Automated Generation of Synthetic in-Car Dataset for Human Body Pose Detection

João Borges[1], Bruno Oliveira[1], Helena Torres[1], Nelson Rodrigues[1], Sandro Queirós[1,2,3], Maximilian Shiller[4], Victor Coelho[5], Johannes Pallauf[4], José Henrique Brito[6], José Mendes[1] and Jaime C. Fonseca[1]

[1]*Algoritmi Center, University of Minho, Guimarães, Portugal*

[2]*Life and Health Sciences Research Institute (ICVS), School of Medicine, University of Minho, Braga, Portugal*

[3]*ICVS/3B's – PT Government Associate Laboratory, Braga/Guimarães, Portugal*

[4]*Bosch Engineering GmbH, Abstatt, Germany*

[5]*Bosch Car Multimédia S.A., Braga, Portugal*

[6]*2Ai - Polytechnical Institute of Cávado and Ave, Barcelos, Portugal*

jpbsilva@algoritmi.uminho.pt

Keywords: Automotive Applications, Synthetic Dataset Generation, Supervised Learning, Human Pose Estimation.

Abstract: In this paper, a toolchain for the generation of realistic synthetic images for human body pose detection in an in-car environment is proposed. The toolchain creates a customized synthetic environment, comprising human models, car, and camera. Poses are automatically generated for each human, taking into account a per-joint axis Gaussian distribution, constrained by anthropometric and range of motion measurements. Scene validation is done through collision detection. Rendering is focused on vision data, supporting time-of-flight (ToF) and RGB cameras, generating synthetic images from these sensors. Ground-truth data is then generated, comprising the car occupants' body pose (2D/3D), as well as full body RGB segmentation frames with different body parts' labels. We demonstrate the feasibility of using synthetic data, combined with real data, to train distinct machine learning agorithms, demonstrating the improvement in their algorithmic accuracy for the in-car scenario.

1 INTRODUCTION

Following recent developments in automated driving (AD) cars, the future for in-car human interaction and safety will evolve to a new paradigm. Once the driving time is out of the equation, most of the time inside the car will be spent with other types of activities that should be monitored to predict car-human interactions and for safety-related procedures (*e.g.* re-engage in manual driving). With this new trend, occupant monitoring through human body pose detection gains increased importance in AD. In the last decade, multiple machine learning approaches have been proposed in the literature for human body pose detection in RGB and depth images, showing high inference accuracy with low computation cost (Shotton et al., 2013; Rodrigues et al., 2019; Torres et al., 2019). However, this type of approach requires a large and generic image database for training. Real datasets are the primary choice to be used to train every detector. Since real sensor data is used, such datasets provide inputs with characteristics similar to those seen during inference while also providing information about the sensors' inherent noise model. The recording procedure is very time consuming and requires a lot of manual interaction, hampering the task of creating a large and generic dataset. Unlike real datasets, synthetic datasets can be generated in large quantities and with higher generalization, allowing to obtain more training data in less time. However, synthetic images may suffer from the lack of realism when it comes to modeling the visual sensor noise. Acknowledging that no publicly available human body pose dataset focus on the in-car scenario, the main goal of this work was to develop an automated and user-friendly customizable toolchain to generate realistic synthetic human body pose datasets in an in-car environment. The major novelty of this work is the toolchain itself which comprises the following contributions/advantages: realistic synthetic human body pose generation; ability to customize car, image sensor, car occupants and their motion profile; and rendering realism is achieved by image processing techniques to mimic camera's output.

The rest of this paper is organized as follows. In section 2, the related work on toolchains for synthetic dataset generation and depth-based rendering is summarized. In section 3, the overall toolchain methodology is presented, also detailing some implementation considerations of the tools and sensors used. The potential interest of the proposed toolchain is studied and discussed in section 4. Finally, the main conclusions are given in section 5.

2 RELATED WORK

Generating data for machine learning methods is an important task in a wide variety of areas. Data can be both real or synthetic, with inherent advantages and disadvantages between them.

Currently, there are several real datasets publicly available that focus on human body pose. The CMU Graphics Lab Motion Capture Database (CMU, 2016) is by far one of the most extensive dataset of publicly available motion capture data. The dataset is comprised by human body poses that include markers' 3D/2D positions and human skeleton data, and RGB frames. All the information is directly related to the Vicon system (Vicon, Oxford, UK), both for the Vicon skeleton template and RGB Vicon cameras. Currently, there are 2605 trials in 6 categories and 23 subcategories, but unfortunately none of them are related to in-car scenarios. The HumanEva Dataset (Sigal et al., 2010) is another database for human motion and pose estimation. The toolchain is comprised of a method of synchronized recording of multiple RGB video sources and 3D/2D motion capture data. The similarities between CMU and HumanEva are related to the human body pose that derives from the Vicon system. This system gives an accurate body pose, but unfortunately cannot be used for in-car scenarios due to occlusions. Within the in-car scenario, Borghi *et al.* (Borghi et al., 2017) used the Pandora dataset for the POSEidon head and shoulder pose estimator. The Pandora dataset is generated in a laboratory environment with minimal occlusion, with different subjects performing similar driving behaviours while seating on a chair. Head and shoulder orientation were captured through inertial sensors.

For scenarios where it is hard to generate robust datasets, such as in-car, or even when a larger quantity of data is required, there is the alternative to generate the data synthetically. Varol *et al.* (Varol et al., 2017) created the SURREAL dataset and toolchain. The toolchain can not be seen as a fully synthetic dataset generation pipeline, because it relies on real motion capture data. Using the Blender engine (Blender Foundation, Amsterdam, Netherlands), real motion capture data from CMU and Human3.6M (Ionescu et al., 2014) datasets is fitted into a Skinned Multi-Person Linear Model (SMPL) (Loper et al., 2015), which is textured and clothed, while the scenery is comprised of a background image, light and a camera that renders depth, RGB, surface normals, optical flow (motion blur) and segmentation frames, as well as ground-truth data for body joint locations (2D/3D). Outside the in-car human body pose focus, the SYNTHIA dataset was also created. Ros *et al.* (Ros et al., 2016) proposed a virtual world to automatically generate realistic synthetic images with pixel-level annotations, something that would be extremely time consuming if it had to be done manually. The entire urban scenery is customizable through the Unity engine (Unity Technologies, San Francisco, USA), including urban object placement, textures, weather seasons, time of day and clouds with dynamic illumination engines. Two datasets are generated: 1) the SYNTHIA-Rand that consists in RGB and labeled frames from multiple cameras placed in the city and 2) SYNTHIA-Seqs that simulates multiple depth cameras on top of a moving virtual car, generating 360° LIDAR data. The toolchain does not focus in RGB sensor realism, but instead focuses in the generalization of the scenery variables, and the automated generation of sensor image and segmentation data for training and evaluation. For object detection, the VANDAL dataset was developed (Carlucci et al., 2017). A semi-automatic procedure was used to gather 3D CAD models from Web resources, allowing to generate depth images for each one. Object classes were manually queried and downloaded from 3D CAD model repositories, making a total of 319 categories with 30 objects each. Through the Blender engine and a Python script, depth data was generated for each object, while increasing the dataset size by changing object orientation with respect to the camera and its morphology. The last two toolchains (Loper et al., 2015; Carlucci et al., 2017) focus on the advantages of 3D engines, such as Unity and Blender, enabling the access to easily customizable scenes and generation of new data from it. Also outside the in-car scenario, there are more recent approaches that make use of the Pandora dataset to exploit the generation of synthetic depth images through a GAN approach (Pini et al., 2018) to tackle the head pose estimation problem. For the same estimation problem, other methods are able to train only on synthetic images (Liu et al., 2016), however the feature input is based on RGB images, making it a more rich feature frame but less robust to light conditions.

3 TOOLCHAIN OVERVIEW

In this section, the pipeline of the proposed toolchain is described. The proposed pipeline can be divided into three conceptual modules, as illustrated in Figure 1. The first module corresponds to the human model creation (section 3.1). In this module, different human models are created through the MakeHuman engine (Bastioni, 2001), with the associated skeleton and skin texture. The second module concerns the scene engine (section 3.2). Using as input the human models created in the first stage, this module is responsible for the initialization of the scene considering all objects in it (humans, car and camera), followed by the body pose generation for each human model and associated validation with respect to collisions. Finally, the third module corresponds to the rendering phase (section 3.3), where specific camera frames and human body poses are rendered taking into account the camera perspective and customization. This last module also improves the output data (ToF images and point-cloud) through image processing procedures (namely using the Neural Style Transfer [NST] technique (Gatys et al., 2017)). Both modules (sections 3.2 and 3.3) are based on Blender and Python programming.

3.1 Human Model Creation

The first step of the proposed toolchain is the creation of different human models using the MakeHuman engine. The higher the variability of the different human models, the higher the quality of the dataset, allowing to give the expected generality for the body pose detection algorithm. This step is fully automated in the toolchain, as the models are randomly created in the initialization step and customized through specific parameters accessible to the user. Each created human model is comprised of a full body skin mesh, a full body skin texture, and a body skeleton with 6 degrees-of-freedom (DoF) per joint. Each human skin texture is fully segmented according to the different body parts. This texture is transversal to all body models, being deformed to fit the different body shapes. The segmented skin labels have specific RGB color codes, as illustrated in Figure 2, and are directly related with human body joints and body segments that represent joint connectivity.

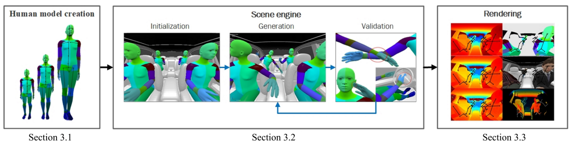

Figure 1: Overview of the toolchain pipeline.

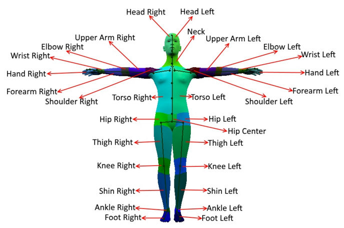

Figure 2: RGB body part segmentation of the human model. A total of 30 body part labels are used with specific RGB color codes, and are directly related with human body joints and body segments that represent joint connectivity.

3.2 Scene Engine

The scene engine module is responsible for the creation of the scene to be rendered. This block is subdivided into three different parts, namely initialization, generation, and validation. This engine uses user input to create the scenes. First, a camera model is chosen, with associated user-defined camera parameters, namely camera resolution, field of view, position, orientation and axial noise model. A car model is also given as input to this block, where a realistic 3D car model with 6DoF is used. Finally, a human motion profile is defined within the toolchain, where each human model has a Gaussian motion profile associated to each of its joints.

Initialization: The first step of the initialization stage is to position the humans, the camera, and the car model in the synthetic scenario. Every human model comes with an unrealistic motion profile, in which each joint is comprised of 3 axis with $\pm$ 180° of range of motion (RoM). Considering the real anthropometric constraints of the human body (National Aeronautics and Space Administration, 2000), the motion profile is modified.

Generation: After the initialization stage, a new scene is generated, where the human models perform random movements, each human has a user-customized motion profile that follows a Gaussian distribution for each joint axis (*e.g.* μ and σ). For every new frame generated by the toolchain, a new pose is sampled from these distributions.

Validation: After the scene has been generated, it must be validated. The validation process consists in searching for scene collisions. There are three identified collision types that need to be considered to generate realistic datasets. These collision types are illustrated in Figure 3. The toolchain automatically checks for all collisions after generating all poses, in order to guarantee the realism of the scene and dataset. If

there are no collisions for all humans, then the dataset is rendered, otherwise a new scene is generated.
Body to body collisions refer to inner body intersections (Figure 3a), inspecting the body 3D skeleton against the human model skin mesh. For this detection, the toolchain automatically creates a 3D skeleton for each human model, using the human model interior dimensions (joints' position and segments' length). Human to human collisions refer to human intersections (Figure 3b), inspecting the human models' skin mesh against each other. Human to car collisions refer to human to car intersections (Figure 3c), inspecting the human models' skin mesh against the car model mesh.

3.3 Rendering

After scene validation, the toolchain renders the dataset information (Figure 4), including: (1) depth frames (clean, noise and NST); (2) a 3D point-cloud; (3) an RGB frame, which adds extra post-processing effects to the scene; and (4) a body parts' segmentation frame. Moreover, the 2D and 3D body pose of each human model is generated and exported to JSON format.

Clean Depth Frame: The depth frame (Figure 4a) $\alpha_{x,y}$, $x = 1,\ldots,X$, $y = 1,\ldots,Y$, where X is the camera's horizontal resolution and Y its vertical resolution, is a matrix with the same size as the RGB sensor, where each pixel information is the distance from the object on the pixel in the camera XY plane.

Noise Depth Frame: The noise frame (Figure 4b) is generated using a 2^{nd} degree equation model, where the standard deviation of the error $\sigma_{x,y}$ is related to the distance of each pixel to the object. This model is used to generate a Gaussian value for each pixel of the frame $\gamma_{x,y}$. The noise model only relates the axial noise with the depth component, and it does not consider the angle between the surface normal and the camera axis (Iversen and Kraft, 2017). To use a realistic noise model, we performed a noise regression to a specific ToF camera (Pico Monstar 105). The method consisted in placing the camera in front of a white wall, at D different distances, and recording F frames for each distance (5 and 100 respectively in our experiments). The standard deviation of the error is then calculated for each distance σ_d, $d = 1,\ldots,D$, finally estimating a regression for the model *noise* using equation 1. For the specific case of the ToF camera used, we found $a = 1$, $b = 5$ and $c = 1$ as illustrated in Figure 5.

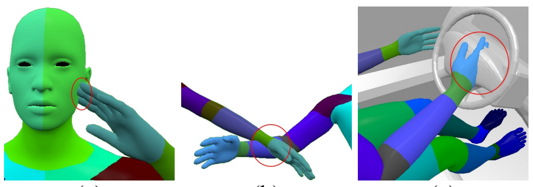

Figure 3: Collision detection types: (a) body to body, (b) human to human and (c) human to car.

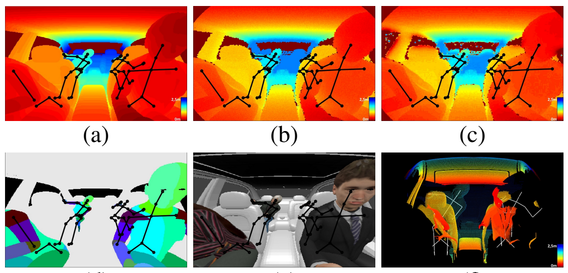

Figure 4: Rendered frames: (a) depth, (b) noise, (c) NST, (d) labels, (e) RGB and (f) point-cloud. Depth images (a, b, c) are represented in color for better visualization.

$$\begin{aligned}\sigma_{x,y} &= a + b \times \alpha_{x,y} + c \times (\alpha_{x,y})^2,\\ \gamma_{x,y} &= Gaussian(\sigma = \sigma_{x,y}, \mu = \alpha_{x,y}).\end{aligned} \tag{1}$$

Additionally, a specific Gaussian noise is added to abrupt depth edges (identified using an empirical threshold applied to an edge image computed by central finite differences), while a circular crop is applied to simulate the real ToF images field-of-view (FoV).

NST Depth Frame: The NST frame (Figure 6c) is generated with a NST method (Gatys et al., 2017). The toolchain generates a new NST frame $\beta_{x,y}$ for each synthetic noise frame $\gamma_{x,y}$, by feeding the network with a real ToF image serving as style x_S (Figure 6b), and the synthetic noise frame serving as content x_C (Figure 6a). The strategy aims to better infer the noise style and add it into the generated synthetic noise frames (Figure 6c). This frame can be seen as the most realistic synthetic depth frame from the toolchain.

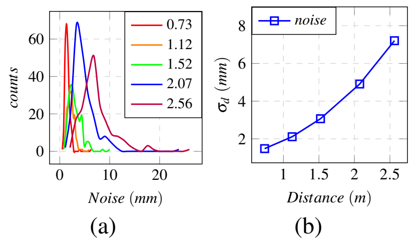

Figure 5: Pico Monstar 105 gaussian noise: (a) gaussian noise $\sigma_{d,x,y}$ for each distance value in plot (b); (b) gaussian noise regression, $noise = a - bz + cz^2$.

Labels Frame: The labels frame (Figure 4d) $labels_{x,y}$, is a matrix with the same size of the RGB sensor, where each pixel information is the RGB color code from the object projected on the pixel. Note that no post-processing effects (such as ray-tracing, ambient occlusion, ambient light, shadowing) are used, giving a raw RGB code of the projected texture and preserving the human models' segmented skin (Figure 2).

RGB Frame: The RGB frame (Figure 4e) $RGB_{x,y}$, is a matrix similar to the labels frame but with the added post-processing effects, thus improving scene realism. Also, the toolchain automatically switches the human models skin texture with a realistic one before rendering. This skin is preselected by the user for each human model, in the GUI.

Point-cloud: The 3D point-cloud (Figure 4f) has the Cartesian coordinates xyz of the voxel that was projected in each pixel, pcx_i, pcy_i, pcz_i, $i = 1, \ldots, X \times Y$. As mentioned before, Blender does not give this information in a simple and straightforward way. The toolchain relies on the information of the NST depth frame $\beta_{x,y}$, the camera's resolution (X and Y), and its horizontal field-of-view, H_{FoV}, to calculate each voxel position with respect to its pixel projection (Equation 2).

$$\begin{aligned} pcx_i &= \beta_{x,y} \times \left(x - {}^{X}\!/_{2}\right) \times \frac{\tan\left({}^{H_{FoV}}\!/_{2}\right)}{\left({}^{X}\!/_{2}\right)}, \\ pcy_i &= \beta_{x,y} \times \left(y - {}^{Y}\!/_{2}\right) \times \frac{\tan\left({}^{H_{FoV}}\!/_{2}\right)}{\left({}^{X}\!/_{2}\right)}, \\ pcz_i &= \beta_{x,y}, \end{aligned} \tag{2}$$

Ground-truth: The ground-truth information (black skeleton lines in Figure 4) consists in exporting the pose information for each human model, with respect to the camera. With that in mind, we have 2 types of ground-truth: 2D pose for depth, labels and RGB frames; and 3D pose for the point-cloud. Both types of ground-truth consist in the same pose information for each human model, that is a structure comprised of all joints' pixel (2D) or voxel (3D) positions.

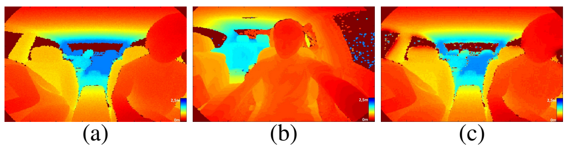
(a) (b) (c)

Figure 6: NST frames: (a) content, (b) style, and (c) resulting NST-based synthetic image.

4 EXPERIMENTAL EVALUATION

To understand the validity of the data being generated with our toolchain, as well as its ability to increase ML algorithmic accuracy, we defined four distinct experimental scenarios: 2D pose estimation from depth images; 2D pose estimation from point-cloud; 3D pose estimation from 2D pose; human body parts segmentation from depth images.

To provide the experimental scenarios with valid datasets that allow us to evaluate the advantage of combining synthetic and real data, we require both real and synthetic samples. In this sense, we used a publicly available dataset MoLa R10k InCar Dataset (Borges et al., 2019), plus the synthetic dataset generated by the proposed toolchain. The real dataset consists in three recorded subjects (S), two redundant actions (A) each, totaling 10482 samples. In its turn, the synthetic dataset comprises data generated using seven car models (N_{cm}) and eighteen subjects (Z) with associated Gaussian poses (N_{hgp}), totalling 25200 samples. Both datasets are identical in terms of sample data types for depth frame, point-cloud, 2D and 3D body pose, giving us the opportunity to evaluate the three first experimental scenarios in a quantitative way. Lack of real in-car body parts' segmentation frames in publicly available datasets led us to define a qualitative evaluation for the fourth experiment. The available samples were divided into 3 groups: (1) a training set, with all synthetic samples plus 6946 real samples (corresponding to subjects $S1$ and $S2$); (2) a validation set with 900 real samples; and (3) a test set with 2636 samples. Groups (2) and (3) use samples from subject $S3$ performing distinct actions. To assess the influence of the ratio between real and synthetic images and the influence of the number of real images available, we trained each network with different amounts of samples, establishing ten sub-evaluations ($R1$ to $R10$) for each of the first three experiments (Table 1). To account for the stochastic nature of the training, each sub-evaluation was repeated 3 times, using a different set of samples, with the metrics being averaged over the 3 trained models. For $R8$ to $R10$, due to lack of new samples for the different folds, the training was repeated thrice upon shuffling the samples.

2D Pose Estimation from Depth Images (EV1): To evaluate the synthetically generated depth frames and corresponding 2D ground-truth, the Part Affinity Fields (Cao et al., 2017) method was used. From it, a custom CNN was implemented consisting only on the first stage of the original PAF CNN. In each sub-evaluation, $R\#$, the method used the depth frame as

Table 1: Evaluations related with real and synthetic data quantities/ratios.

Evaluation	Real	Synthetic	Total	Ratio
R1	900	0	900	1:0
R2	900	2700	3600	1:3
R3	900	4500	5400	1:5
R4	900	9000	9900	1:10
R5	1800	0	1800	2:0
R6	1800	5400	7200	2:6
R7	1800	9000	10800	2:10
R8	1800	18000	19800	2:20
R9	6946	0	6946	7.7:0
R10	6946	25200	25200	7.7:28

input features (raw depth for real dataset, NST depth for synthetic dataset) and the 2D body pose as output labels (Figure 7). In this experiment, the PCKh measure (in pixels, using a matching threshold given by 50% of the head segment length) and the Area Under Curve (AUC) were used as metrics (Andriluka et al., 2014). Table 2 summarizes the average results for the full body. Figure 8a presents the PCKh@0.5 values for the full body for each sub-evaluation. We demonstrate the interest in including synthetic data in ML training, showing PCKh improvements in almost all sub-evaluations when adding synthetic data. However, some sub-evaluations showed that higher ratios would induce overfitting to synthetic data (*e.g.* $R2 > R3 > R4$, $R7 > R8$).

2D Pose Estimation from Point-cloud (EV2): To evaluate the synthetically generated point-cloud and corresponding 2D ground-truth, the Part Affinity Fields (Cao et al., 2017) method was used. In this experiment, the point-cloud was used as input features (raw point-cloud for real samples and NST-based point-cloud for synthetic samples). To this end, each point-cloud was normalized and converted into a 3-channel matrix. The same metrics from EV1 were employed. Results are shown in Table 2, and Figure 8b. In this experiment we experience the same observations as in $EV1$, except overfitting, *e.g.* addition of synthetic samples seem to always improve accuracy. Interestingly, $EV2$ achieved higher AUC values then $EV1$, meaning more detections for a lower normalized distance. These observations may be related with the link between increased complexity of the input and the need for larger number of training samples to induce overfitting to synthetic data. In the end, for the in-car 2D body pose estimation problem, we achieved higher PCKh@0.5 total score in $R10$ (*i.e.* full dataset training) of 91.97% and corresponding AUC of 63.61%.

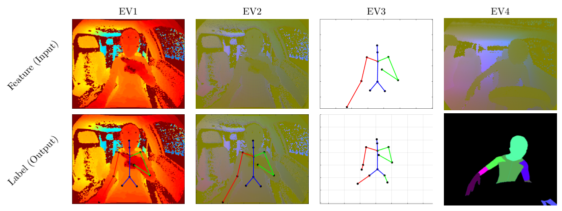

Figure 7: Visual representation of input features and output used for each experimental scenario $EV\#$: (EV1) 2D pose estimation from depth images using normalized depth frame as input and 2D body pose as output; (EV2) 2D pose estimation from point-cloud using normalized point-cloud as input and 2D body pose as output; (EV3) 3D pose estimation from 2D pose using 2D body pose as input and 3D body pose as output; and (EV4) human body parts segmentation from point-cloud images using normalized point-cloud as input and segmentation frame as output.

Table 2: PCKh measure and AUC values averaged over all 14 joints, for the 3 experimental scenarios and all 10 sub-evaluations.

	EV1		EV2		EV3	
	PCKh	AUC	PCKh	AUC	PCKh	AUC
R1	13.47	4.42	39.57	19.62	79.93	25.58
R2	86.6	41.46	89.58	62.36	90.89	35.37
R3	74.04	35.15	87.99	66.55	91.76	33.06
R4	64.79	33.57	90.67	69.05	92.95	37.53
R5	74.77	36.80	89.97	54.98	79.09	22.34
R6	83.82	38.45	91.00	67.86	91.74	31.49
R7	85.17	42.37	90.89	64.71	92.72	32.26
R8	60.18	29.05	89.79	63.57	93.64	38.64
R9	88.62	50.25	90.32	54.52	92.74	29.78
R10	87.66	41.30	91.97	63.61	95.55	39.72

3D Pose Estimation from 2D Pose (EV3): To evaluate the synthetically generated 3D ground-truth, a 3D pose estimation method (Martinez et al., 2017) was used. The method uses a 2D body pose as input features (provided as joint pixel coordinates) and the 3D body pose as output (Figure 7). Once again, similar metrics were employed, but in this case PCKh matching threshold was normalized to a fixed head size of 200 mm. Results are shown in Table 2, and Figure 8c. Similar improvements with added synthetic samples were observed as in $EV2$. We achieved higher PCKh@0.5 total score in $R10$ with 95.55% and AUC of 39.72%.

Human Body Segmentation from Depth Images (EV4): To evaluate the synthetically generated point-cloud and corresponding segmentation frames, the U-net (Ronneberger et al., 2015) method was used (Figure 9). The method was implemented in two stages, where the first stage (St_1) inferes the human silhuete from the background to then mask the input image features that are then used on the second stage

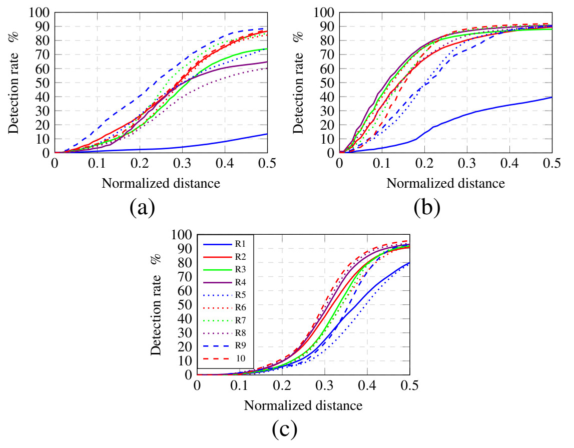

Figure 8: PCKh total for all sub-evaluations, *R#*, and the three first experimental scenarios, *EV#*: (a) 2D pose estimation from depth images (EV1); (b) 2D pose estimation from point-cloud (EV2); and (c) 3D pose estimation from 2D pose (EV3). Color gradient represents synthetic data increase with constant real data. Continuous, dashed and dotted lines represent increasing amounts of real samples (for the same real-synthetic ratio).

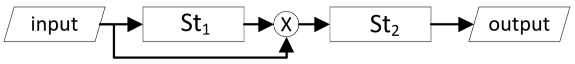

Figure 9: Inference pipeline for the experimental scenario EV4. Normalized point-cloud is used as input for the first U-Net stage (St_1) and then masked with its output. The second U-Net stage (St_2) uses the masked input and inferes the body parts' labels.

(St_2) to infere the human body parts. The method uses the NST point-cloud as input features and the labels frame as output labels (Figure 7). For all samples, the point-cloud was normalized and converted into a 3-channel matrix, while each labels frame was converted into a grayscale frame with 8 body parts. Evaluation is done qualitatively (in both unseen synthetic samples and real images) and results are shown in Figure 10. We showed that pure synthetic training is still capable of achieving good results in real data, however requiring increased method complexity to cope with lack of noise realism in synthetic data. Notwithstanding, from the results obtained for $EV1$ to $EV3$ (mix of real with synthetic samples), it is expectable a noticeable improvement if a few real samples are included during training (which was not available in the present experiments due to lack of ground-truth labels).

5 CONCLUSIONS

In this work, a novel toolchain for the generation of realistic synthetic images for human body pose detection in an in-car environment is presented. The toolchain demonstrated its potential for increased algorithmic accuracy during body pose estimation in an in-car scenario.

In terms of scene realism several considerations can be made for improvements in future work, as discriminative algorithms seem to improve their accuracy proportionally to the training data realism. In this regard, ToF noise characterization or the used NST methods can be improved, as well as the RGB image rendering. The ability to synthetically recreate human behaviour would be another important feature, enlarging the applicability of our dataset towards other monitoring tasks (like action recognition). Hereto, fusion of real human motion capture data with synthetic scenarios could be employed. However, issues such as collision detection between animated models and synthetic car models would have to be handled. Besides the currently supported pose and segmentation maps, another relevant output to be added would be the gaze for each human model.

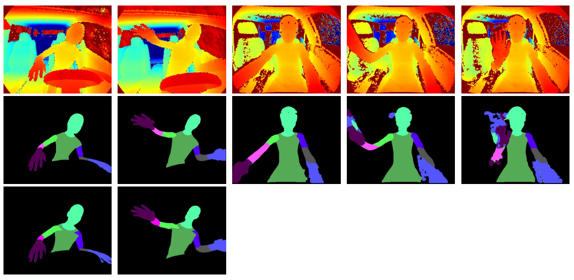

Figure 10: EV4 visual representation: first row represents the point-cloud input features (represented in depth frame to improve understanding); middle row represents infered body parts' segmentation; and bottom row represents the label frame. The first two columns represent synthetic samples from the MoLa S25k InCar Dataset (not used in training), while the last three columns represent real samples from the MoLa R10k InCar Dataset (no label frames available).

ACKNOWLEDGEMENTS

This work is supported by: European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project n^{o} 039334; Funding Reference: POCI-01-0247-FEDER-039334].

REFERENCES

Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2D human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Bastioni, M. (2001). Makehuman, Open Source tool for making 3D characters.

Borges, J., Queirós, S., Oliveira, B., Torres, H., Rodrigues, N., Coelho, V., Pallauf, J., Henrique Brito, J., Mendes, J., and C. Fonseca, J. (2019). MoLa R10k InCar Dataset.

Borghi, G., Venturelli, M., Vezzani, R., and Cucchiara, R. (2017). POSEidon: Face-from-Depth for driver pose estimation. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:5494–5503.

Cao, Z., Simon, T., Wei, S. E., and Sheikh, Y. (2017). Realtime multi-person 2D pose estimation using part affinity fields. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:1302–1310.

Carlucci, F. M., Russo, P., and Caputo, B. (2017). A deep representation for depth images from synthetic data. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1362–1369.

CMU (2016). CMU Dataset @ mocap.cs.cmu.edu.

Gatys, L. A., Ecker, A. S., Bethge, M., Hertzmann, A., and Shechtman, E. (2017). Controlling perceptual factors in neural style transfer. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pages 3730–3738.

Ionescu, C., Papava, D., Olaru, V., and Sminchisescu, C. (2014). Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339.

Iversen, T. and Kraft, D. (2017). Generation of synthetic Kinect depth images based on empirical noise model. *Electronics Letters*, 53(13).

Liu, X., Liang, W., Wang, Y., Li, S., and Pei, M. (2016). 3D head pose estimation with convolutional neural network trained on synthetic images. *Proceedings - International Conference on Image Processing, ICIP*, 2016-Augus:1289–1293.

Loper, M., Mahmood, N., Romero, J., Pons-moll, G., and Black, M. J. (2015). SMPL : A Skinned Multi-Person Linear Model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1—-248:16.

Martinez, J., Hossain, R., Romero, J., and Little, J. J. (2017). A Simple Yet Effective Baseline for 3d Human Pose Estimation. In *Proceedings of the IEEE International Conference on Computer Vision*.

National Aeronautics and Space Administration (2000). Anthropometry and Biomechanics. *Man-Systems Integration Standards NASA-STD-3000*, 1(1):16–45.

Pini, S., Grazioli, F., Borghi, G., Vezzani, R., and Cucchiara, R. (2018). Learning to generate facial depth maps. *Proceedings - 2018 International Conference on 3D Vision, 3DV 2018*, pages 634–642.

Rodrigues, N., Torres, H., Oliveira, B., Borges, J., Queirós, S., Mendes, J., Fonseca, J., Coelho, V., and Brito, J. H. (2019). Top-down human pose estimation with depth images and domain adaptation. In *VISIGRAPP 2019 - Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, volume 5, pages 281–288.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9351:234–241.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., and Lopez, A. M. (2016). The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (600388):3234–3243.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. (2013). Real-time human pose recognition in parts from single depth images. *Studies in Computational Intelligence*, 411:119–135.

Sigal, L., Balan, A. O., and Black, M. J. (2010). HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 87(1-2):4–27.

Torres, H. R., Oliveira, B., Fonseca, J., Queir&#x00F3;s, S., Borges, J., Rodrigues, N., Coelho, V., Pallauf, J., Brito, J., and Mendes, J. (2019). Real-Time Human Body Pose Estimation for In-Car Depth Images. In *IFIP Advances in Information and Communication Technology*, volume 553, pages 169–182. Springer New York LLC.

Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M. J., Laptev, I., and Schmid, C. (2017). Learning from synthetic humans. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:4627–4635.