative when the accuracy is rather high overall. Here the learning set is varied (50% of all available data) while the test set is kept fixed (the remaining 50%). A better view of where the classification do go wrong can be achieved by showing the errors instead, which can be done by computing a new diagonal, taking 1 minus the old diagonal, as shown in 8b.

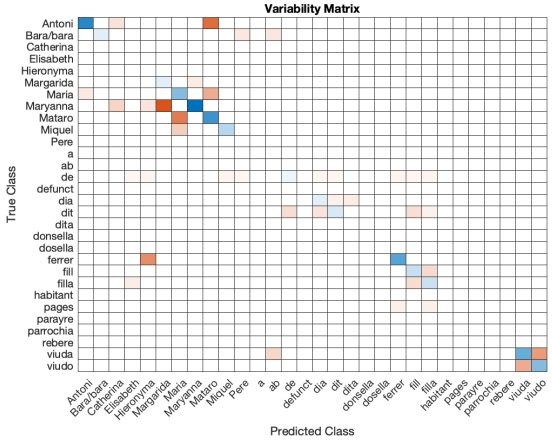
Another way to show where the classification goes wrong is to look at the Variability. This can be done by computing the SD of the F_1-score from the confusion matrices for all runs as shown in 8c. Note, that if there are outliers that always are misclassified, then they will



(a) Confusion matrix for HOG with 50% learning data. Mean over 200 runs.

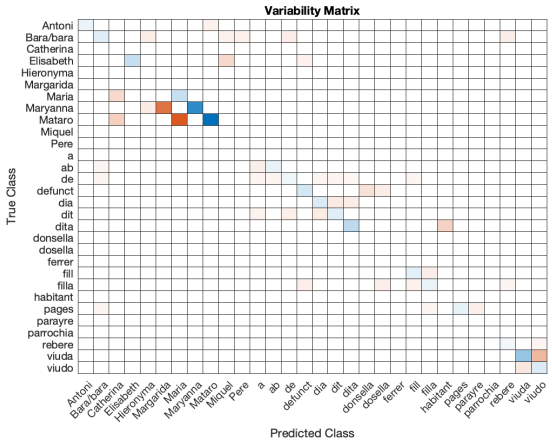


(b) Error matrix for HOG with 50% learning data. Mean over 200 runs.

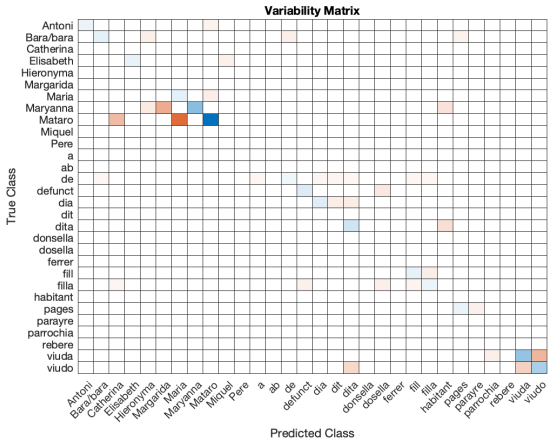


(c) Variability matrix for HOG with 50% learning data. Standard deviation over 200 runs.

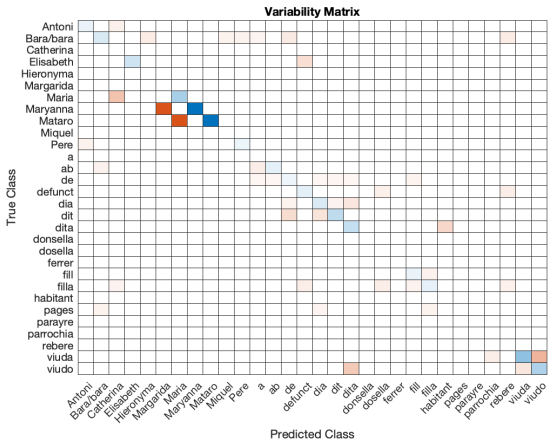
Figure 8: Confusion, Error and Variability matrices. When confusion is low, both the Error and variability matrices tells more about when classifications goes wrong, as the fluctuations become apparent.



(a) Variability matrix for MoSTiF 1512 with 50% learning data. Standard deviation over 200 runs.



(b) Variability matrix for MoSTiF 1670 with 50% learning data. Standard deviation over 200 runs.



(c) Variability matrix for MoSTiF 1816 with 50% learning data. Standard deviation over 200 runs.

Figure 9: Comparison of Variability Matrices for the three different MoSTiF features. Words with high variability are more dependent on the set for learning.

appear in the error matrix but not in the variability matrix, since they do not vary. Hence, the Variability matrix will only show which classes that depends on the actual set for learning, and therefore it will point out which classes that would need more learning data to become more stable.

Figure 9 shows the variability matrices for the three different MoSTiF feature vectors proposed. While comparing the three, one can note that some words are varying regardless of feature vector being used, which indicates that more learning data is necessary for those words. Some vary for only one of the vectors at a time, which indicates that an ensamble of classifiers, using different feature vectors, could be used to more correctly classify those words.

7 CONCLUSION

The MoSTiF feature descriptor turns out to be a better choice than HOG for EPSC and word recognition. Of the three different partitionings examined, the MoSTIF 1816 generally gives better F_1-score, lower Variability and performs better than the others when being compressed. However, since only one dataset was tested, it is not said that the results generalise to any kind of data. Nevertheless, it works very well for the task of word recognition together with EPSC. Furthermore, by using only about 2 subspaces per class, performing only 6 projections per subspace, and being able to compress the features down to only 20% of its original size, the computational cost for inference is indeed very low.

8 ACKNOWLEDGMENTS

This work has been partially supported by the Riksbankens Jubileumsfond (Dnr NHS14-2068:1). The computations were performed on resources provided by SNIC through UPPMAX under project SNIC 2020/15-177. The author wish to thank Raphaela Heil and Ekta Vats for fruitful discussions in the development of the ideas presented. The dataset used is publicly available at `https://andershast.com/datasets/`

9 REFERENCES

[ADRS*19] Arrieta A. B., D'iaz-Rodr'iguez N., Ser J. D., Bennetot A., Tabik S., Barbado A., Garc'ia S., Gil-L'opez S., Molina D., Benjamins R., Chatila R., Herrera F.: Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *ArXiv abs/1910.10045* (2019).

[BETVG08] Baya H., Essa A., Tuytelaarsb T., Van Goola L.: Speeded-up robust features (surf). *Computer vision and image understanding 110*, 3 (2008), 346–359.

[BHB16] Barrera T., Hast A., Bengtsson E.: A chronological and mathematical overview of digital circle generation algorithms : Introducing efficient 4- and 8-connected circles. *International Journal of Computer Mathematics 93*, 8 (2016), 1241–1253.

[BJ21] Buchholz T., Jug F.: Fourier image transformer. *CoRR abs/2104.02555* (2021).

[CATvS17] Cohen G., Afshar S., Tapson J., van Schaik A.: EMNIST: an extension of MNIST to handwritten letters. *CoRR abs/1702.05373* (2017).

[CBK*18] Clanuwat T., Bober-Irizar M., Kitamoto A., Lamb A., Yamamoto K., Ha D.: Deep learning for classical japanese literature. *CoRR abs/1812.01718* (2018).

[CHTT96] Carbon M., Hallin M., Tat Tran L.: Kernel density estimation for random fields: the l 1 theory. *Journal of nonparametric Statistics 6*, 2-3 (1996), 157–170.

[CM02] Comaniciu D., Meer P.: Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 5 (May 2002), 603–619.

[CPC19] Carvalho D. V., Pereira E. M., Cardoso J. S.: Machine learning interpretability: A survey on methods and metrics. *Electronics 8*, 8 (Jul 2019), 832.

[CWW15] Chen C., Wang D.-H., Wang H.: Scene character recognition using pcanet. In *Proceedings of the 7th International Conference on Internet Multimedia Computing and Service* (New York, NY, USA, 2015), ICIMCS '15, Association for Computing Machinery.

[DKMJ18] Dutta K., Krishnan P., Mathew M., Jawahar C.: Improving cnn-rnn hybrid networks for handwriting recognition. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)* (2018), IEEE, pp. 80–85.

[DT05] Dalal N., Triggs B.: Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (2005), vol. 1, IEEE, pp. 886–893.

[FH75] Fukunaga K., Hostetler L.: The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory 21*, 1 (January 1975), 32–40.

[GDDM14] Girshick R., Donahue J., Darrell T., Malik J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2014), pp. 580–587.

[GSC*19] Gunning D., Stefik M., Choi J., Miller T., Stumpf S., Yang G.-Z.: Xai—explainable artificial intelligence. *Science Robotics 4*, 37 (2019).

[HL20] Hast A., Lind M.: Ensembles and cascading of embedded prototype subspace classifiers. *Journal of WSCG 28*, 1/2 (2020), 89–95.

[HLV19] Hast A., Lind M., Vats E.: Embedded prototype subspace classification : A subspace learning framework. In *The 18th International Conference on Computer Analysis of Images and Patterns (CAIP)* (2019), Lecture Notes in Computer Science, pp. 581–592.

[HM18] Hernandez W., Mendez A.: Application of principal component analysis to image compression. In *Statistics*, GÃ¶ksel T., (Ed.). IntechOpen, Rijeka, 2018, ch. 7.

[HSSK18] Hast A., Sablina V. A., Sintorn I.-M., Kylberg G.: A fast fourier based feature descriptor and a cascade nearest neighbour search with an efficient matching pipeline for mosaicing of microscopy images. *Pattern Recognition and Image Analysis 28*, 2 (2018), 261–272.

[HV18] Hast A., Vats E.: Radial line fourier descriptor for historical handwritten text representation. *Journal of WSCG 26*, 1 (2018), 31–40.

[HV21] Hast A., Vats E.: Word recognition using embedded prototype subspace classifiers on a new imbalanced dataset. *Journal of WSCG 29*, 1-2 (2021), 39–47.

[HW79] Hartigan J. A., Wong M. A.: Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 28*, 1 (1979), 100–108.

[JJL07] Ji P., Jin L., Li X.: Vision-based vehicle type classification using partial gabor filter bank. In *2007 IEEE International Conference on Automation and Logistics* (2007), pp. 1037–1040.

[KDJ18] Krishnan P., Dutta K., Jawahar C.: Word spotting and recognition using deep embedding. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)* (2018), IEEE, pp. 1–6.

[KLR*77] Kohonen T., Lehtiö P., Rovamo J., Hyvärinen J., Bry K., Vainio L.: A principle of neural associative memory. *Neuroscience 2*, 6 (1977), 1065 – 1076.

[KO76] Kohonen T., Oja E.: Fast adaptive formation of orthogonalizing filters and associative memory in recurrent networks of neuron-like elements. *Biological Cybernetics 21*, 2 (Jun 1976), 85–95.

[Koh82] Kohonen T.: Self-organized formation of topologically correct feature maps. *Biological Cybernetics 43*, 1 (Jan. 1982), 59–69.

[Koh95] Kohavi R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2* (San Francisco, CA, USA, 1995), IJCAI'95, Morgan Kaufmann Publishers Inc., pp. 1137–1143.

[Kri19] Krishnan M.: Against interpretability: a critical examination of the interpretability problem in machine learning. *Philosophy & Technology* (2019).

[KRMV76] Kohonen T., Reuhkala E., Mäkisara K., Vainio L.: Associative recall of images. *Biological Cybernetics 22*, 3 (Sep 1976), 159–168.

[KW96] Kohavi R., Wolpert D.: Bias plus variance decomposition for zero-one loss functions. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning* (San Francisco, CA, USA, 1996), ICML'96, Morgan Kaufmann Publishers Inc., pp. 275–283.

[Laa07] Laaksonen J.: *Subspace classifiers in recognition of handwritten digits*. G4 monografiaväitöskirja, Helsinki University of Technology, 1997-05-07.

[LCB10] LeCun Y., Cortes C., Burges C.: Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist 2* (2010).

[LCZ*18] Luan S., Chen C., Zhang B., Han J., Liu J.: Gabor convolutional networks. *IEEE Transactions on Image Processing 27*, 9 (2018), 4357–4366.

[Low04] Lowe D. G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (2004), 91–110.

[MH08] Maaten L. v. d., Hinton G.: Visualizing data using t-sne. *Journal of machine learning research 9*, Nov (2008), 2579–2605.

[MH18] McInnes L., Healy J.: UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints* (Feb. 2018).

[MHWS17] Matuszewski D. J., Hast A., Wählby C., Sintorn I.-M.: A short feature vector for image matching: The log-polar magnitude feature descriptor. *PLOS ONE 12*, 11 (11 2017), 1–21.

[MNR92] Mehrotra R., Namuduri K., Ranganathan N.: Gabor filter-based edge detection. *Pattern Recognition 25*, 12 (1992), 1479–1494.

[OB21] Opitz J., Burst S.: Macro f1 and macro f1, 2021.

[OK88] Oja E., Kohonen T.: The subspace learning algorithm as a formalism for pattern recognition and neural networks. In *IEEE 1988 International Conference on Neural Networks* (July 1988), vol. 1, pp. 277–284.

[RFS*13] Romero V., Fornés A., Serrano N., Sánchez J. A., Toselli A. H., Frinken V., Vidal E., Lladós J.: The ESPOSALLES database: An ancient marriage license corpus for off-line handwriting recognition. *Pattern Recognition 46*, 6 (2013), 1658–1669.

[SF16] Sudholt S., Fink G. A.: Phocnet: A deep convolutional neural network for word spotting in handwritten documents. In *ICFHR* (2016), IEEE Computer Society, pp. 277–282.

[Sha18] Shapshak P.: Artificial intelligence and brain. *Bioinformation 14*, 1 (2018), 38.

[SSTF*15] Simo-Serra E., Trulls E., Ferraz L., Kokkinos I., Fua P., Moreno-Noguer F.: Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)* (USA, 2015), ICCV '15, IEEE Computer Society, pp. 118–126.

[WLK*67] Watanabe W., Lambert P. F., Kulikowski C. A., Buxto J. L., Walker R.: Evaluation and selection of variables in pattern recognition. In *Computer and Information Sciences* (1967), Tou J., (Ed.), vol. 2, New York: Academic Press, pp. 91–122.

[WP73] Watanabe S., Pakvasa N.: Subspace method in pattern recognition. In *1st Int. J. Conference on Pattern Recognition, Washington DC* (1973), pp. 25–32.

[XRV17] Xiao H., Rasul K., Vollgraf R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR abs/1708.07747* (2017).

[ZK15] Zagoruyko S., Komodakis N.: Learning to compare image patches via convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015).

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Color-dependent pruning in immersive video coding

Dawid Mieloch [1] dawid.mieloch@put.poznan.pl
Adrian Dziembowski [1]
Marek Domański [1]
Gwangsoon Lee [2]
Jun Young Jeong [2]

[1] Institute of Multimedia Telecommunications, Poznań University of Technology
Polanka 3, 61-131 Poznań, Poland

[2] Electronics and Telecommunications Research Institute
Daejeon, Republic of Korea

ABSTRACT

This paper presents the color-dependent method of removing inter-view redundancy from multiview video. The pruning of input views decides which fragments of views are redundant, i.e., do not provide new information about the three-dimensional scene, as these fragments were already visible from different views. The proposed modification of the pruning uses both color and depth and utilizes the adaptive pruning threshold which increases the robustness against the noisy input. As performed experiments have shown, the proposal provides significant improvement in the quality of encoded multiview videos and decreases erroneous areas in the decoded video caused by different camera characteristics, specular surfaces, and mirror-like reflections. The pruning method proposed by the authors of this paper was evaluated by experts of the ISO/IEC JTC1/SC29/WG 11 MPEG and included by them in the Test Model of MPEG Immersive Video.

Keywords

Immersive video coding, multiview compression, virtual reality.

1. INTRODUCTION

In an immersive video, the viewer can interactively change his/her position in the three-dimensional scene, allowing virtual traversing, e.g., using a virtual reality set [Dom17]. Typically, in order to generate a virtual view, some kind of three-dimensional representation of an acquired scene has to be utilized. The most widespread is the multiview video plus depth representation (MVD) [Mül11]. The so-called depth maps are used to store, in the form of an additional grayscale video, the distance from the camera to the 3D point for each pixel of the corresponding video.

Immersive video applications are gaining recently a large interest both from the video processing researchers and from the standardization community, as dedicated compression standards are emerging [Boy21]. It makes the introduction of such kinds of services to possible consumers much easier.

Independent compression of multiple views and corresponding depth maps (e.g., using HEVC [Sul12]) results in high bitrates [Dom21]. Instead, the compression of the immersive video should take advantage of the inter-view redundancy existing in the multiview representation. For example, input representation may consist of multiple typical camera-acquired videos with vastly overlapping fields of views, or it may consist of a few overlapping omnidirectional (360-degree) videos. The redundancy resulting from the spatial overlap between input views can be used to decrease the size of data required to fully represent the whole three-dimensional scene.

Basic multiview encoders (such as MPEG's Multiview Video Coding – MVC [Mer06] or Multiview extension of High Efficiency Video Coding – MV-HEVC [Tec16]) usually utilize the inter-view prediction, based on motion vectors estimated between neighboring frames (similarly as it is done in the temporal domain in typical inter-frame prediction). However, in this approach, textures and depth maps are encoded independently, so the searching of motion vectors does not utilize the information about three-dimensional geometry of an encoded scene. This improvement was introduced in 3D-HEVC in the form of the depth-based inter-view prediction [Tec16]. While both described types of prediction highly decrease the bitrate of encoded videos (more than 40% reduction in comparison with simulcast encoding), it still requires allocating a part of the bitstream for residual data left after the prediction. Moreover, as the prediction does not fully eliminate the inter-view redundancy, the pixel-rate (understood as the number of pixels to be decoded per second to produce the requested view) is not decreased.

The current state-of-the-art technology for immersive video compression was developed by the MPEG ISO/IEC group under the MPEG Immersive video name (MIV) [ISO22]. The foundations of this standard were built on technologies presented by proponents of Call for Proposals for 3DoF+ video coding [Dom19], [Fle19]. Most of the proposals followed a similar core idea, that few base views that gather most of the information of the scene should be encoded in their entirety, while supplementary information (visible from the remaining viewpoints) can be transmitted in the form of a mosaic of patches, collectively forming an atlas. Fig. 1 presents an example of this representation: an atlas with base views is presented on the left, an atlas with patches is in the middle, while on the right two corresponding atlases with depth maps can be seen. These atlases are then encoded using a typical video encoder, e.g. HEVC [Sul12], while in the decoder, the atlases are used to synthesize the requested viewpoint for a final viewer. The full description of MPEG Immersive video can be found in [Boy21].

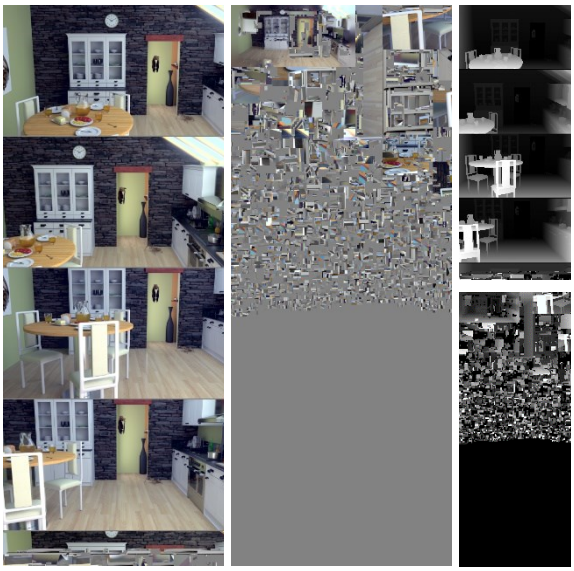


Figure 1. Atlases for sequence Kitchen [Boi18]: two texture atlases and two depth atlases (reduced resolution).

In order to decide which information should be packed into atlases, the so-called pruning process is utilized. It is based on the recognition of the inter-view redundant parts of input views and removing them from further processing (the details are described in Section 2), as these areas do not provide new information about the three-dimensional scene – these fragments were already visible from different views.

This paper introduces the pruning process improvement in which the color information is utilized in a way that minimizes the number of patches required to properly encode the non-Lambertian surfaces present in a 3D scene. The organization of the paper is as follows: Section 2 shows the description of the basic pruning process and the details of the proposed color-dependent pruning method; Section 3 includes the results of the comparison of the proposed method with the basic depth-dependent pruning. In the end, Section 4 summarizes the paper and includes conclusions drawn from the performed experiments.

2. INTER-VIEW PRUNING OF MULTIVIEW VIDEOS

Overview of the pruning process

The simplified process of pruning is shown in Fig. 2. First of all, basic views are inserted into the pruning graph (as root nodes). Then, all pixels of basic views are projected (using the depth information) to each additional view. After creating the pruning mask for each additional view, the additional view with the maximum number of preserved pixels is selected (to prefer larger patches). This selected additional view is then added to the pruning graph (as a child node of other nodes in the graph). The projection of all preserved pixels of the selected view to remaining additional views is repeated and the pruning mask is iteratively updated for each remaining additional view.

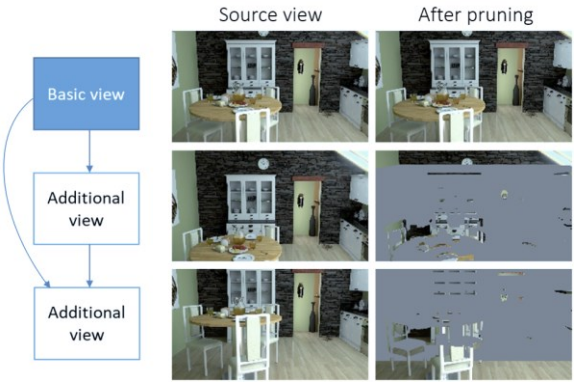


Figure 2. The idea of source views pruning.

The described process is dependent on the condition which has to be met to decide if the element should be pruned from an input view. Initially, the pruning process was based only on depth information, i.e., if the difference between the depth of the transferred point and the point corresponding to it was higher than a depth pruning threshold (10%), then the corresponding point was marked as redundant and removed. This approach eliminated the problem of noisy input views, for which it is much harder to determine the similarity between the points in neighboring views when the decision is based on color. On the other hand, when non-Lambertian surfaces are present in the scene, the depth-based pruning was leaving only one instance of such surface (only from one of the views), because the depth of a reflective surface is the same in all views, so specular reflections are irreversibly lost in the pruning.

Removal of inter-view redundancy based only on depth of a point is also encountered in methods of producing the multiview layered depth image (LDI) [Anj17] and compression of meshes [Tan18] and voxels [Käm16]. Although LDI representation is used for efficient synthesis of virtual views rather than for compression purposes, abovementioned methods show that depth-based type of inter-view redundancy check was widely considered to be state-of-the-art in multiview processing.

The proposed color-dependent pruning

In the approach proposed by the authors of this paper, two types of information are taken into account: depth and texture. Depth information is analyzed in the same way as described above, while color information is analyzed in a point-to-block comparison (Fig. 3). In this example, a pixel from view v0 is reprojected to v1. Depth similarity is being checked only for the colocated pixel (dark blue). The color of the pixel marked in orange is compared to the color of all pixels in the 3×3 neighborhood of the colocated one.

A similar approach was proposed earlier in [Mie21], where the point-to-block matching was used as an inter-view similarity metric for depth estimation purposes. This metric was shown to work especially well for highly compressed input views or when the amount of noise present in a multi-view sequence is significant. Even though the pruning is always performed on uncompressed views, the decreased influence of noise on inter-view matching is desirable and preferable in this part of multiview processing. Color-based matching of objects and points have also been shown to be efficient in single video temporal tracking [Ker10].

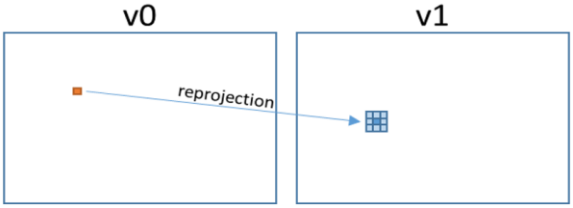


Figure 3. The idea of point-to-block similarity measurement used in color-dependent pruning.

In the proposal, if the minimum color error within a block is lower than a threshold (and the depth-based condition is also met), the pixel of view v1 is being pruned. Using of the 3×3 neighborhood instead of a larger one was based on results from [Mie21] which showed that such block size is the best for inter-view matching in multiview videos. Having this variable fixed, during the preliminary experiments, the pruning threshold was set to 4% of the bit depth of color (i.e., 40 for 10 bits per sample views).

As can be seen in Fig. 4, the same area can reflect the light differently when acquired with the camera facing different directions. The atlas generated with and without proposed color-dependent block-based pruning is presented in Fig. 5. As shown, the proposed solution allows preserving regions with different texture/lighting conditions, which were otherwise pruned.

Moreover, because of block-based characteristics, the proposed pruning is less sensitive to noise and only slightly increases the non-pruned area for such type of content. Fig. 6 presents two consecutive frames of ClassroomVideo sequence and the difference between them (to illustrate the amount of noise), while Fig. 7 presents the difference in the atlas when it is generated with color-dependent pruning.

Nevertheless, in order to further decrease the influence of noise, we included further enhancement to the proposed pruning method, which allows adapting to sequence characteristics.

The threshold for color-based pruning should be increased for noisy sequences to reduce redundancy in atlases. On the other hand, for sequences with negligible noise, the pruning threshold should be smaller to allow preserving also fragments of the scene with slight lighting inconsistencies.

In the proposed solution, the fixed pruning threshold is multiplied by a global inter-view luma standard deviation. This value is calculated for the first frame of each encoded group of pictures (GoP) as a standard deviation of a set A, which contains luma differences between inter-view corresponding pixels. GoP can be changed in the configuration of MIV encoder, by default it is equal to 32 to match the GoP used in HEVC encoder. Calculating this threshold for a whole GoP is sufficient as in MIV the results of pruning are in the end gathered for these frames.

In order to populate set A, first of all, all pixels are reprojected between all pairs of input views. For each pixel, the luma of the pixel is compared with the luma of all pixels in the 3×3 neighborhood of the pixel from another input view. If the smallest difference in this neighborhood is 0, then the inter-view correspondent pixels were found, so the luma difference between the reprojected pixel and the center of the co-located block is included in set A.

Figure 4. Two views of multiview test sequence Chess [Ilo19].

Figure 5. Atlas of patches generated for Chess sequence: depth-dependent pruning (left) vs. proposal (right).

Figure 6. Fragments of two consecutive frames of ClassroomVideo sequence and the difference between them (on the right).

Figure 7. Atlas of patches generated for ClassroomVideo sequence: depth-dependent pruning (left) vs. proposal (right).

3. EXPERIMENTAL RESULTS

Methodology and design of experiments

To evaluate the proposed color-dependent pruning, the method was implemented in TMIV 5 [MPEG20c]. Test Model for Immersive Video is implemented as a C++ project which is publicly available. All changes which are being added to the standard and have to be implemented in TMIV, are earlier accepted by its software coordinator to avoid adding low-quality code to the repository.

Two experiments were performed. The first one shows the results for encoding with color-dependent pruning without taking into account the noise characteristics, therefore, for the fixed pruning threshold. The latter experiment evaluates the encoding efficiency with an adaptive threshold automatically calculated for each encoded test sequence.

11 multiview sequences from MIV Common Test Conditions (CTC) [MPEG20a] were used in experiments. Short characteristics of sequences are presented in Table 1, while an example of input view from each sequence was shown in Figs. 8 and 9.

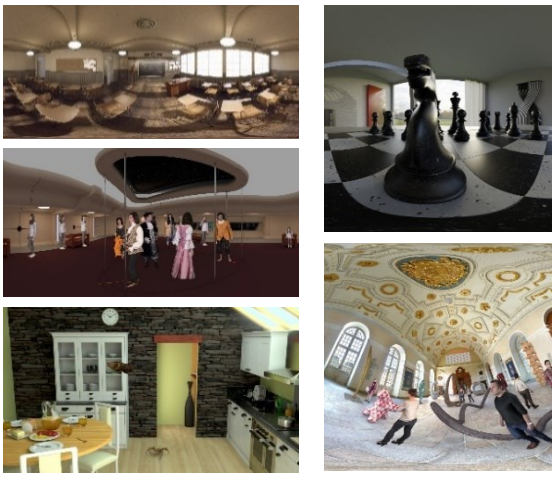


Figure 8. Computer-generated sequences. Left column: ClassroomVideo, Hijack, Kitchen; right column: Chess, Museum.

Figure 9. Natural sequences. Left column: Carpark, Street, Frog; right column: Hall, Fencing, Painter.

Test sequences were encoded with 5 different rate points (RP) using the default configuration of the TMIV encoder, described in the CTC document [MPEG20a]. The final performance was measured

using two objective quality metrics: WS-PSNR [Sun17], and IV-PSNR [MPEG20b]. The first of these metrics is based on the peak signal-to-noise ratio but it was adapted to take into account the spherical projection of 360-degree videos, while the latter is adapted to measure the distortion introduced by the virtual view synthesis process.

The results are shown as Bjøntegaard delta (BD-rate [Bjo01]) change for low and high bitrates. The Bjøntegaard delta shows the percentage change in the bitrate required to achieve the same quality for two tested techniques. It was calculated both for the four smallest QPs (high bitrates – High-BR BD-rate) and for the four largest ones (low bitrates – Low-BR BD-rate). A gain or a loss larger than 3% is indicated by a green or red cell respectively.

Sequence name	Views	Type	Resolution	Source	Pruning-related sources of difficulty
Carpark	9	NC/Persp.	1920×1088	[Mie20]	Reflections on cars
Chess	10	CG/ERP	2048×2048	[Ilo19]	Reflections on the floor
Classroom Video	16	CG/ERP	4096×2048	[Kro18]	Highly noticeable noise
Fencing	10	NC/Persp.	1920×1080	[Dom16]	Not inter-view consistent color characteristics
Frog	13	NC/Persp.	1920×1080	[Sal18]	Not inter-view consistent color characteristics
Hall	9	NC/Persp.	1920×1088	[Mie20]	Reflections on the floor
Hijack	10	CG/ERP	4096×2048	[Dor18]	Reflections on clothes
Kitchen	25	CG/Persp.	1920×1080	[Boi18]	Reflections and transparency of kitchen objects
Museum	24	CG/ERP	2048×2048	[Dor18]	-
Painter	16	NC/Persp.	2048×1088	[Doy18]	-
Street	9	NC/Persp.	1920×1088	[Mie20]	Reflections on cars

Table 1. Test sequences used in experiments: ERP – Equirectangular Projection, CG – Computer-Generated, NC – Natural Content.

The evaluation of color-dependent pruning with a fixed threshold

The results for encoding with color-dependent pruning with a fixed threshold in comparison with the pruning dependent only on depth are presented in Table 2. As it can be observed, for most test sequences, the proposed pruning method significantly decreases the BD-rate, i.e., the same quality of a final decoded image can be obtained for reduced bitrate. On average, the bitrate is decreased by 20% for high bitrates.

Fig. 10 shows the plot of the peak signal to noise ratio (PSNR) for different bitrates of encoded Carpark sequence, for which the results are the most similar to the averaged ones. It can be seen that the bitrate was slightly increased for all 5 rate points, what is the result of the increased number of patches observed in atlases. However, adding these patches caused a significant increase in the quality of the encoded video, what, in the end, results in increased compression efficiency.

Sequence	High-BR BD rate Y-PSNR	Low-BR BD rate Y-PSNR	High-BR BD rate IV-PSNR	Low-BR BD rate IV-PSNR
Carpark	-19.8%	-8.6%	-17.2%	-6.7%
Chess	-64.0%	-46.9%	-61.8%	-47.8%
ClassroomVideo	6.8%	15.4%	9.7%	15.3%
Fencing	---	---	---	-39.4%
Frog	-32.3%	11.5%	-23.1%	14.7%
Hall	-51.1%	-33.5%	-40.8%	-29.4%
Hijack	-19.8%	-10.2%	-24.9%	-15.9%
Kitchen	-33.1%	-9.0%	-35.1%	-14.0%
Museum	0.7%	1.5%	0.2%	1.2%
Painter	1.3%	2.0%	1.7%	2.0%
Street	-10.9%	-2.5%	0.5%	4.3%
Average	**-20.2%**	**-7.3%**	**-17.3%**	**-10.5%**

Table 2. BD-rate savings of MIV encoding with color-dependent pruning for a fixed threshold over MIV encoding with depth-dependent pruning.

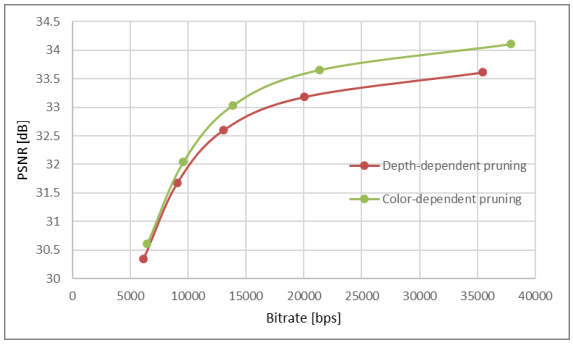


Figure 10. PSNR values for Carpark sequence encoded for 5 rate points using MIV encoder with depth-dependent pruning and with MIV encoder with color-dependent pruning.

As it was shown in Table 3, the proposal increases the runtime of MIV encoding and decoding, what is the result of the increased number of patches included in atlases. The runtime of HEVC encoding was increased insignificantly.

Sequence	MIV encoding	HEVC encoding	MIV decoding
Carpark	100.5%	107.5%	103.0%
Chess	109.4%	103.4%	109.6%
ClassroomVideo	103.4%	95.0%	102.5%
Fencing	101.8%	107.2%	101.4%
Frog	113.3%	108.2%	115.3%
Hall	101.2%	93.2%	102.8%
Hijack	106.9%	115.0%	109.3%
Kitchen	110.7%	103.1%	109.2%
Museum	112.3%	105.0%	114.2%
Painter	101.2%	104.7%	102.0%
Street	103.4%	94.9%	105.5%
Average	106.9%	101.9%	107.7%

Table 3. Runtime ratio of MIV and HEVC encoding and MIV decoding with color-dependent pruning for a fixed threshold over depth-dependent threshold.

The evaluation of color-dependent pruning with an adaptive threshold

The adaptive pruning threshold is based on the global inter-view luma standard deviation, calculated at the beginning of the encoding. The values of standard deviations for tested sequences and the resulting pruning thresholds can be found in Table 4.

Table 5 presents the efficiency of adaptive color-dependent pruning in comparison with fixed-threshold color-dependent pruning. The introduction of an adaptive threshold further improves the compression efficiency for almost all test sequences.

Fig. 11 presents the fragments of the final encoded views for the encoder with depth-dependent pruner (left) and adaptive color-dependent pruner (right). These results prove that the proposal increases the quality of video presented to a viewer by eliminating the errors caused not only by specular surfaces (see Chess sequence), and mirror-like reflections (Hall) but also by different camera characteristics (Fencing). The possibility of increasing the quality in cases where camera color characteristics are not matched can be seen as surprising, as the presented color-based pruning is based only on luma value, however, results presented in [Dzi21] show that inter-view and temporal fluctuations of luma and chromas are correlated in natural sequences.

Detailed results for Painter indicate that the PSNR was very slightly increased (0.03 dB), but the bitrate was increased by 2.5%. This sequence provided very high quality even without the proposal and does not include any reflective surfaces, what explains lower quality, as with the proposal we increased the amount of sent data and increased quality to a very minor degree.

Sequence	Calculated standard deviation	Calculated pruning threshold for 10 bps video
Carpark	0.9337	38
Chess	0.3891	16
ClassroomVideo	0.9555	39
Fencing	0.8132	33
Frog	1.6505	68
Hall	0.2664	11
Hijack	0.2084	9
Kitchen	0.8698	36
Museum	0.8711	36
Painter	0.5670	23
Street	0.8560	35

Table 4. The average global sequence inter-view luma standard deviation and calculated pruning threshold for used dataset.

Sequence	High-BR BD rate Y-PSNR	Low-BR BD rate Y-PSNR	High-BR BD rate IV-PSNR	Low-BR BD rate IV-PSNR
Carpark	-0.7%	0.0%	0.0%	0.3%
Chess	-37.3%	-4.9%	-25.4%	-0.3%
ClassroomVideo	0.4%	0.6%	1.1%	1.0%
Fencing	-16.8%	-5.3%	-6.9%	-0.7%
Frog	-5.5%	-8.3%	-7.4%	-9.4%
Hall	-34.4%	-12.4%	-10.9%	1.4%
Hijack	-11.7%	5.0%	-18.0%	3.2%
Kitchen	-20.4%	-10.8%	-18.7%	-10.5%
Museum	0.5%	0.5%	0.3%	0.4%
Painter	3.2%	5.2%	4.1%	5.6%
Street	-9.4%	-2.8%	-7.9%	-2.6%
Average	**-12.0%**	**-3.0%**	**-8.1%**	**-1.0%**

Table 5. BD-rate savings of MIV encoding with color-dependent pruning for an adaptive threshold over MIV encoding with color-dependent pruning for a fixed threshold.

Table 6 shows the increase of runtime in comparison with the fixed threshold. As it can be observed, the increase of the runtime is correlated with the adaptive threshold – if the threshold is small, then the time of processing is increased, as the number of patches included in atlases is further increased. However, as indicated by the results of the quality of decoded views, these additional patches are required to achieve high-quality reconstruction in the decoder.

Sequence	MIV encoding	HEVC encoding	MIV decoding
Carpark	96.9%	96.9%	99.8%
Chess	118.8%	118.9%	117.0%
ClassroomVideo	104.4%	101.7%	103.4%
Fencing	94.3%	94.1%	99.0%
Frog	95.5%	95.2%	95.1%
Hall	146.0%	146.3%	148.5%
Hijack	122.9%	123.4%	121.6%
Kitchen	108.6%	107.1%	109.7%
Museum	123.1%	119.8%	123.4%
Painter	144.9%	145.6%	149.9%
Street	106.4%	106.2%	111.0%
Average	124.8%	123.8%	128.1%

Table 6. Runtime ratio of MIV and HEVC encoding and MIV decoding with color-dependent pruning for an adaptive threshold over color-dependent pruning for a fixed threshold.

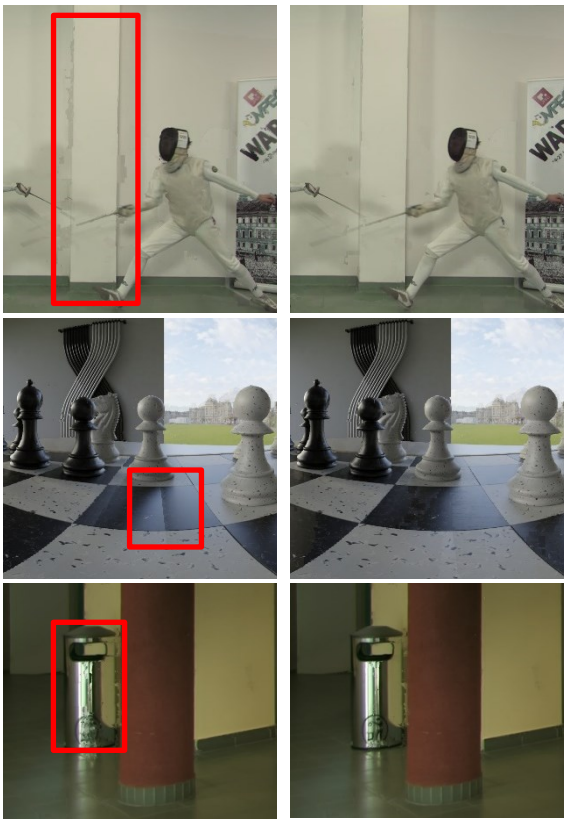


Figure 11. The fragments of decoded views of Fencing, Chess, and Hall sequences for MIV with depth-dependent pruning (left) and MIV with color-dependent pruning (right).

4. CONCLUSIONS AND FUTURE WORKS

This paper presented the color-dependent method of removing inter-view redundancy from multiview video. In order to decide which fragments of views are redundant, i.e., do not provide new information about the three-dimensional scene, as these fragments were already visible from different views, the proposal uses both color and depth. The paper introduces the adaptive pruning threshold which increases the robustness against the noisy input.

The proposed pruning method was shown to provide significant improvement in the quality of videos encoded using a modified Test Model of MPEG Immersive video codec. The largest improvement was due to the elimination of errors caused by different camera characteristics, specular surfaces, and mirror-like reflections, therefore, the proposed method highly increases the efficiency of encoding for complex multiview sequences.

The pruning method proposed by the authors of this paper was evaluated by experts of the ISO/IEC JTC1/SC29/WG 11 MPEG and included by them in the next version Test Model. As works on the new edition of MPEG Immersive video coding standard are starting in mid-2022, the authors of the proposal are planning to evaluate the color-based pruning in this new framework. MIV ed. 2 is planned to include the step of determining the non-Lambertian areas in the scene [MPEG22], so the proposed method can be potentially used for this application.

5. ACKNOWLEDGMENTS

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00207, Immersive Media Research Laboratory).

6. REFERENCES

[Anj17] Anjos R., Madeiras Pereira J., Gaspar J., Fernandes C. Multiview layered depth image. Journal of WSCG, vol. 25. pp.115-122, 2017.

[Bjo01] Bjøntegaard, G. Calculation of average PSNR differences between RD986 curves. ISO/IEC JTC1/SC29/WG11, MPEG/M15378, Austin, TX, 2001.

[Boi18] Boissonade P., and Jung J. Proposition of new sequences for Windowed-6DoF experiments on compression, synthesis, and depth estimation. Document ISO/IEC JTC1/SC29/WG11 MPEG/M43318, Ljubljana, Slovenia, Jul. 2018.

[Boy21] Boyce J., et al. MPEG Immersive Video Coding Standard. Proceedings of the IEEE, vol. 109, no. 9, pp. 1521-1536, Sep. 2021.

[Dom16] Domański M., Dziembowski A., Grzelka A., Mieloch D., Stankiewicz O., and Wegner K. Multiview test video sequences for free navigation exploration obtained using pairs of cameras. Document ISO/IEC JTC1/SC29/WG11, MPEG M38247, 2016.

[Dom17] Domański M., Stankiewicz O., Wegner K. and Grajek T. Immersive visual media – MPEG-I: 360 video, virtual navigation and beyond. 2017 IEEE International Conference on Systems, Signals and Image Processing (IWSSIP), Poznań, pp. 1–9, 2017.

[Dom19] Domański M., et al. Technical description of proposal for Call for Proposals on 3DoF+ Visual prepared by PUT and ETRI. ISO/IEC JTC1/SC29/WG11 MPEG/M47407, Geneva, Switzerland, 2019.

[Dom21] Domański M., Al-Obaidi Y., and Grajek T. Universal Modeling of Monoscopic and Multiview Video Codecs with Applications to Encoder Control. 2021 IEEE International Conference on Image Processing (ICIP), pp. 2144-2148, 2021.

[Dor18] Doré R. Technicolor 3DoF+ Test Materials. Document ISO/IEC JTC1/SC29/WG11 MPEG/M42349, San Diego, CA, USA, Apr. 2018.

[Doy18] Doyen D., Boisson G., and Gendrot R. [MPEG-I Visual] New Version of the Pseudo-Rectified Technicolorpainter Content. Document ISO/IEC JTC1/SC29/WG11 MPEG/M43366, Ljubljana, 2018.

[Dzi21] Dziembowski A., Mieloch D., Różek S., Domański M. Color Correction for Immersive Video Applications. IEEE Access, vol. 9, pp. 75626-75640, 2021.

[Fle19] Fleureau J., et al. Technicolor-Intel Response to 3DoF+ CfP. ISO/IEC JTC1/SC29/WG11 MPEG/M47445, Geneva, Switzerland, 2019.

[Ilo19] Ilola L., Vadakital V.K.M., Roimela K., and Keränen. New test content for Immersive Video – Nokia Chess. Document ISO/IEC JTC1/SC29/WG11 MPEG/M50787, Geneva, Switzerland, Oct. 2019.

[ISO22] Standard ISO/IEC FDIS 23090-12. Information technology – Coded representation of immersive media – Part 12: MPEG Immersive video. 2022.

[Käm16] Kämpe V., et al. Exploiting coherence in time-varying voxel data. Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D '16). pp. 15–21, 2016.

[Ker10] Kerdvibulvech C. Real-time augmented reality application using color analysis. 2010 IEEE Southwest Symposium on Image Analysis & Interpretation (SSIAI), pp. 29-32, 2010.

[Kro18] Kroon B. Test sequence ClassroomVideo. Document ISO/IEC JTC1/SC29/WG11 MPEG/M42415, San Diego, CA, USA, Apr. 2018.

[Mer06] Merkle P., Muller K., Smolic A., and Wiegand T. Efficient Compression of Multi-View Video Exploiting Inter-View Dependencies Based on H.264/MPEG4-AVC. 2006 IEEE International Conference on Multimedia and Expo, pp. 1717-1720, 2006.

[Mie20] Mieloch D., Dziembowski A., and Domański M. [MPEG-I Visual] Natural Outdoor Test Sequences. Document ISO/IEC JTC1/SC29/WG11 MPEG/M51598, Brussels, Jan. 2020.

[Mie21] Mieloch D., Klóska D., Woźniak, M. Point-to-Block Matching in Depth Estimation, International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2021.

[MPEG20a] Common Test Conditions for Immersive Video. Document ISO/IEC JTC1/SC29/WG11 MPEG, N19214, Alpbach, Austria, Apr. 2020.

[MPEG20b] Software manual of IV-PSNR for Immersive Video. Document ISO/IEC JTC1/SC29/WG04 MPEG VC, N0013, Online, Oct. 2020.

[MPEG20c] Test Model 5 for Immersive Video. Document ISO/IEC JTC1/SC29/WG11 MPEG, N19213, Online, Apr. 2020.

[MPEG22] Use cases and requirements for MIV. Edition-2 (final), ISO/IEC JTC1/SC29/WG02 MPEG Technical requirements, N00157, Online, January 2022.

[Mül11] Müller K., Merkle P., and Wiegand T. 3-D Video Representation Using Depth Maps. 2011 Proceedings of the IEEE, vol. 99, no. 4, pp. 643-656, 2011.

[Sal18] Salahieh B., et al. Kermit test sequence for Windowed 6DoF Activities. Document ISO/IEC JTC1/SC29/WG11 MPEG/M43748, Ljubljana, Slovenia, Jul. 2018.

[Sul12] Sullivan G.J., Ohm J., Han W., and Wiegand T. Overview of the High Efficiency Video Coding (HEVC) Standard. IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[Sun17] Sun Y., Lu A., and Yu L. Weighted-to-spherically-uniform quality evaluation for omnidirectional video. IEEE Signal Processing Letters, vol. 24, no. 9, pp. 1408-1412, Sep. 2017.

[Tan18] Tang D., et al. Real-time compression and streaming of 4D performances. ACM Trans. Graph, vol. 37, no. 6, pp 1-11, Dec. 2018.

[Tec16] Tech G., et al. Overview of the Multiview and 3D Extensions of High Efficiency Video Coding. IEEE Transactions on Circuits and Systems for Video Technology, vol. 26, no. 1, pp. 35-49, Jan. 2016.

ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

Design Space of Geometry-based Image Abstraction Techniques with Vectorization Applications

Lisa Ihde
Hasso Plattner Institute,
Digital Engineering Faculty,
University of Potsdam, Germany
lisa.ihde@student.hpi.uni-potsdam.de

Amir Semmo
Digital Masterpieces GmbH,
Potsdam, Germany
amir.semmo@digitalmasterpieces.com

Jürgen Döllner
Hasso Plattner Institute,
Digital Engineering Faculty,
University of Potsdam, Germany
juergen.doellner@hpi.uni-potsdam.de

Matthias Trapp
Hasso Plattner Institute,
Digital Engineering Faculty,
University of Potsdam, Germany
matthias.trapp@hpi.uni-potsdam.de

Figure 1: Result of vectorized pencil hatching using optimized tonal art maps based on the presented approach.

ABSTRACT

The paper presents a new approach of optimized vectorization to generate stylized artifacts such as drawings with a plotter or cutouts with a laser cutter. For this, we developed a methodology for transformations between raster and vector space. Moreover, we identify semiotic aspects of Geometry-based Stylization Techniques (GSTs) and the combination with raster-based stylization techniques. Therefore, the system enables also Fused Stylization Techniques (FSTs).

Keywords: Image stylization, Image processing, Vectorization, Non-photorealistic Rendering, Plotting

1 INTRODUCTION

1.1 Motivation

Artists use different tools and materials for creative expression, e.g., from traditional techniques such as oil painting or pencil hatching to modern digital fabrication. In recent years, research has focused on imitating these techniques to create computer graphics-based images using Non-photorealistic Rendering (NPR) techniques [Kyp13; Dev13]. The resulting images enjoy great popularity and are increasingly used and shared on social media platforms [Sem162].

While the visual quality of these results closely resembles the original techniques, they lack certain qualities during reproduction, e.g., using canvas printing or similar. However, by using fabrication techniques based on pen plotters — a commodity reproduction hardware that enjoys popularity in the maker culture — pencil hatching or stippling can be easily implemented using a variety of real pencils. The capability and functionality of these devices range from professional to hobby level, and are mainly self-built by participant of the maker scene [Mee20]. For the latter, the building instructions are freely available and partially customizable, thus not limited to certain spatial sizes or resolutions.

With respect to this, a pen plotter uses a vectorized image representation to produce a line drawing. Vector graphics add advantages compared to raster images. For example, the resolution of the graphic is then no longer limited and can be scaled up. In addition, connectivity information of graphic primitives facilitate editing processes. Furthermore and due to the geometric representation, vector graphics enable, next to printers or plotters, the use of production or fabrication devices such as laser cutters [Mue15].

1.2 Problem Statement

Mostly limited by input resolution, for a stylized photo (e.g., transformed using a pencil hatching stylization technique [Sem20]) it would be difficult to trace or reconstruction the individual lines or edges when automatically converted to a vector graphic. This challenge exists due to crossing of lines and used textures to represent the pen pressure (cf. Tonal Art Maps (TAMs) [Pra01]). To obtain an optimized and actually usable vector graphic representation, we need to

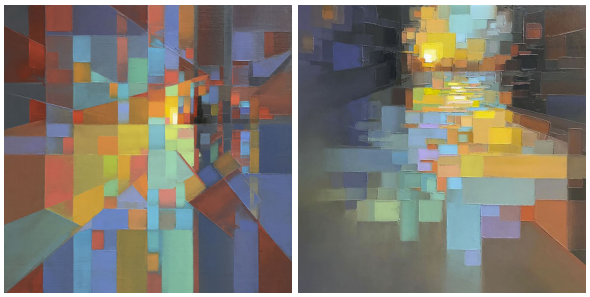


Figure 2: From "Landscapes" by Jason Anderson.

examine how a stylization technique is concretely constructed.

On the one hand, there are raster-based stylization techniques such as watercolor [Bou06], oil-painting [Sem161], Cartoon [Win06], or pencil hatching [Sem20]. On the other hand, there are Geometric Stylization Techniques (GSTs), which denote image stylization techniques that output a number of geometric primitives to represent the stylization result. In addition, there are also mixtures of both techniques, e.g., the artist Jason Anderson and his work on "Landscapes" (Figure 2). His work shows a real-world example of combining geometry-based and raster-based image stylization techniques. We denote such combinations as Fused Stylization Technique (FST). The two classes of stylization techniques can be differentiated of the different data representations of the results, namely vector and raster.

1.3 Approach and Contributions

For the approach, a system generates image stylization results based on a digital photo. Therefore, we develop a general approach of semiotic aspects to classify different geometry-based stylization techniques. The system provides different geometric stylization techniques. In addition, we also investigate the combination with raster-based techniques and how fused stylization techniques can be achieved. This is based on the elaboration of stylization operations between vector-based and pixel-based representation. To summarize, this paper makes the following contributions:

1. It presents the concept and semiotic aspects for Geometric Stylization Techniques (GSTs).

2. It Implementation of framework with various geometry-based stylization techniques, which is demonstrated by different application examples.

3. It presents the concept of fused stylization techniques, which combines geometry-based and image-based stylization techniques.

The remainder of this paper is structured as follows. Section 2 reviews related work and present background to image-based and geometry-based abstraction techniques that represents the basis for our approach. Section 3 presents the design space and fundamental concept of geometry-based image abstraction techniques. Based on this, Section 4 demonstrate it application by means of different examples. Section 5 discusses the presented approach and describes future research directions. Finally, Section 6 concludes this work.

2 BACKGROUND

For the classification of stylization techniques, a comprehensive review exists in the area of NPR [Kyp13]. Kyprianidis et al. presented a taxonomy of artistic stylization techniques for images and video using Stroke-based Rendering (SBR). The algorithms are grouped by elementary artistic rendering primitive and their placement (e.g. regions, strokes, stipples, tiles).

Mould and Rosin developed a standard benchmark data set consisting of 20 photos [Mou17]. A list of image characteristics was taken into account, ranging from the level of detail to contrast and visual clutter.

Kumar *et al*. provided an extensive survey on NPR techniques such as image abstraction, artistic stylization, line drawing, engraving, color enhancement, pencil drawing, dithering, stippling, halftoning, hatching, and mosaicking [Kum19]. Furthermore, they developed a respective benchmark considering technology, algorithms, parameter and design.

NPR techniques have been successfully classified and have benchmarks. We now consider geometry-based techniques and their vectorized representation in more detail. Bertin's Visual Variables (1967/83) include seven properties, which distinguish one graphic object from another [Ber67]. The variables consist of shape, hue, value, position, size, orientation, and texture. They are used in cartographic design, graphic design, and data visualization. This research forms a starting point in the study of geometry-based image stylization, whereby mainly vector-based representations of image stylization are generated. For this we want to find shapes and other properties to optimize vectorization. Selinger described a polygon tracer and how the algorithm generates Bézier curves as vector outline for fonts or logos [Sel03].

There are only a few research about more complex things like a vectorization of an NPR technique. Glöckner *et al*. introduced an optimized vectorization of GPU-based stylized images using intermediate data representations [Glö20]. The approach allows interaction manipulation of parameters and support for various stylization pipelines.

Song *et al*. [Son08] proposed arty shapes and mixed it with NPR techniques [Son08]. Song et al. generates an image segmentation to be able fitting best shapes to each segment. Afterwards, an NPR technique such as

oil or crayon painting is applied. This approach supports the creation of synthetic artworks. Overall, this research demonstrates a contribution to FSTs, which we are further expanding with our contribution.

3 CONCEPT

This section first describes the notation of GSTs (Section 3.1), describes their semiotic aspects (Section 3.2), and briefly outlines a general approach towards a framework that is used for implementation (Section 3.3).

3.1 Geometric Stylization Techniques

A Geometric Stylization Technique (GST) is a type of stylization or abstraction of digital images by means of geometric primitives and their appearance attributes (e.g., color, outline, texture). Therefore, a number of geometric primitives are generated and can be created as a vector representation. These are particularly suitable for line-art production by a pen plotter, laser cutter, programmable embroidery machines or CNC carving. For feasibility, we implement several GSTs, including Pencil Hatching [Pra01], Scribble [Lo19], Voronoi Stippling [Sec03], and Shape Packing [Col03].

3.2 Semiotic Aspects

To characterize GSTs, we examined over 30 artists that use geometry abstraction characteristics in their artwork as well as more than 15 GSTs for characteristic properties. As a result, we identified eight major categories as semiotic aspects (Figure 3), which are briefly described in the following.

Shape Types: We distinguish in three different shapes. Besides points (e.g., in stippling), the shape type can also be one of three line types: single straight lines, poly-lines, and curves. A circular shape is thus represented by a curve. In addition, three different types of polygons are also used: convex polygons, non-convex polygons, and general polygons.

Outline: A geometric primitive can have an outline that can be regular or sketchy by style and comprise stippling patterns.

Fill: The fill of a geometric primitive can be solid using a single, have a gradient, or a texture. The latter two can be parameterized with respect to orientation, scaling, and offset transformations.

Shape Size: For the shape size, we basically distinguish between uniform and non-uniform shape sizes used within a single artwork.

Shape Orientation: Similar to size, the orientation of a geometric primitive is either the same (uniform) or different (non-uniform) within a single artwork.

Shape Type Mixture: There are GSTs that either consist of collection of the same shape types (uniform) or combine different ones (non-uniform).

Shape Placement: The coverage of geometric primitives among themselves can be distinguished into overlapping or non-overlapping.

Placement Approach: The application of operations can be applied local (e.g., by segmentation) or global for the entire artwork.

3.3 Data Processing Operations

The implementation of GSTs usually comprises a number of data processing operation. For the classification of such operations, we distinguish between two basic data representations they operate on: Vector (V) and Raster (R). Based on these two representations, four transformations can be performed: Vector-to-Raster, Raster-to-Vector, Raster-to-Raster, and Vector-to-Vector (Figure 4). In the following, the four main operations are described by input, output, and examples.

Vector-2-Raster (V2R): These type of operations use geometric primitives with appearance information as input and output a number of raster data layers. These are usually implemented using rasterization techniques or diffusion curves [Orz08].

Raster-2-Vector (R2V): These kind of operations implements the inverse process to rasterization. It takes raster data layers as input and computes geometric primitives with associated appearance information. Exemplary implementations are tracing/vectorization with and without intermediate representations [Glö20].

Vector-2-Vector (V2V): These operations enable transformations in the same vector space, e.g. primitive filtering, geometry amplification, tessellation, subdivision, or geometric mapping stages. Thus, input and output consist of geometric primitives with appearance information.

Raster-2-Raster (R2R): Similar to V2V, input and output of the operations consist of the same, only this time a number of raster data layers such as color, depth, surface orientation, segments, structure, or flow. These operations are used for image segmentation, Neural Style Transfer (NST), algorithmic stylization techniques, as well as blending or compositing.

Considering suitable input and output, the four transformations can be combined to obtain optimized results for the implementation of GSTs. However, these can

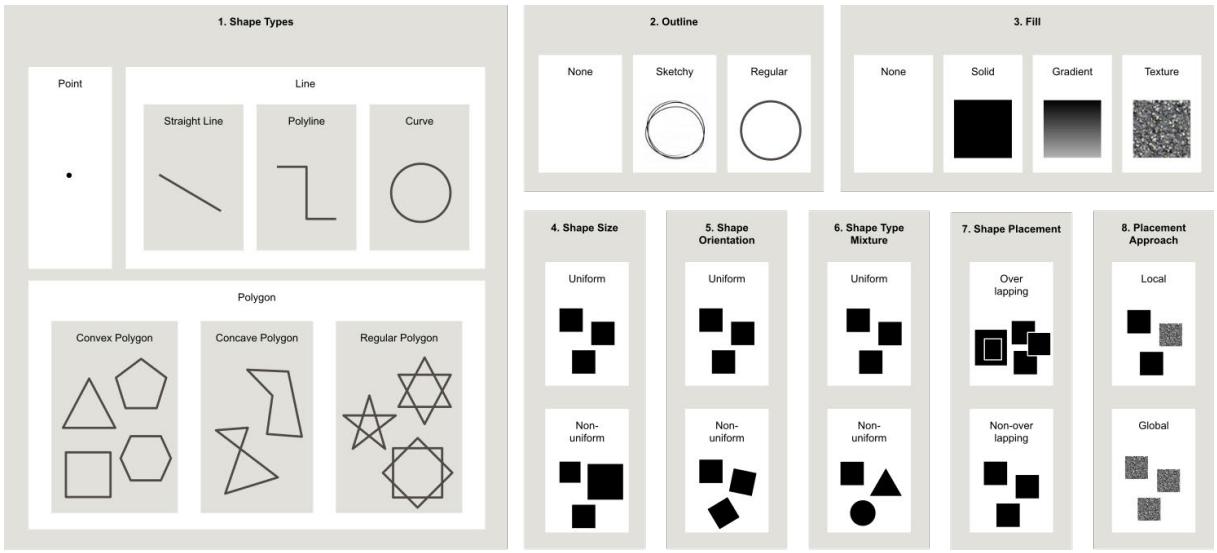


Figure 3: Summarizing presentation of the semiotic aspects of GSTs.

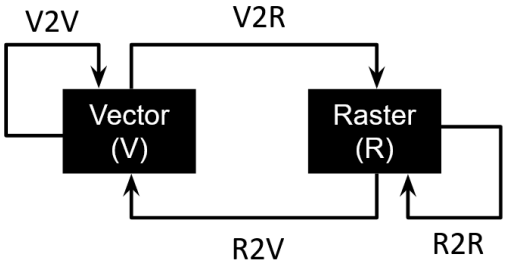


Figure 4: Classification of data processing operation between of vector and raster image representations.

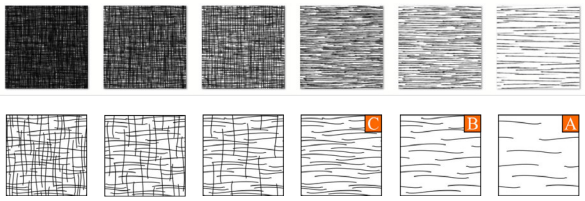


Figure 5: Top row: Conventional grid-based TAMs. Bottom row: new TAMs consisting of three line patterns (A, B, and C) only.

also form the basis for FSTs, i.e., the combination of GSTs and image-based stylization techniques [Kyp13]. This enables the implementation of a framework to imitate works by artists such as Kandinsky, Macdonald-Wright, Vasarely, Gleizes, Malevich, Feitelson, Miró, Picasso, Matisse, and Anderson.

4 APPLICATIONS

This section demonstrates the concept of GSTs by means of different application examples. For each example, we first describe the basic algorithm and following thereto the respective semiotic aspects.

4.1 Vectorized Pencil Hatching

Algorithm. The pencil-hatching approach by Webb *et al.* combines and aligns TAMs to simulate hatching strokes [Pra01]. To achieve vectorized stroke representations, we synthesized vectorized TAMs for enable clear lines in the vectorization step (Figure 5). These vectorized TAMs consist of three vector graphics, that yield another three variation by drawing lines.

For the vectorization, we generate line paths based on the edges in the image. The edges in the image were created with the help of our modified TAMs depending on the luminance in the input image. For the vector representation we support a color gradient of the lines (Figure 1). Therefore, we read luminance and color value of each node at position of input image and set the style of the path as linear gradient with luminance as opacity and individual color for each node. Listing 1 shows that stop-color specifies the color, stop-opacity the luminance, and offset the position on the path. The vectorized result consisting of lines can then also be produced with a plotter (Figure 6).

```
<svg ...>
  <defs>
    <linearGradient id="linearGradientvectorizeEdgePass">
      <stop style="stop-color:#e3da3d;stop-opacity:0.97;" offset="0"/>
      <stop style="stop-color:#d7c600;stop-opacity:0.48;" offset="1"/>
      <stop style="stop-color:#ebe022;stop-opacity:0.67;" offset="2"/>
      <stop style="stop-color:#e8dd43;stop-opacity:0.00;" offset="3"/>
      <stop style="stop-color:#e3da3d;stop-opacity:0.97;" offset="4"/>
    </linearGradient>
  </defs>
  <path
    id="vectorizeEdgePassCPU1"
    fill="none"
    stroke-width="3.0"
    stroke-linecap="round"
    style="stroke:url(#linearGradientvectorizeEdgePass)"
    d="M 5 1273.5 L 3 1254.5 L 4 1249.5 L 3 1235.5 L 1.5 1223.5 ">
  </path>
</svg>
```

Listing 1: Exemplary structure of a SVG using linear gradients to represent the results of a pencil hatching stylization.

```
for(segment in segments) do
{
 trials = TRIALS;
 shapes[] = generateShapes(numShapes, minSize, maxSize);

 for(shape in shapes) do
 {
  while(trials-- > 0) do
  {
   position = generatePosition()

   if(isInsideSegment(shape, position, segment)
      && !collisionPlacedShapes(shape, position))
   {
    placeShape(shape, position)
   }
  }
 }
}
```

Listing 2: Non-overlapping placement of shapes with varying dimensions in a limited space inside a segment.

Semiotic Aspects. Vectorized pencil hatching uses straight lines and poly-lines as shape type (Table 1). The outline comprises regular poly lines an no filled polygons. These lines are of different lengths and directions, thus the shape size and shape orientation is non-uniform. Since only lines are used, shape type mixing is characterized as uniform. The lines are often overlapping, thus the shape placement is overlapping. The algorithm is applied to the entire image, so the placement approach is global.

4.2 Segment-based Shape Packing

Algorithm. Shape packing defines the arrangement of shapes with or without overlapping each other [Col03]. For a segment-based approach, shape packing is applied locally in a confined space inside a segment (Listing 2). For the image segments, an instance segmentation was performed using the Mask R-CNN model trained on Microsoft Coco dataset (with 80 common object categories) [Lin14]. This segmentation was combined with conventional methods such as Mean Shift [Geo03], Watershed [Vin91], or DBSCAN [Est96] applied to the remaining input image to facilitate the segmentation quality. We tested the algorithm using different shape types, such as circular, rectangular, hearts, and other polygons. Per segment the appearance of the shapes can be different in size, frequency, color, and rotation (Figure 7).

Semiotic Aspects. This effect can also be classified according to the different semiotic aspects (Table 1). We have implemented and tested different shape types such as curved lines, convex polygons, as well as regular polygons. The outline appearance can be regular or none. As shape fill is usually solid or none. Settings such as gradients or textures can only hardly be reproduced by pen plotters. The shapes can be of different sizes or uniform, the same for the orientation and the mixing. The shapes should not overlap and the placement approach is local within the segments.

4.3 Scribbled Line-Art

Algorithm. The basic idea of the Scribbled Line-Art is to overlay multiple lines with on varying luminance (Listing 3). For this, random positions are calculated as the start position for the path. After that in a loop further points are calculated depending on the brightness in the image. This way a new point is created where it is the lowest luminance in the neighborhood of the pixel. This point is then added to the path with a slight rotation transformation. These points of the path result later in a line. Several of these lines stacked on top of each other then have the doodle effect. This approach produces a different result each time and is therefore part of the research field of generative art [Gal03]. Similar to Vectorized Pencil Hatching, we set per node of the path the color as it was in the input image. Thus we get a color gradient along the line (Figure 9). This vectorized result can now serve as the basis for fused stylization techniques, where we mix vector and pixel-based image stylization techniques. This is achieved by applying oil-painting, for example (Figure 9b).

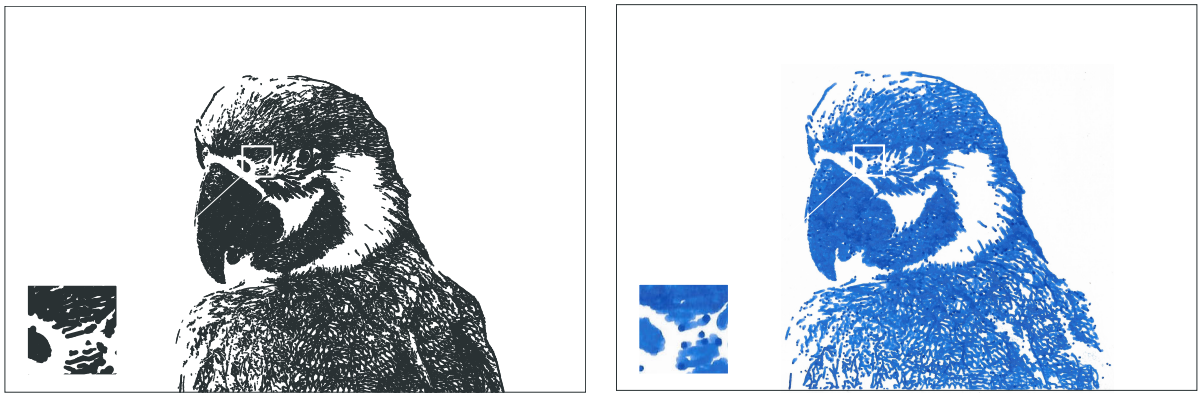
```
1  position = randomPosition();
2  addPositionToPath(position);
3  count = 0;
4
5  while(loopCount-- > 0) do
6  {
7   offset = calculateOffsetBasedOnNeighborsBrightness();
8   rotation = calculateRotationWithPerlinNoise();
9   position += offset * rotation;
10  count++;
11
12  if(count < MAX_COUNT)
13  {
14   addPositionToPath(position);
15   fadePixelsOfPath();
16  } else {
17   addPathAsLineItemAndResetCount();
18   findNextStartPosBasedOnBrightness();
19  }
20 }
```

Listing 3: Sample algorithm for scribbled line-art, which layers lines based on luminance.

Semiotic Aspects. For the semiotic aspects we use as shape type straight lines and poly-lines (Table 1). The outline is regular and the line has no filling. The lengths of the lines vary, therefore shape size is non-uniform. The lines have different rotations, so shape orientation is non-uniform. Only lines are used, so shape type mixing is uniform. The lines are superimposed, so shape placement is overlap. The placement approach is globally applied to the whole image.

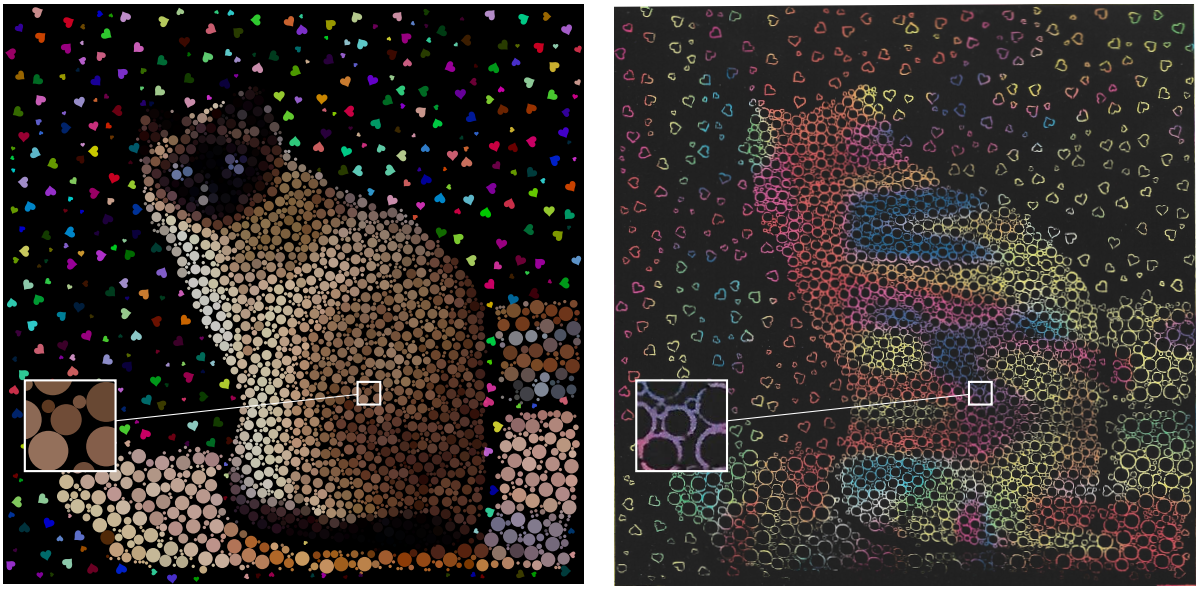
5 DISCUSSION

Evaluation. We categorized different GST according to our concept, classification and semiotic aspects. In our framework, we prototypically implemented different GST that can be combined for FST (Figure 10). In general, the classification worked well to distinguish between different GSTs. However, noticed that some cases the respective GST share the same classification



(a) Mono-colored vectorized pencil-hatching result. (b) Scan of pencil hatching using a Stabilo pen and silhouette plotter.

Figure 6: Comparison of vectorized pencil-hatching as digital and plotted result.



(a) Vectorized shape packing result with different elements. (b) Scan of scratch-art in shape packing style with silhouette plotter.

Figure 7: Comparison of vectorized shape packing as digital and plotted result.

results but the stylization computation show fundamentally different visual appearance. For example, in Table 1, there are the same selected categories for Section 4.1 and Section 4.3.

Preliminary considerations should be made for the particular GST to achieve optimum quality. Synthesis of shape types is not trivial, e.g., Section 4.1 requires new TAMs to make the line detection work. However, there are clear differences between raster-based pencil hatching and vector-based pencil hatching (Figure 8). For example, vectorized lines with gradients applied do not appear coherent and plausible compared to a TAM based on hand-drawn strokes and matched imprint of pencil on paper. Therefore, the vector file could be improved in post-processing by making it exportable with appropriate settings for different plotters. In addition, the line density due to the vectorized TAMs is not as dense as with the raster-based technique. A possible approach would be to add more layers of lines to fill the gaps.

Some materials and tools did not work well with the plotter for certain GST. For example, using scratch paper was not optimal for scribbled line-art because it overlays different lines. A scratch paper consists of only one black layer that covers a colored background. Additionally, the shapes sizes of Section 4.2 should not be set too small, to achieve a pleasing result.

Further, the plotter tools used have a fixed physical width that limits the production with the plotter. Thus, either the canvas size or the shape size need be to increased. The properties in the SVG representing the paths do not cover every desirable aspect, making the differences between raster-based and vector-based results obvious. To counterbalance this, workarounds that

	Shape Type						Outline			Fill				Shape Size		Shape Orientation		Shape Type Mixing		Shape Placement		Placement Approach	
	point	line			convex polygon	concave polygon	regular polygon	none	sketchy	regular	none	solid	gradient	texture	uniform	non-uniform	uniform	non-uniform	uniform	non-uniform	over-lapping	non-overlapping	global	local
		straight line	polyline	curve																				
4.1 Vectorized Pencil Hatching		•	•							•	•				•	•				•		•	
4.2 Segment-based Shape Packing				•	•		•	•			•	•	•	•	•	•	•	•	•	•		•		•
4.3 Scribbled Line-Art		•	•							•	•				•	•				•		•	

Table 1: Comparison of the presented GST example applications based on the presented semiotic aspects.

are specific for a GST are required. For pencil hatching example, the line segments could be grouped by pencil print, as this would be individually selectable especially for a silhouette printer.

Nevertheless, one would then have to draw each group subsequently to achieve the desired quality. However, that approach increases the reproduction time required. In general, there are various plotters on the consumer market, including home-built ones, that usually offer a smaller range of options. It would be desirable to specify different settings per GST for different plotter types.

Limitations & Future Research. As already mentioned, the plotter hardware with its tools and the material used is a major limiting factor with respect to the quality of the reprodcution. Therefore, some GSTs are not suitable for all types of production with the plotter. One solution here was to divide the file into several small parts. Further, we observed that FSTs are not suitable in all combinations.

In order to combine different GSTs and enhance the user experience, a separated stylization and layering of GSTs with parallel processing would be a possible extension. To improve or evaluate the described semiotic aspects of GSTs, further techniques could be implemented. Additionally, further FST can be implemented to imitate geometric abstraction artists. More plotters (manufacturers) should be tested to define settings in each case, so that the production can then provide the best result.

6 CONCLUSIONS

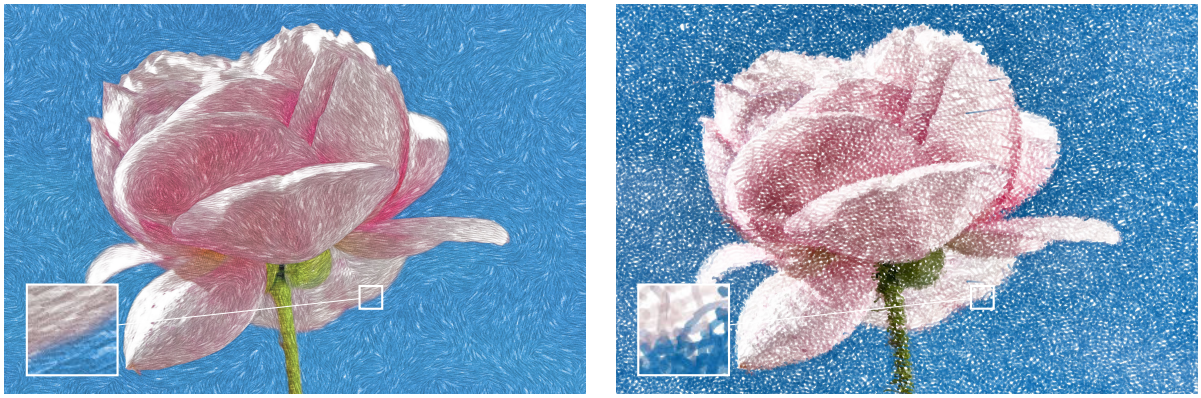
This paper presents the concept and semiotic aspects of geometry-based stylization techniques. To demonstrate its feasibility we developed a prototypical framework for implementation. We evaluate this framework by means of different application examples for geometry-based stylization techniques. By integrating image-based stylization techniques, the present approach enables the implementation of fused stylization technique, which denotes the combination of geometry-based and raster-based stylization techniques. The frame represent a basis for future research in image stylization.

ACKNOWLEDGMENTS

We thank the reviewers for their insightful comments. The project was supported by the Federal Ministry of Education and Research (BMBF), Germany (mdViPro, 01IS18092).

REFERENCES

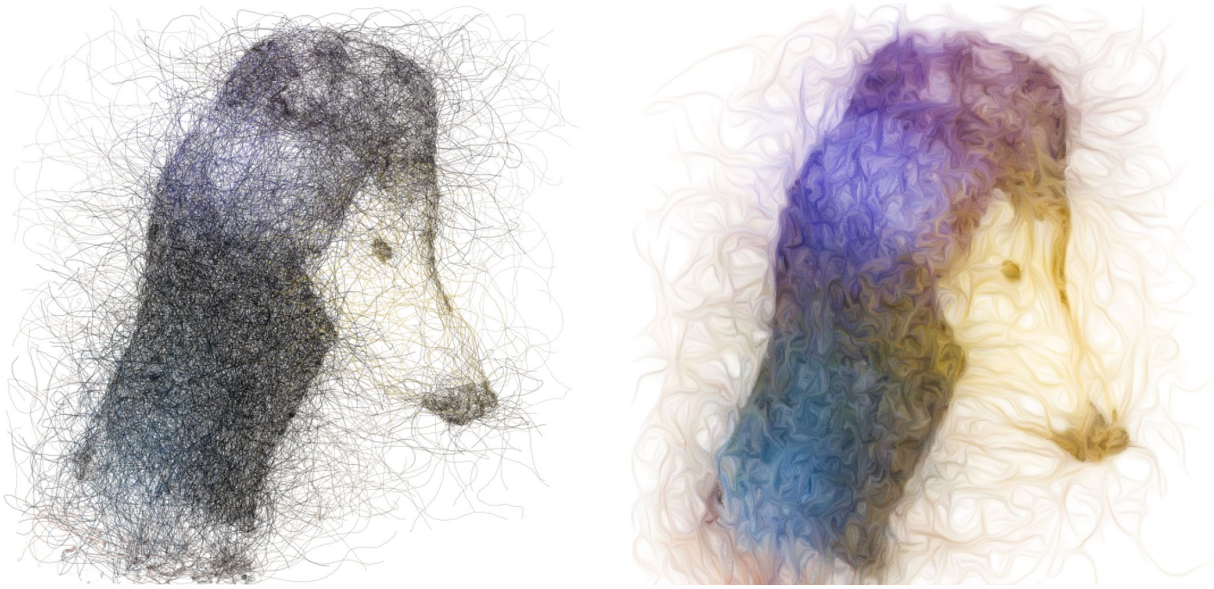
[Ber67] Jacques Bertin. “Sémiologie Graphique - Les diagrammes, les reseaux, les cartes”. In: *Geographical Journal* 135 (Jan. 1967). DOI: 10.2307/1795660.



(a) Raster-based Pencil Hatching Result.

(b) Vectorized Pencil Hatching Result.

Figure 8: Comparison of raster-based and vectorized Pencil Hatching. The vectorized lines with gradient are less realistic and the line density due to the optimized TAMs is not as dense as with the raster-based technique.



(a) Vectorized result of scribbled line-art of a duck.

(b) Applied Oil-painting on the vectorized scribble result.

Figure 9: Example of a FST that combines a scribbled line-art GST and an oil-painting image-based stylization technique.

[Bou06] Adrien Bousseau et al. “Interactive Watercolor Rendering with Temporal Coherence and Abstraction”. In: *Proceedings of the 4th International Symposium on Non-Photorealistic Animation and Rendering*. NPAR ’06. Annecy, France: Association for Computing Machinery, 2006, pp. 141–149. ISBN: 1595933573. DOI: 10.1145/1124728.1124751.

[Col03] Charles R. Collins and Kenneth Stephenson. “A circle packing algorithm”. In: *Computational Geometry* 25.3 (2003), pp. 233–256. ISSN: 0925-7721. DOI: https://doi.org/10.1016/S0925-7721(02)00099-8.

[Dev13] Kapil Dev. “Mobile Expressive Renderings: The State of the Art”. In: *IEEE Computer Graphics and Applications* 33.3 (2013), pp. 22–31. DOI: 10.1109/MCG.2013.20.

[Est96] Martin Ester et al. “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. In: vol. 96. Jan. 1996, pp. 226–231.

[Gal03] Philip Galanter. “What is generative art? Complexity theory as a context for art theory”. In: Jan. 2003.

[Geo03] Bogdan Georgescu, Ilan Shimshoni, and Peter Meer. “Mean Shift Based Clustering

in High Dimensions: A Texture Classification Example". In: vol. 1. Nov. 2003, 456–463 vol.1. ISBN: 0-7695-1950-4. DOI: 10.1109/ICCV.2003.1238382.

[Glö20] D.-Amadeus J. Glöckner et al. "Intermediate Representations for Vectorization of Stylized Images". In: *J. WSCG* 28.1-2 (2020), pp. 187–196.

[Kum19] Mp Kumar et al. "A comprehensive survey on non-photorealistic rendering and benchmark developments for image abstraction and stylization". In: *Iran Journal of Computer Science* 2 (Sept. 2019). DOI: 10.1007/s42044-019-00034-1.

[Kyp13] Jan Eric Kyprianidis et al. "State of the "Art": A Taxonomy of Artistic Stylization Techniques for Images and Video". In: *IEEE Transactions on Visualization and Computer Graphics* 19.5 (2013), pp. 866–885. DOI: 10.1109/TVCG.2012.160.

[Lin14] Tsung-Yi Lin et al. "Microsoft COCO: Common Objects in Context". In: vol. 8693. Apr. 2014. ISBN: 978-3-319-10601-4. DOI: 10.1007/978-3-319-10602-1_48.

[Lo19] Yi-Hsiang Lo, Ruen-Rone Lee, and Hung-Kuo Chu. "Generating Color Scribble Images using Multi-layered Monochromatic Strokes Dithering". In: *Computer Graphics Forum* 38 (May 2019), pp. 265–276. DOI: 10.1111/cgf.13636.

[Mee20] Jayananden Meenakchisundaram. *Review on Building A Cost Efficient Pen Plotter*. Oct. 2020.

[Mou17] David Mould and Paul Rosin. "Developing and applying a benchmark for evaluating image stylization". In: *Computers & Graphics* 67 (June 2017). DOI: 10.1016/j.cag.2017.05.025.

[Mue15] Stefanie Mueller and Patrick Baudisch. "Laser cutters". In: *interactions* 22 (Aug. 2015), pp. 72–74. DOI: 10.1145/2811292.

[Orz08] Alexandrina Orzan et al. "Diffusion Curves: A Vector Representation for Smooth-Shaded Images". In: *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2008)*. Vol. 27. 2008.

[Pra01] Emil Praun et al. "Real-Time Hatching". In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '01. New York, NY, USA: Association for Computing Machinery, 2001, p. 581. ISBN: 158113374X. DOI: 10.1145/383259.383328.

[Sec03] Adrian Secord. "Weighted Voronoi Stippling". In: *Proc. of the 2nd Int. Symp. on Non-photorealistic Animation and Rendering* (Mar. 2003). DOI: 10.1145/508530.508537.

[Sel03] Peter Selinger. "Potrace : a polygon-based tracing algorithm". In: 2003.

[Sem161] Amir Semmo et al. "Image stylization by interactive oil paint filtering". In: *Computers & Graphics* 55 (2016), pp. 157–171. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2015.12.001.

[Sem162] Amir Semmo et al. "Interactive Image Filtering with Multiple Levels-of-Control on Mobile Devices". In: *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications*. SA '16. Macau: Association for Computing Machinery, 2016. ISBN: 9781450345514. DOI: 10.1145/2999508.2999521.

[Sem20] Amir Semmo and Sebastian Pasewaldt. "Graphite: Interactive Photo-to-Drawing Stylization on Mobile Devices". In: *ACM SIGGRAPH 2020 Appy Hour*. SIGGRAPH '20. Virtual Event, USA: Association for Computing Machinery, 2020. ISBN: 9781450379656. DOI: 10.1145/3388529.3407306.

[Son08] Yi-Zhe Song et al. "Arty Shapes." In: Jan. 2008, pp. 65–72. DOI: 10.2312/COMPAESTH/COMPAESTH08/065-072.

[Vin91] Luc Vincent and Pierre Soille. "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations." In: *IEEE Trans. Pattern Anal. Mach. Intell.* 13 (Jan. 1991), pp. 583–598.

[Win06] Holger Winnemöller, Sven C. Olsen, and Bruce Gooch. "Real-time Video Abstraction". In: *ACM Trans. Graph.* 25.3 (July 2006), pp. 1221–1226. ISSN: 0730-0301. DOI: 10.1145/1141911.1142018.

Figure 10: Results produced with the proposed system for geometry stylization techniques: vectorized pencil hatching, segment-based shape packing, scribbled line-art, and combinations. The input images are obtained from the benchmark of Mould and Rosin.