ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

School of Information Sciences and Technology
Department of Informatics
Athens, Greece

Master Thesis
in
Computer Science

Privacy awareness in point clouds using AI-based object detection

Fakidis George

Supervisor: Prof. George Xylomenos
Department of Informatics
Athens University of Economics and Business

Committee: Prof. Georgios Papaioannou
Department of Informatics
Athens University of Economics and Business

Prof. Vasilios A. Siris
Department of Informatics
Athens University of Economics and Business

March 2026

Fakidis George

Privacy awareness in point clouds using AI-based object detection

March 2026

Supervisor: Prof. George Xylomenos

Athens University of Economics and Business

School of Information Sciences and Technology

Department of Informatics

Mobile Multimedia Laboratory

Athens, Greece

Abstract

Point clouds captured by depth sensors are used in both indoor and outdoor settings. This raises privacy concerns that increase the friction between researchers and people that might be identifiable in the data collected. This thesis proposes a privacy-aware pipeline that detects faces and people in point clouds in real-time and blurs them before rendering the point cloud to an output device. A software proof-of-concept was developed using Python, Open3D, OpenCV and Intel's tooling for an actual Intel depth camera. We explored the possibility of using models that operate directly on 3D point clouds instead of established 2D technologies, in order to investigate the software ecosystem's maturity. Our results show that unlike 2D object detection, the 3D object/face detection ecosystem is not ready for widespread adoption and deployment in production, without considerable programming effort.

Περίληψη

Τα νέφη σημείων που καταγράφονται από αισθητήρες βάθους χρησιμοποιούνται τόσο σε εσωτερικούς όσο και σε εξωτερικούς χώρους. Αυτό σαφώς δημιουργεί ανησυχίες για την ιδιωτικότητα που αυξάνουν την τριβή μεταξύ ερευνητών και ατόμων που ενδέχεται να είναι αναγνωρίσιμα στα δεδομένα που συλλέγονται. Αυτή η διπλωματική εργασία προτείνει έναν τρόπο επεξεργασίας λαμβάνοντας υπόψη την ιδιωτικότητα, ο οποίος ανιχνεύει πρόσωπα και ανθρώπους σε πραγματικό χρόνο σε νέφη σημείων και τα θολώνει πριν από την εμφάνιση του τρισδιάστατου νέφους σημείων σε μια συσκευή εξόδου. Αναπτύχθηκε ένα λογισμικό ως προτότυπο προκειμένου να εξερευνηθεί η ιδέα, χρησιμοποιώντας Python, Open3D, OpenCV και τα εργαλεία της Intel για την κάμερα βάθους. Επίσης, εξερευνήσαμε τη δυνατότητα χρήσης μοντέλων που λειτουργούν απευθείας σε τρισδιάστατα νέφη σημείων αντί για καθιερωμένες τεχνολογίες που λειτουργούν σε δισδιάστατες εικόνες, προκειμένου να διερευνήσουμε την ωριμότητα του οικοσυστήματος λογισμικού. Τα αποτελέσματά μας δείχνουν ότι σε αντίθεση με την ανίχνευση αντικειμένων 2D, το οικοσύστημα ανίχνευσης αντικειμένων/προσώπων 3D δεν είναι έτοιμο για ευρεία υιοθέτηση και ανάπτυξη σε περιβάλλον παραγωγής χωρίς σημαντική προγραμματιστική προσπάθεια.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor *George Xylomenos*, for their support and guidance throughout the course of this thesis. I would also like to thank the reviewers, Professor *Georgios Papaioannou* and Professor *Vasilios A. Siris* , for taking their time to grade and review this work.

This work was carried out within the Mobile Multimedia Lab (MMLab), and I am grateful to the lab and its people for providing the resources and environment that made this research work possible.

I owe a special thanks to Dr. *Yiannis Thomas*, whose academic guidance and hands-on involvement were invaluable throughout this project.

I would also like to thank Dr. *Iakovos Pittaras* for his support and advice, particularly on matters of academic writing.

Finally, I would like to thank my family and friends for their patience and encouragement during this period.

Contents

1 Introduction

1.1 Motivation

A Point Cloud (PC) is a representation of a 3D scene in the form of a set of points with positioning and texture information (such as depth and color). PCs are increasingly used in robotics, autonomous vehicles and surveillance, settings that involve people being captured digitally in some form. This is a considerable invasion of privacy, especially considering the popularity of sensors embedded in modern cars that capture their surroundings at all times while driving, in order to provide autonomy related services. Technological privacy solutions should be applied, such as blurring or pixelation, but in a smart manner, so as to not to obstruct the original goals of PC capturing devices. Unlike 2D camera images, privacy in 3D point cloud data has not yet received enough attention in the literature. Existing anonymization techniques and tools focus mostly on 2D imagery, leaving a gap for 3D sensor data. The goal of this thesis is to examine how such anonymization techniques can be applied to an actual 3D point cloud stream from a depth camera.

1.2 Contribution

In this thesis, we developed a proof-of-concept solution that utilizes a depth camera that captures point clouds from a short distance and blurs any detected faces and people to enhance privacy. We explored PC-based AI models to find faces and people in a volumetric frame. The models we experimented with are called *RandLA-Net*, *KP-FCNN* and *PointPillars*. The experiments with PC-based AI models did not bring the results originally expected, most likely due to the stark differences between our use case and the datasets and settings on which PC-based AI models are trained. As a fallback mechanism, we used existing 2D image-based technologies for face and people detection, flattening the 3D point cloud to apply them. Nevertheless, we built an open system that captures points clouds and processes them, which can be used by other researchers to experiment with newer models, as they become available. The code is available in an open repository on **Github**.

1.3 Thesis Structure

In **Chapter 2** we review the terminology and present background information around point clouds and depth cameras, face/people detection, and object detection, in order to provide a clear picture of the fundamentals required to understand all of this work. We also present various approaches to privacy preservation from the literature in the related work section. In **Chapter 3** we present and discuss the implemented system and get more in depth about its architecture and the tools used, also offering a brief demonstration of its capabilities. In **Chapter 4** we discuss the insights from this work, the limitations within the domain, and the future directions that could be further explored by other researchers interested in the topic.

2 Background and Related Work

2.1 Background

Point clouds are a set of points positioned in three-dimensional space in the form of (x, y, z) coordinates, where z is the depth within the two-dimensional image. Depending on the use case and the capture device, points can be associated with other data, such as colors and reflectance properties. PCs can be captured using specialized devices such as LiDAR (Light Detection and Ranging) cameras that project laser pulses and measure their return time in order to provide the third dimension of the points. There is also a broad category of RGB-D cameras that provide color information along with depth data, using IR sensors to estimate depth. LiDARs and RGB-D devices differ significantly in their properties and appropriate use-case scenarios. LiDARs can create high precision PCs covering a larger distance while RGB-D cameras can capture a shorter range. RGB-D devices provide color information while LiDARs can only provide reflectance values. This is the reason that in existing work LiDARs are combined with conventional cameras in order to provide colored PCs. LiDARs are expensive and mostly used for industrial applications and modern cars. In contrast, RGB-Ds are relatively cheap and more suitable for indoor applications and consumer-facing applications.

In this work we are using an Intel Realsense D435I RGB-D camera (Fig. 2.1) which is a stereo-depth camera that combines two IR lenses and an RGB lens in order to provide a colored point cloud. There are other technologies for achieving this result, but they are outside of the scope of this work. The most popular file formats for PCs are .PLY which is a polygon file format (binary and ascii text variants) and .LAS for LiDAR data. It should be noted that when using the Realsense SDK[1] for this thesis, I found a bug related to the export of textual .PLY files and fixed it; it was accepted upstream as can be seen in my Github Pull Request[2].

Bird's Eye View (BEV) is a 2D projection of a 3D point cloud onto the ground plane, effectively collapsing the vertical dimension and representing the scene as if viewed from directly above. It is commonly used in machine learning as it allows the use of standard 2D convolutional networks, and provides a natural representation for detecting objects

[1] https://github.com/realsenseai/librealsense
[2] https://github.com/realsenseai/librealsense/pull/14711

Fig. 2.1: Intel Realsense stereo-depth camera, model D435I

such as cars and pedestrians whose size remains consistent, regardless of their distance from the sensor.

Object detection is a domain of computer vision that attempts to classify objects within an image and find their bounds. In the case of 2D object detection, the goal is to find a rectangle and in PCs it is about providing a cube. *Face detection* is a specific case of object detection, where the goal is to detect faces in an image. It should not be confused with *face recognition*, which relies on comparing a face scan against the scans from a face recognition database [26]. In face recognition the focus is on extracting facial features and trying to discern different faces. In this work we only explore object and face detection and not face recognition. *Semantic segmentation* is a task similar to object detection but differs in the predictions it provides. Instead of predicting boxes that contain objects, it classifies each point in the PC as belonging to a specific class. As there is the assumption that each point belongs to a class, there is always an unlabeled class in models. An example from our own experimentation with pretrained models can be seen in Fig. 2.2.

Point clouds produced by our camera are $\sim$ 300K points per frame, a number based on the technical specification of the camera. Higher resolution devices can transmit even larger point clouds. The actual size of a point cloud in RGB-D cameras also relies on the shot, as the camera has a limited depth resolution, therefore areas further away do not produce points. In some cases point clouds are converted into a three-dimensional grid comprised of cubes that each contains a certain number of points. These cubes are also known as *voxels* in the literature and are primarily used for reducing the computational cost of operations on PCs and grouping neighboring points in 3D representations of objects.

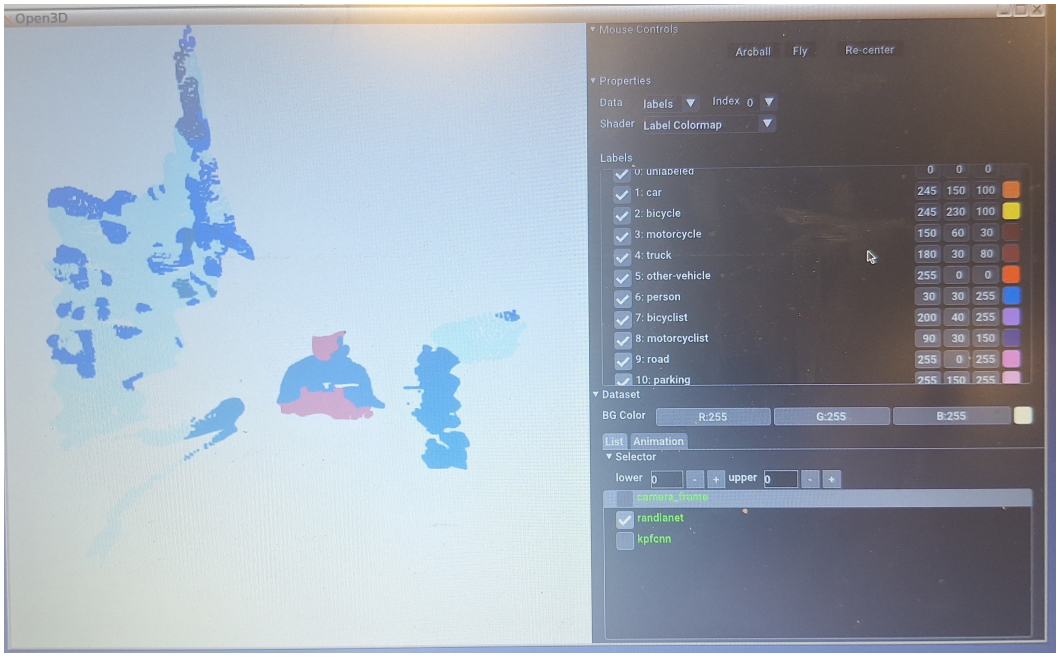

Fig. 2.2: Semantic segmentation example on a captured point cloud using pretrained models.

2.2 Related Work

Object detection in PCs imposes extra constraints on the AI architectures developed for learning-based tasks. The model result should ideally follow some prescriptive rules in order to be effective for the task, at hand as outlined in [20]. Those rules are implicitly followed in all methodologies presented in this section.

1. The ordering of the points should not influence the end result.

2. Transformations of the points, such as translation (moving the points around together) and rotation should not change the final prediction.

3. The neighborhood of a point contains significant information that should be actively utilised in the learning process.

The model should be able to handle sparse point clouds, which commonly occur when points are captured using LiDAR cameras.

Foundational work in object detection for PCs has been done in [20]. In this work the authors introduce *PointNet*, a neural network architecture that operates on PCs and not some other derivative form, as we will see in other approaches. They approximate a symmetric function by utilising a multi-layer perceptron (MLP) and a max-pooling function

and therefore achieving the first model requirement as outlined earlier. The work in this model and the refined version in the next paragraph is reused in newer architectures.

The authors of PointNet followed up on their work and presented [21] where they introduce an enhanced version of PointNet, called *PointNet++*. In this work they replaced the single max pooling operation of PointNet with a hierarchical mechanism that samples centroids from the point cloud, groups the points in regions defined by the centroids and then extracts local features from the regions. In this way, patterns are captured in multiple scales making features more detailed and robust.

In [38] the authors take a different approach and instead of raw-point clouds their model, *VoxelNet*, uses voxels as the data representation, allowing them to take advantage of more points in a point cloud efficiently. More specifically they introduce a *Voxel Feature Encoding* (VFE) layer that summarizes point-wise features in a voxel. Stacking VFE layers achieves a high-dimensional representation, which is then used by a Region Proposal Network (RPN) for object detection.

In [13] the authors propose *PointPillars*, an architecture that consumes point clouds as input and outputs 3D bounding boxes for the labels car, pedestrian and cyclist. Instead of doing full voxelization like in [38] they cut the point cloud into slices along the z dimension, called pillars, which improves performance significantly reaching 16 ms for detection, making it faster than models released afterwards but with less accuracy [39]. They use a simplified PointNet to generate features which are fed into a 2D convolutional network to produce a high-level representation of the features. Lastly they use a standard *Single Shot Detector* (SSD) to do the 3D object detection. We attempted to use this model in our experiments as it is the fastest one and we found pretrained weights for it.

In [25] the authors present *Complex-Yolo*, an architecture that uses a bird's eye view produced by a colored PC which is then fed into a simplified YOLOv2 network layer with an additional RPN layer that regresses to the orientation of the final bounding box. They achieve a considerable performance of 50 fps on an NVIDIA TitanX GPU, and for training they used the dataset from [7].

In [23] the authors propose *Point-RCNN* a two-stage object detection methodology based on R-CNNs and PointNet++ as a backbone network. The first stage generates a small number of 3D bounding boxes and separates points into foreground and background ones. The second stage then refines those proposals using the information produced in the first stage.

The work in [24] brings the *Graph Neural Network* (GNN) architecture into point clouds. It constructs a graph of the point cloud by adding edges between neighboring points in a

fixed radius. For the detection it uses a MLP for classification and a MLP that computes a bounding box for each class.

The work in [19] introduces *PointFormer*, a Transformer based backbone for learning features from point clouds. To capture local, global and the interaction between local and global information the authors create a local Transformer (LT) module, a global Transformer (GT) module and a local-global transformer (LGT). The LT module learns features in small regions, the GT module learns context-aware representations at the captured point cloud level while the LGT aggregates local features into the context used by the global transformer.

The authors at [10] present *Randla-net*, a network architecture based on random sampling and shared MLPs which make it extremely efficient in contrast to other models. We utilised *Randla-net* in the semantic segmentation scenario because the tools we used provided pretrained weights for it.

The work in [29] introduces a novel point convolution operator that places learnable weights at a set of kernel points in Euclidean space and applies them to neighboring points, allowing convolution to operate directly on raw point clouds without any intermediate representation. The paper presents two network architectures built on this operator: KP-CNN for object classification and KP-FCNN for semantic segmentation. We utilised KP-FCNN in our semantic segmentation example as there were available pretrained weights.

Face detection as a distinct task is not that popular in point cloud research. Face recognition has been covered in more past work, as point clouds offer richer information than conventional 2D images. In contrast, people detection has been a topic of interest in far more work, as it is massively more represented in datasets and very important for self-driving related tasks. In [11] the authors present a single stage 3D person detection network that uses a submanifold sparse convolution as an attempt to evaluate its effectiveness in this domain. In [2] the authors implement people detection by using a Random Forest Classifier (RFC) based on hand-crafted features such as voxel-based curvature histograms and local surface curvature variations. The point clouds are first pre-processed by removing the ground points and thus keeping points richer in information. As we will see in the datasets section, most datasets include the category *pedestrian*, which is a less invasive process compared to face detection.

Datasets containing faces are mostly oriented towards tasks such as face recognition, pose estimation and other tasks that require more tuning than conventional object detection. Face recognition datasets differ significantly from object detection datasets as they already have cropped and aligned the faces in the data without leaving the rest of the background intact. In addition, access to face recognition datasets is not publicly open, unlike object detection and semantic segmentation datasets. Most datasets for object detection and

semantic segmentation in point clouds are for scenarios involving self-driving vehicles and are captured by LiDAR devices combined with conventional cameras in order to provide a colored a full 360 degrees PC. They include pedestrians as a category but not faces. An overview of the datasets can be seen in Table 2.1.

Dataset	Task	Classes	Setting	Size
KITTI [7]	Object Detection	3	Outdoor	7,480 frames
nuScenes [4]	Object Detection	23	Outdoor	1,000 scenes × 20s
ONCE [15]	Object Detection	15	Outdoor	1M scenes
Argoverse 2 [33]	Object Detection	26	Outdoor	1,000 scenarios
JRDB [16]	Object Detection	1	Indoor/Outdoor	∼1 hour @ 30fps
SUN RGB-D [27]	Both	37	Indoor	10,335 images
SemanticKITTI [1]	Semantic Segmentation	28	Outdoor	43,000+ scans
Toronto-3D [28]	Semantic Segmentation	8	Outdoor	∼75M points
Semantic3D [8]	Semantic Segmentation	8	Outdoor	∼400M pts/scan
nuScenes-lidarseg [6]	Semantic Segmentation	32	Outdoor	40,000 scans

Tab. 2.1: Overview of datasets for 3D object detection and semantic segmentation

The work at [7] created KITTI, one of the earliest benchmarks and datasets for LiDAR-based 3D object detection, annotating classes such as car, pedestrian, and cyclist. An improvement upon KITTI, the *nuScenes* dataset from [4] is richly annotated with 23 labeled classes and 8 attributes. It differs from previous work significantly, as it uses all of the available autonomous vehicle sensors with a full 360 degree field of view. In [15] the authors present ONCE, a dataset with 1 million LiDAR scenes and annotations for 15 classes. The *Argoverse 2* dataset is introduced in [33], notable for its broad 26-category taxonomy. The *JRDB* dataset introduced in [16] contains around an hour of sensor data in an uncontrolled environment, including data from LIDAR channels and an RGB-D camera, produced from a mobile robot. The dataset was then annotated with 3D bounded boxes by human labelers for pedestrians. The *SUN RGB-D* dataset from [27] contains 10.335 RGB-D images that are annotated with 2D polygons and 3D bounding boxes for objects, taken with four different rgb-d devices. The rich information of this dataset makes it suitable for a variety of tasks in the general domain of scene understanding and more specifically for object detection and semantic segmentation which we are interested in.

Datasets for *semantic segmentation* are also important and mostly include outdoor scenes and driving footage. They are either new datasets or annotated versions of existing datasets. The *semanticKitti* dataset from [1] was created by annotating appropriately the KITTI Vision Odometry Benchmark from [7]. The *Toronto-3D* dataset from [28] was created using a Mobile Laser Scanning (MLS) system in Toronto, Canada. It uses 8 labeled object classes and consists of $\sim$ 75 million points. They also have a ranking list of models and how well they perform using this dataset at their github repository. The *Semantic3D.net* dataset from [8] provides dense point clouds acquired with static terrestrial lasers scanners. Each scan contains $\sim$ 400 million points, making it the highest resolution amongst all

datasets. The *nuScenes-lidarseg* dataset from [6] was created by annotation work for semantic segmentation on the [4] *nuScenes* dataset.

Privacy is important in many domains such as pedestrian tracking [12, 36, 17] and medicine [34]. To ensure privacy in images by hiding faces or even people, there are several methods that can be implemented. Blurring and pixelation, the simplest and most popular ones, belong to the category of image filtering techniques for privacy protection [18]. Blurring obfuscates an image by replacing pixel values with a weighted average of its neighbors, where the weights most often follow a Gaussian distribution based on the distance of the neighboring pixel from the original pixel. Pixelation first divides the image into pixel blocks. Each pixel block takes as color the average color of all the pixels in the block.

Blurring and pixelation often fail to completely remove identifiable information, as discussed in [22]. More sophisticated methods mostly rely on Generative Adversarial Networks(GANs) such as the approach taken in [3], where both biometric(e.g. faces) and non-biometric information such as clothing and hairstyle are replaced. Face swapping is also used for privacy preservation as seen in works such as [36, 14, 35]. In the source image, faces are swapped with similar faces and integrated into the image for a smooth and consistent result. To address limitations of GANs the work of [5] introduces Face transformer, a Transformer-based network architecture that achieves considerably detailed face swapping results. To achieve privacy protection, individuals can also take some preventive measures to evade existing face detection and face recognition systems, presented in [9]. However, this is outside of the scope of this work and we do not investigate it further.

There are several methodologies and systems implemented related to our work, trying to enhance privacy in real scenarios. In [30] the authors implement a real-time system for human face detection and recognition. This case involves CCTV images in contrast to our application that is based on point clouds. The face detection module is based on existing 2D based methods, more specifically a cascading classifier. In this work we also utilize a cascading classifer as an alternative mechanism for face detection for when point cloud based methods cannot be reasonably applied.

The authors in [17] use colorless point clouds instead of conventional cameras to preserve privacy while doing pedestrian trajectory tracking. Their choice of hardware and configuration is what allows them to create a private testbed.

The authors of [36] have created an end to end pipeline for protecting pedestrian privacy in 2D videos. They achieve this by swapping the detected faces with dissimilar but relevant faces from an existing face library combining multiple AI models in order to have a consistent and smooth anonymized video as a result. In this work we work with point clouds as our primary data where an analogous process is not directly applicable. A similar approach to [36] is explored in [12] where the authors go a step further when it comes to

privacy, replacing people with wireframes created using a generative method introduced in [3] that produces results such as Fig. 2.3. They also ensure that the poses of pedestrians are preserved, in order to maximize the utility of the privacy-enhanced data.

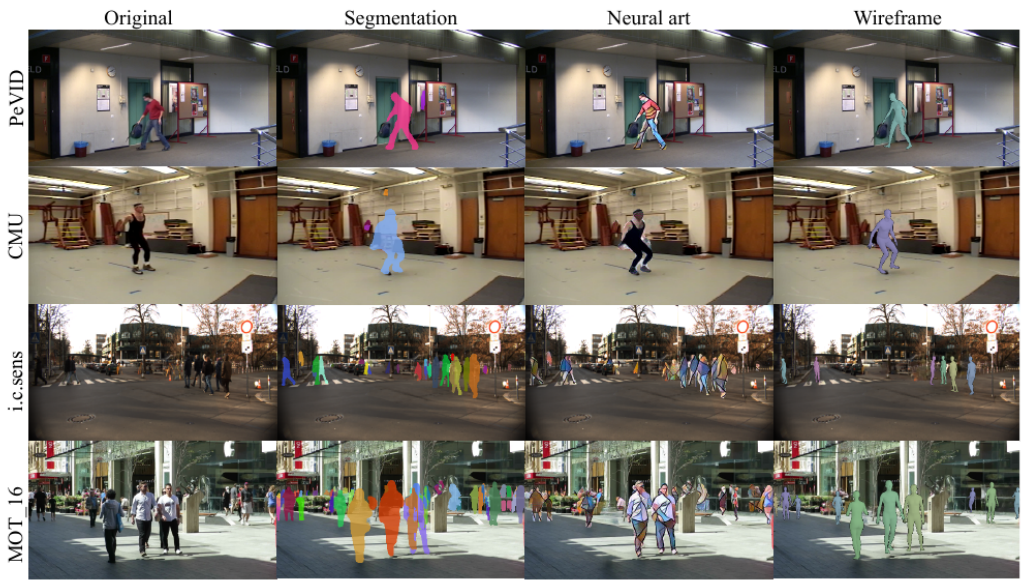

Fig. 2.3: Anonymization of pedestrian videos using generative methods.

Commercial software from RIEGL[3] includes an image anonymizer as part of their product suite. Another solution for automated blurring in laser scans comes from Celantur[4]. The Celantur software suite, as stated in their technology brief, seems to rely on 2D based CNNs, not using 3D models. This provides bits of evidence that 3D PC solutions are not yet adopted by the industry.

[3]https://www.riegl.com
[4]https://www.celantur.com/image-anonymization-terrestrial-laser-scanning/

3 System Design and Implementation

3.1 Work Setting

To obtain real-time images, we used an Intel Realsense camera model D435I as seen in Fig. 2.1. Its technical specs are the following:

- *Resolution*: up to 1280x720
- *Depth Field of View*: 87 degrees wide horizontally, 58 degrees vertically
- *FPS*: 30
- *Operating range*: Min. 0.3m, Max 3m

For complete technical specifications, one can consult *Intel's official specification.*

The setting is at an office, an indoor place, lit either by indirect natural light or normal overhead office lights. A simple rendering of a point cloud captured by the Intel camera can be seen in Fig. 3.1. What we want to achieve is to cover people and faces that appear in the point cloud frame, using a simple blurring effect. There are more advanced techniques for anonymization, as previously discussed, such as the sophisticated face swapping demonstrated in [36]. The implemented system architecture allows for their integration in an easy and straightforward manner.

Fig. 3.1: Point cloud rendered without blurring.

3.2 Demonstration

The software works by first reading the depth and color frame at each iteration of the program. Then, different face or people detection modules run using the depth and color frames as data returning a label and a box in either 2D or 3D coordinates based on the data it is operating on. 2D object detection modules can use only the color frame to run their detection. Finally the rendering engine applies all the blurring effects based on the detection results and renders the resulting PC as it can be seen in 3.2. The blurring at Fig. 3.2 is done using *OpenCV's Face Detection* module, applying a gaussian blur filter on the detection result.

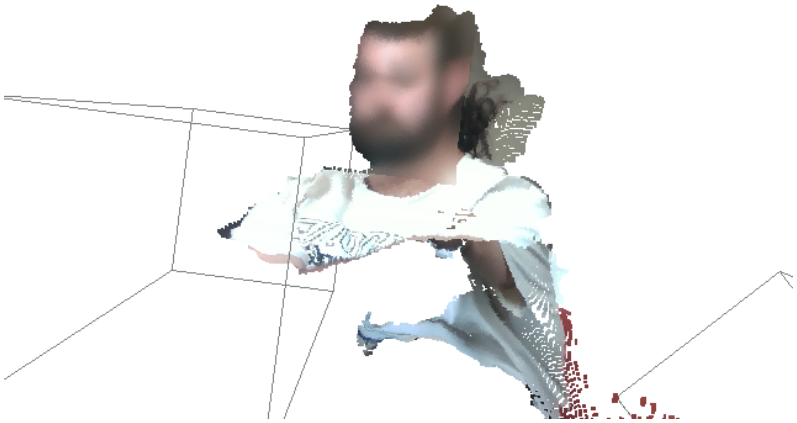

Fig. 3.2: Point cloud rendered with blurred faces. The bounding box demonstrates we can also add 3D detection.

A semantic segmentation scenario was implemented using the *RandLANet* and *KPFCNN* models provided with pretrained weights from a library (open3d-ml, discussed in Sec. 3.4). The results can be seen in Fig. 3.3. Both models did not manage to correctly identify the person in the middle of the scene.

A 3D object detection experiment was then ran using the *PointPillars* model from [13]. The model did not correctly identify the person at ∼1m away from the camera. This result could most likely be due to the fact that such models are trained on datasets from LiDAR, which are very different to our environment, both in setting and in scale. LiDAR scans often cover tens of meters ahead, while our camera can only reach 3 meters, so the scale of the faces and people is very different.

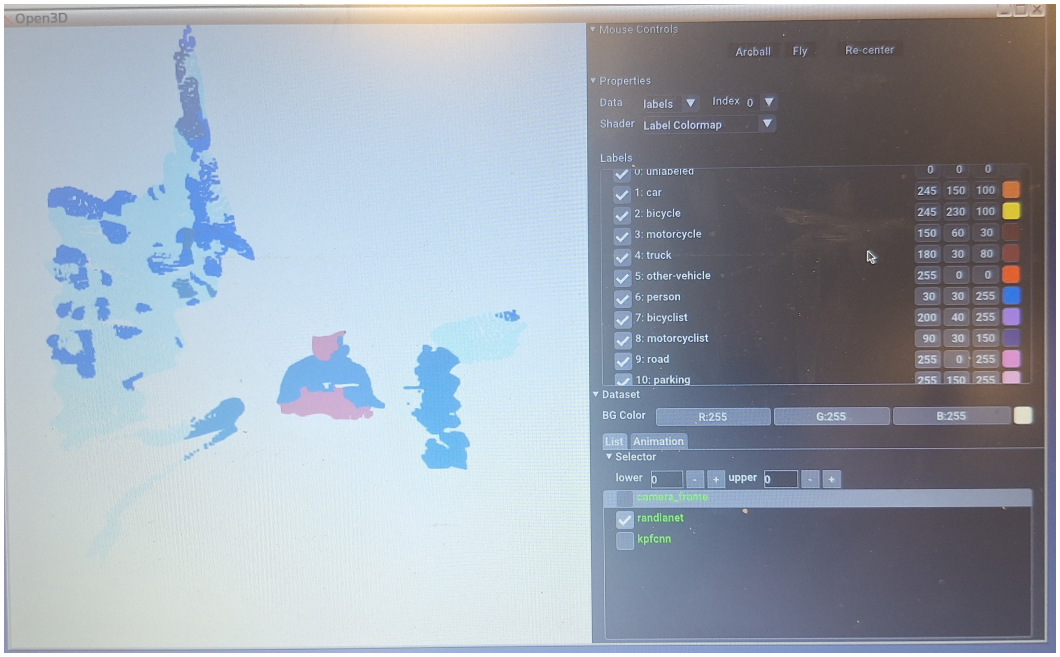

Fig. 3.3: Semantic segmentation using RandLaNet and KPFCNN from Open3D library.

3.3 Design

The design of our system ensures features can be added seamlessly, as explained in Sec. 3.4. A CameraReader interface was creaated in order to be able to support cameras from different manufacturers, by implementing the appropriate derived class, including cameras that stream point clouds over the network.

The EffectsManager class is responsible for running the tools that operate on the depth, color and point cloud data. It produces a list of effects that tell the renderer what Effects to apply on the initial PC. It ensures that all of the detection tools use as input the same data without mutations. In addition, it could allow for running the detections in parallel (using different threads to spread the load to different CPU cores) for better performance and for toggling tools on and off based on certain conditions or user input. Adding a PC-based model is as simple as adding to the EffectsManager an EffectProducer that returns a list of CubeAdd objects.

3.4 Implementation

Python was the programming language of choice in this project, as most AI related models and libraries support python. Another reason was to speed up development; otherwise it would make more sense to implement everything in C++ as AI models and related libraries are usually implemented in it. Several tools were used in order to focus more

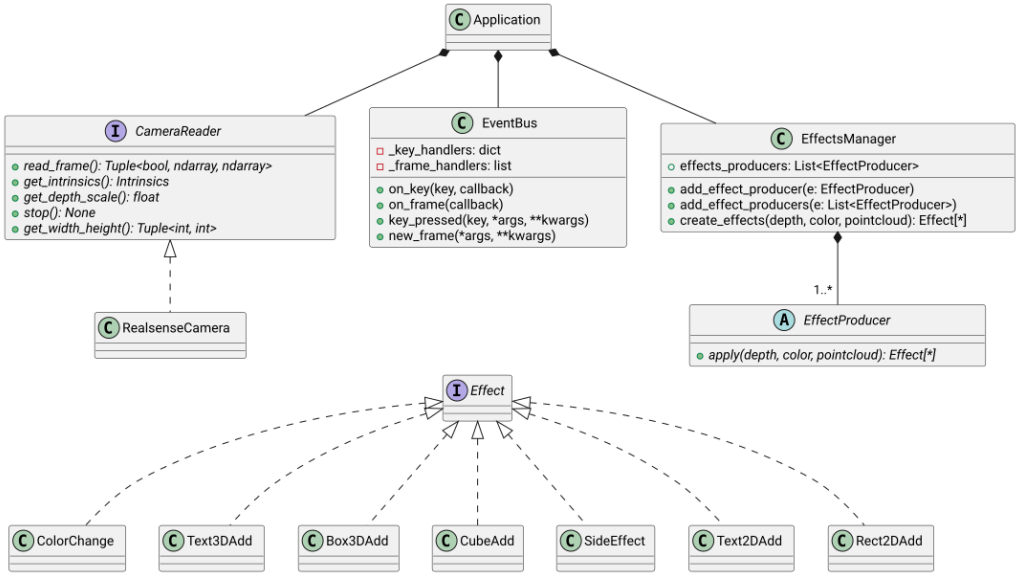

Fig. 3.4: UML Class Diagram of our system's architecture.

on the use case and less on the implementation details outlined in Fig. 3.5. *Open3D* is a C++ based library with a python wrapper for convenience made by the Intel Intelligent Systems Lab organization for processing and visualizing PCs, presented in [37]. *Open3D-ML* is an extension to the open3D library that focuses on AI-based tasks such as object detection and semantic segmentation. It supports both PyTorch and Tensorflow as the 3D machine learning base tool and supports GPU acceleration for computationally expensive 3D operations. Both libraries are properly maintained and were used in this work.

For 3D object detection the PointPillars model was used as the research indicated it is reasonably fast and within this library it is implemented and pretrained on the KITTI dataset from [7] in both PyTorch and Tensorflow implementations. For 3D semantic segmentation the *RandLANet* model from [10] and the KP-FCNN model from [29] were used, with pretrained weights on the semanticKITTI dataset from [1].

OpenCV is the industry standard when it comes to computer vision in 2D scenarios. We used this library for 2D face detection as a fallback, as the PC based models were too complicated to setup and expensive to fine-tune. More specifically we used the Haar Cascade Classifier in openCV that is based on early machine learning work at [31]. OpenCV was also used to implement blurring in an image as the 3D equivalent would be more computationally expensive. Gaussian blurring replaces each pixel's value with a weighted average of its neighboring pixels; the weights follow a Gaussian distribution, which in this case results in pixels closer to the center contributing more than the pixels further away to the final color of the blurred pixel. We also used the OpenCV DNN module to do general object detection using a MobileNet-SSD implementation with pretrained weights

and network architecture provided by a *.caffemodel* and a *.prototxt* file respectively. We used *Pyrealsense2* from Intel's official SDK for Realsense cameras,for interfacing with the camera D435I. People interested in the implementation of the system can find the code in **Github**.

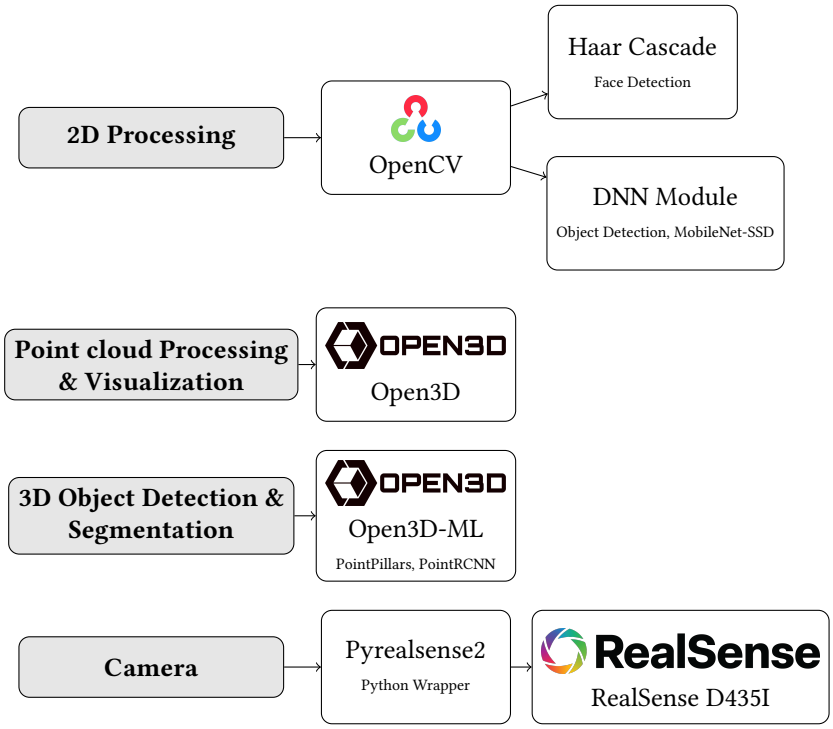

Fig. 3.5: Technologies used in the system implementation.

Tools that were not used that could otherwise be useful are the following:

- *OpenPCDet*,which heavily depended on CUDA without AMD GPU/CPU equivalents, therefore it was not suitable for this work.

- *MMDetection3D*, supports lots of models we reviewed, but we had issues when trying to integrate it with our work; a great tool nonetheless. Specifically, the problem was that it required a different version of PyTorch (1.8+), in contrast to Open3D (2.2+).

- *Learning3D*, does not support many models, we tried to use it and did not have the desired results. Also integrating with our existing code would be infeasible as it required pytorch 1.3+ and a CUDA GPU.

- *CUDA-PointPillars*, a CUDA optimized implementation of PointPillars from NVIDIA that achieves comparable detection performance but at considerable less time, 6.84ms.

4 Conclusions

4.1 Limitations

In general, the 3D AI-based Object Detection and general PC processing Python ecosystem is not that mature yet. There are great tools that are either still at an early research phase or not properly maintained. This makes them unsuitable for software in production environments. A notable exception was the *Open3D* package.

Face detection as a task is not that popular in PC related settings. The target of the latest models is usually detecting pedestrians in an outdoor setting, which makes more sense for driving scenarios. There has been more focus in face recognition tasks as point clouds can provide richer features of a face than the two-dimensional equivalent and can therefore improve the effectiveness of face recognition systems. Most of the attention is on traffic settings, other use cases remain mostly unexplored e.g. indoors 3D cameras. In [32] the authors improve upon an existing model to make it more suitable for complex traffic conditions indicating the research attention of this use case.

To achieve results analogous to 2D face detection we would need to create a face dataset and train a PC-based model. Training from scratch a PC-based model is expensive computation wise and time wise so we could only use models with pretrained weights openly available. In addition, as there are not specialized face detection datasets it would be necessary to create our own, a resource hungry process.

4.2 Discussion

Anything in 3D Point-Clouds is more complex than the 2D equivalent for many reasons. The non-grid-like structure makes simple operations more advanced to implement, the sparseness of PCs necessitates extra attention when it comes to statistical metrics used and the significant payload size drives up the computation cost considerably. Being able to efficiently and effectively provide privacy awareness in PCs could enable work on traffic settings by reducing the legal process that datasets have to go through, and enable more people-centered applications of PC work.

4.3 Future Work

Several research directions occur from the research conducted in this thesis.

Evaluation of 2D vs 3D detection models. A systematic comparison between 2D and 3D object detection models for face and people detection in both indoor and outdoor settings would provide more insight into the trade-offs between the two approaches. Such an evaluation should consider detection accuracy, robustness to lighting conditions, occlusion handling, and computational cost.

Optimization of 3D detection models for embedded systems. Investigating how to optimize 3D detection models for deployment on embedded and low-powered systems, is an important next step. Currently models achieve inference times around 30-60 FPS but with special server/desktop hardware as noted in [40].

Color-independent model training. Training detection models on colorless point clouds and evaluating whether they can match the performance of their color-dependent counterparts would be a valuable contribution. Color information in existing datasets can introduce biases tied to specific environments or demographics. A model that relies solely on geometric features would generalize better across different sensors and scenarios, and would be applicable in settings where color data is unavailable or unreliable.

End-to-end 3D video anonymization pipelines. Building on work such as [36], developing end-to-end pipelines for 3D video anonymization would enable privacy-compliant data collection at scale. Such pipelines would need to handle temporal and visual consistency across frames, real-time performance, and robustness to challenging conditions, representing a significant but impactful engineering and research challenge.

Dataset creation with indoor scenes including people. Existing datasets mostly include a pedestrian/person category in outdoor settings involving driving, which means people are mostly walking upright and there is not a variety of poses. Indoor large-scale datasets such as SUN RGB-D at [27] are more geared towards objects and rooms that do not include people. Creating a dataset specifically addressing this case could possibly enable a wider range of use cases and applications of PC-based people detection.

Investigation of hybrid approaches for people detection. Hybrid approaches, meaning combining 2D and 3D people detection cohesively, could be investigated to see its effectiveness. Intuitively, at short distance 2D methods should perform better and 3D methods should be more effective at larger distances(as most datasets used cover around 30-40 meters of distance in front of the LiDAR device).

Bibliography

[1]Jens Behley, Martin Garbade, Andres Milioto, et al. "Semantickitti: A dataset for semantic scene understanding of lidar sequences". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 9297–9307.

[2]Assia Belbachir, Antonio M. Ortiz, Atle Aalerud, and Ahmed Nabil Belbachir. "Advancing Point Cloud Perception: A Focus on People Detection". en. In: *SN Computer Science* 6.6 (July 2025), p. 698.

[3]Karla Brkic, Ivan Sikiric, Tomislav Hrkac, and Zoran Kalafatic. "I Know That Person: Generative Full Body and Face De-identification of People in Images". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. ISSN: 2160-7516. July 2017, pp. 1319–1328.

[4]Holger Caesar, Varun Bankiti, Alex H. Lang, et al. "nuscenes: A multimodal dataset for autonomous driving". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11621–11631.

[5]Kaiwen Cui, Rongliang Wu, Fangneng Zhan, and Shijian Lu. "Face transformer: Towards high fidelity and accurate face swapping". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 668–677.

[6]Whye Kit Fong, Rohit Mohan, Juana Valeria Hurtado, et al. "Panoptic Nuscenes: A Large-Scale Benchmark for LiDAR Panoptic Segmentation and Tracking". In: *IEEE Robotics and Automation Letters* 7.2 (Apr. 2022), pp. 3795–3802.

[7]Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[8]Timo Hackel, Nikolay Savinov, Lubor Ladicky, et al. "Semantic3D.net: A new Large-scale Point Cloud Classification Benchmark". In: arXiv:1704.03847 (Apr. 2017). arXiv:1704.03847 [cs].

[9]Md Rezwan Hasan, Richard Guest, and Farzin Deravi. "Presentation-level Privacy Protection Techniques for Automated Face Recognition—A Survey". en. In: *ACM Computing Surveys* 55.13s (Dec. 2023), pp. 1–27.

[10]Qingyong Hu, Bo Yang, Linhai Xie, et al. "Randla-net: Efficient semantic segmentation of large-scale point clouds". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11108–11117.

[11]Dan Jia and Bastian Leibe. *Person-MinkUNet: 3D Person Detection with LiDAR Point Cloud*. arXiv:2107.06780 [cs]. July 2021.

[12]Anil Kunchala, Mélanie Bouroche, and Bianca Schoen-Phelan. "Towards a framework for privacy-preserving pedestrian analysis". In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2023, pp. 4370–4380.

[13]Alex H. Lang, Sourabh Vora, Holger Caesar, et al. "Pointpillars: Fast encoders for object detection from point clouds". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 12697–12705.

[14]Yixuan Li, Chao Ma, Yichao Yan, Wenhan Zhu, and Xiaokang Yang. "3d-aware face swapping". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 12705–12714.

[15]Jiageng Mao, Minzhe Niu, Chenhan Jiang, et al. *One Million Scenes for Autonomous Driving: ONCE Dataset*. arXiv:2106.11037 [cs]. Oct. 2021.

[16]Roberto Martín-Martín, Mihir Patel, Hamid Rezatofighi, et al. "JRDB: A Dataset and Benchmark of Egocentric Robot Visual Perception of Humans in Built Environments". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.6 (June 2023), pp. 6748–6765.

[17]Masakazu Ohno, Riki Ukyo, Tatsuya Amano, Hamada Rizk, and Hirozumi Yamaguchi. "Privacy-preserving Pedestrian Tracking using Distributed 3D LiDARs". In: *2023 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. ISSN: 2474-249X. Mar. 2023, pp. 43–52.

[18]José Ramón Padilla-López, Alexandros Andre Chaaraoui, and Francisco Flórez-Revuelta. "Visual privacy protection methods: A survey". In: *Expert Systems with Applications* 42.9 (2015), pp. 4177–4195.

[19]Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. "3d object detection with pointformer". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 7463–7472.

[20]Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[21]Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space". In: *Advances in neural information processing systems* 30 (2017).

[22]Daniel A. Reid, Sina Samangooei, Cunjian Chen, Mark S. Nixon, and Arun Ross. "Soft biometrics for surveillance: an overview". In: *Handbook of statistics* 31 (2013), pp. 327–352.

[23]Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. "Pointrcnn: 3d object proposal generation and detection from point cloud". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 770–779.

[24]Weijing Shi and Raj Rajkumar. "Point-gnn: Graph neural network for 3d object detection in a point cloud". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 1711–1719.

[25]Martin Simony, Stefan Milzy, Karl Amendey, and Horst-Michael Gross. "Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds". In: *Proceedings of the European conference on computer vision (ECCV) workshops*. 2018.

[26]Sima Soltanpour, Boubakeur Boufama, and Q. M. Jonathan Wu. "A survey of local feature methods for 3D face recognition". In: *Pattern Recognition* 72 (Dec. 2017), pp. 391–406.

[27]Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. "Sun rgb-d: A rgb-d scene understanding benchmark suite". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 567–576.

[28]Weikai Tan, Nannan Qin, Lingfei Ma, et al. "Toronto-3D: A large-scale mobile LiDAR dataset for semantic segmentation of urban roadways". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2020, pp. 202–203.

[29]Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, et al. "Kpconv: Flexible and deformable convolution for point clouds". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6411–6420.

[30]Rehmat Ullah, Hassan Hayat, Afsah Abid Siddiqui, et al. "A Real-Time Framework for Human Face Detection and Recognition in CCTV Images". en. In: *Mathematical Problems in Engineering* 2022 (Mar. 2022). Ed. by Nouman Ali, pp. 1–12.

[31]Paul Viola and Michael Jones. "Rapid object detection using a boosted cascade of simple features". In: *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*. Vol. 1. Ieee, 2001, pp. I–I.

[32]Hai Wang, Zhiyu Chen, Yingfeng Cai, et al. "Voxel-RCNN-Complex: An Effective 3-D Point Cloud Object Detector for Complex Traffic Conditions". In: *IEEE Transactions on Instrumentation and Measurement* 71 (2022), pp. 1–12.

[33]Benjamin Wilson, William Qi, Tanmay Agarwal, et al. *Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting*. arXiv:2301.00493 [cs]. Jan. 2023.

[34]Yahan Yang, Junfeng Lyu, Ruixin Wang, et al. "A digital mask to safeguard patient privacy". In: *Nature medicine* 28.9 (2022), pp. 1883–1892.

[35]Sahng-Min Yoo, Tae-Min Choi, Jae-Woo Choi, and Jong-Hwan Kim. "FastSwap: A lightweight one-stage framework for real-time face swapping". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2023, pp. 3558–3567.

[36] Zixian Zhao, Xingchen Zhang, and Yiannis Demiris. "3PFS: Protecting Pedestrian Privacy Through Face Swapping". In: *IEEE Transactions on Intelligent Transportation Systems* 25.11 (Nov. 2024), pp. 16845–16854.

[37] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. "Open3D: A Modern Library for 3D Data Processing". In: *arXiv:1801.09847* (2018).

[38] Yin Zhou and Oncel Tuzel. "Voxelnet: End-to-end learning for point cloud based 3d object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4490–4499.

[39] Walter Zimmer, Emec Ercelik, Xingcheng Zhou, Xavier Jair Diaz Ortiz, and Alois Knoll. "A Survey of Robust 3D Object Detection Methods in Point Clouds". In: arXiv:2204.00106 (Mar. 2022). arXiv:2204.00106 [cs].

[40] Walter Zimmer, Emec Ercelik, Xingcheng Zhou, Xavier Jair Diaz Ortiz, and Alois Knoll. *A Survey of Robust 3D Object Detection Methods in Point Clouds*. arXiv:2204.00106 [cs]. Mar. 2022.

List of Acronyms

API Application Programming Interface

WWW World Wide Web

PC Point Cloud

MLP multi-layer perceptron

BEV Bird's Eye View

RPN Region Proposal Network

GAN Generative Adversarial Network

RFC Random Forest Classifier

MLS Mobile Laser Scanning

List of Figures

List of Tables