GFPNet: A Deep Network for Learning Shape Completion in Generic Fitted Primitives

Tiberiu Cocias, Alexandru Razvant and Sorin Grigorescu

Abstract—In this paper, we propose an object reconstruction apparatus that uses the so-called Generic Primitives (GP) to complete shapes. A GP is a 3D point cloud depicting a generalized shape of a class of objects. To reconstruct the objects in a scene we first fit a GP onto each occluded object to obtain an initial raw structure. Secondly, we use a model-based deformation technique to fold the surface of the GP over the occluded object. The deformation model is encoded within the layers of a Deep Neural Network (DNN), coined GFPNet. The objective of the network is to transfer the particularities of the object from the scene to the raw volume represented by the GP. We show that GFPNet competes with state of the art shape completion methods by providing performance results on the ModelNet and KITTI benchmarking datasets.

Index Terms—Shape completion, surface modeling, deep learning in robotics and automation, GFPNet

I. INTRODUCTION

Shape completion of partial represented objects remains one of the fundamental problems in 3D perception and a key requirement for robots which interact with the physical world (e.g. mobile manipulation of objects). The main limitation is the sensor that cannot perceive the full 3D shape of the object. Therefore, many researchers focus on 3D reconstruction approaches that use one or multiple views of the object of interest to fill out the occluded information.

The robotics and computer vision communities are tackling the problem mainly by using constraints and prior knowledge of object shapes. These solutions achieve full 3D representations by registering an a-priori volume from a database onto the perceived object [1] [2] [3] [4]. The goal is to obtain a reconstructed volume that resembles the object's true form as close as possible.

Deep learning paved the way to new solutions for 3D shape completion. The objective is to learn shape models by training Deep Neural Networks (DNNs) on large collections of 3D shapes. Furthermore, at runtime, the shape is used either to retrieve or to reconstruct the object of interest [5], [6], [7], [8], [9].

In this paper, we focus on the specific problem of registering and modeling 3D shapes based on sparse and occluded 3D point clouds of objects, as illustrated in Fig.1. To cope with this

Manuscript received: January, 27, 2020; Revised May, 01, 2020; Accepted May, 30, 2020.

This paper was recommended for publication by Editor Cesar Cadena Lerma upon evaluation of the Associate Editor and Reviewers' comments. (Corresponding author: Tiberiu Cocias.)

The authors are with Elektrobit Automotive and the Robotics, Vision and Control Lab (ROVISLab), Transilvania University of Brasov, Romania (e-mail: tiberiu.cocias@elektrobit.com, alexandru.razvant@elektrobit.com, sorin.grigorescu@elektrobit.com).

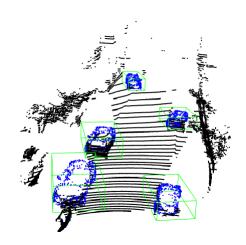


Fig. 1. Output of the GFPNet 3D shape completion apparatus. Black points describe the input scene perceived by a LIDAR sensor, while blue points define completed 3D shapes of the objects of interest (in this particular example, traffic participants imaged in a LIDAR point cloud).

problem, we propose a two stage shape completion framework that *i*) takes as input a point cloud, registering first a *Generic Primitive* (GP) onto it, and *ii*) we use GFPNet to deform (model) the appearance of the fitted GP.

To avoid learning a specific deformation model for each type of object class, we propose to decompose the GP shape into sub-regions and generically learn a deformation model from them. The neural network behaves like a solver that has to optimize following objectives:

- re-position the GP points as close as possible to the incomplete representation of the object;
- ensure that the GP points remain compact and describe a smooth surface.

The proposed GFPNet is an improvement of our previous work [10], where the GP shape was modeled using a first and second order differential equations solver used to compute internal and external shape contour energies. We present here an improvement of our 3D modeling method by replacing the solver with a DNN able to outperform [10], as well as other similar algorithms, especially on surfaces with strong deformations. We demonstrate the performance of GFPNet on the 3D shapes available in the ModelNet [11] and KITTI [12] datasets.

The main contributions of the paper are:

- introduction of GFPNet as a framework for 3D shape completion from sparse and incomplete observations;
- a deep neural network architecture for modeling 3D surfaces in point clouds.

The rest of the paper is organized as follows. We discuss the related work in Section I-A. In Section II, we present the components of the proposed shape completion framework and give a detailed description on the DNN architecture. Subsequently, in Section III, we present the test setup used for benchmarking, along with the obtained quantitative results. Finally, conclusions are stated in Section IV.

A. Related Work

The problem of 3D shape completion is usually approached in robotics and computer vision from two perspectives: analytical methods [4], [10] and deep learning algorithms for shape modeling [6], [9]. Both aim at obtaining 3D shapes from incomplete and sparse data. The analytical approaches are focused either on objects' symmetries [13], shape retrieval [14], [2], [15] or surface modeling [16], [17]. The main drawback of the analytical techniques is that they require a large amount of the shape's structure to be visible in the input point cloud.

Given the availability of increased computation backed by the evolution of Graphical Processing Units (GPUs), in the past years numerous deep neural network architectures were proposed for solving the 3D shape completion problem [9], [8], [18], [19], [7], [5], [20], [21]. In the following, we will address the most relevant approaches.

In [9], the authors propose the Point Completion Network (PCN), which is a novel learning-based approach for shape completion that operates on the shape's raw point cloud without any structural assumption (e.g. symmetry) or annotation (e.g. semantic class). The method is effective across multiple object categories and works with inputs from different sensors. Unfortunately, the network fails to work on objects with disconnected parts as well as objects containing thin structures.

Acknowledging that the shape completion problem can be tackled using shape priors, in [19] the authors proposed a weakly-supervised learning-based approach which requires no direct supervision. The network learns a shape prior on synthetic data, as well as a maximum likelihood fitting measure based on a DNN. The proposed network has difficulties completing the structure of thin objects as well as identifying the correct object category of the prior shape. This is one of the main reason why shape prior solutions are less generic.

The shape completion process in [8] uses a data-driven approach to complete partial 3D shapes through a combination of volumetric DNNs and 3D shape synthesis. The solution first infers a low-resolution, but complete output, while a 3D-Encoder-Predictor Network is used to predict and fill in the missing data. In a final pass, the authors use a patch-based 3D shape synthesis method to impose the 3D geometry from the retrieved shapes. Similar to the previous described approaches, this solution also fails to infer small and thin objects.

An important difference between GFPNet and the related work is that our network is applied on local regions instead of the entire shape at once. Given the focus of local surface modeling, the method in [16] uses 3D active contours to inflate/deflate raw 3D volumes depicting general sphere- or cube-like shapes. The approach has low complexity, with no prior model required to drive the deformation of the surface. The drawbacks of the concept are the input raw shapes, which

have to be as similar as possible to the real shapes, as well as the convergence method, which is not guaranteed due to the global minimization of the functional energy, that may get stuck in a local minimum. One other drawback is the computational time of the algorithm, making it unsuitable for real time applications.

Approaching the modeling problem from a learning perspective, the method in [22] proposes to represent the surface of a 3D object in terms of a hyper-plane. A Support Vector Machine (SVM) is used to learn a model that deforms the surface. Although this approach implicitly solves most of the 3D active contour problems, it still has difficulties in coping with surface-holes and outliers.

Using DNNs, the authors in [23] introduce a volumetric Convolution Neural Network (CNN) to learn deformation flows directly in the 3D Cartesian space. The network's architecture takes as input the voxelized representation of the shape, as well as a semantic deformation intention, generating a deformation flow as output. The authors claim comparable results with state of the art methods when applied to CAD models, whereas an $\approx 60\%$ error rate is obtained when the algorithm is applied to single frame depth scans.

In the light of the current approaches and their limitations, we propose GFPNet for enabling shape completion of thin and complex structures from spare data. The data is represented by single point cloud observations.

II. METHOD OVERVIEW

Let O be a set of 3D points lying on the observed surfaces of an object that is perceived from a single perspective. Let GP be a dense set of 3D points that describe the generic shape of the observed object, while MGP is a clone of the GP's point cloud whose 3D points have been re-positioned. We define the shape completion problem as predicting the MGP given the GP as an initial shape and O as the desired appearance (objective).

The block diagram of the proposed shape completion framework is illustrated in Fig. 2. The proposed framework has three stages. In the first stage, we apply the PointRCNN 3D object detector [24] for extracting the objects directly from the raw point cloud depicting the scene. We have chosen this particular detector based on our extensive experiments on the KITTI dataset, PointRCNN showing the best performance when compared to other state-of-the-art methods.

In the second stage, we register a GP onto the observation points in O. For this stage we use PCRNet's neural network [25], with the objective of fining the transformation which best aligns two point clouds. As with PointRCNN, we have chosen PCRNet based on its alignment accuracy and computation time, when compared with other techniques, such as *Iterative Closest Point* (ICP) [26]. For computational efficiency, we limit the number of iterations to five, as the network converges rapidly. During testing, we have determined that the average run-time for registering a GP is around three milliseconds for our proposed pipeline.

Finally, in the last stage, we apply GFPNet with the purpose of modeling the surface of the GP such that it captures the

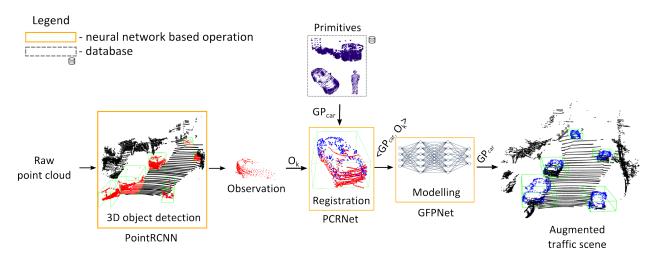


Fig. 2. Block diagram of the proposed GFPNet 3D shape completion framework. The orange colored blocks depict DNN-based operations. We register and model a GP onto each detected object O, thus capturing the particular surface characteristics of O.

particular geometries of O. The output is an MGP that best represents the real shape that generated O.

A. Generic Primitives (GP)

A Generic Primitive (GP) is a point cloud depicting a 3D volume resembling many similar objects of the same class. For example, the GP of class *car* is used to represent different types of car shapes and brands (e.g. sedan, minivan, cabrio, etc.). The GP shape is obtained using the *Generalized Procrustes Analysis* [27], by averaging the shapes of multiple objects from the same class. To ensure a complete and smooth surface of the GP we use a Moving Least Squares (MLS) surface reconstruction method to smooth and resample noisy data [28]. We are thus ensuring that all the small errors are corrected and the so-called *double walls* artefacts resulted from registering multiple shapes together are smoothed. The result is a fine shape represented by the lowest possible number of 3D points.

In total, we have defined 30 GPs depicting common household items (cups, bottles, plates, etc.), as well as dynamic traffic objects (cars, pedestrians, etc.). To compute these GPs, we have used the 3D shapes from the ModelNet [11] and ShapeNet [5] datasets. Examples of car, person and truck GPs are illustrated in Fig. 3.

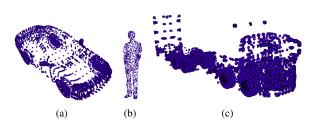


Fig. 3. **3D Point clouds of various generic primitives.** (a) Car. (b) Person. (c) Truck.

B. GFPNet

Our proposed DNN architecture for modelling 3D surfaces is presented in Fig. 4. Given the GP registered onto O, let p_i be the i-th point of the GP and S a set of 3D points depicting the neighboring points of the GP that lays inside a sphere centered on p_i (see Fig. 5). Let T be a second set of 3D points depicting the neighboring points from O which lay in the same spherical area. We define MS as a set of 3D points depicting a modeled version of S. The aim of the neural network is to calculate MS by predicting the 3D point positions of S given template T. GFPNet thus acts as a bi-objective optimizer governed by a shape representation encoded within the layers of the DNN.

The modeling of the entire GP is achieved by applying the GFPNet modeling approach on each GP point. A relevant situation is when point p_i does not have any neighboring points belonging to O. In this case, since there is no available template T, the modeled surface MS will be the same as S.

To achieve the modeling task, the GFPNet architecture uses an encoder-decoder schema, composed of sequences of convolutional (CNN) network layers. The first half of the network behaves as a feature extractor encoding the geometrical particularities of the two inputs (S and T), while the second half behaves as a decoder which regresses towards a modeled version of the source cloud MS.

The encoding of the features is performed separately for the source and template inputs, as shown in Fig. 4, such that the GFPNet will be able to learn a deformation model only for the source points. Each branch extracts features using CNN layers of sizes 64, 128, 512 and 1024, respectively. On each branch, we apply a symmetric max-pooling function to extract a strong feature vector. This step will ensure invariance to input permutations. Finally, the two feature vectors are glued together using a concatenation operation.

The decoder consist of a mirrored CNN, having layers of size 1024, 512, 256 and N, where the last dimension N is the number of 3D points in the input source cloud. The sizes of the CNNs were chosen based on performance tests.

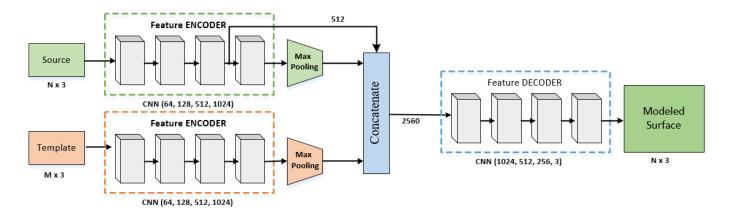


Fig. 4. **GFPNet architecture**. The architecture is primarily composed of convolutional layers. M and N are the number of points in the template and source clouds, respectively.

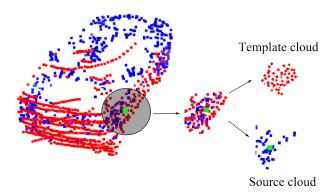


Fig. 5. Example of an input sample. Left most is a registered GP of a vehicle (blue) aligned with the respective observation O (red). The green point from the center of the gray hashed region represents point p_i , around which the GP surface is modeled. The right most regions represent the source S (blue) and template T (red) clouds.

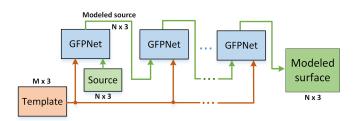


Fig. 6. Iterative GFPNet architecture used for calculating refined modeled surfaces. The template cloud remains the same, while the modeled source cloud is iteratively looped back into the network as a source cloud.

C. Iterative GFP

Depending on the complexity of the shape that must be completed, it may be the case that a single modeling iteration will not be sufficient to converge towards the optimal shape. For this reason, we introduce the iterative scheme from Fig. 6, where we iteratively apply the modelling process onto the local region around each GP point p_i .

For the iterative implementation, the CNNs use only three hidden layers of size 64, 128, 1024. In order to avoid over-fitting, we have added an additional dropout layer before the output. The reason for introducing these iterations is that it allows us to use a lower number of hidden layers, thus

increasing the processing time.

In the first iteration, the template and original source clouds are fed into the GFPNet, providing a modeled source cloud as output. In the next iterations, the modeled source cloud and the template cloud are fed iteratively to the GFPNet. The process is repeated on the same point of the GP for m iterations. A final primitive shape that best reproduces the observed object is obtained after modeling all GP points.

D. Loss Function

Throughout the deformation process GFPNet optimizes two loss functions:

- loss₁: minimize the distance between two point cloud densities;
- 2) $loss_2$: ensure a smooth modeled surface.

The two objectives are combined in the following weighted loss function:

$$loss = \alpha \cdot loss_1 + (1 - \alpha) \cdot loss_2, \tag{1}$$

where, α is the weight factor. $loss_1$ is based on the Chamfer Distance CD, used to calculate the average closest point distance between the modeled GP and the observation object O. The loss function $loss_1$ is used to quantify how similar two surfaces are. We have chosen this particular distance metric for three reasons: i) it solves the optimal bipartite matching problem, ii) it is permutation invariant, while iii) two compared point clouds do not need to be of the same size. Within GFPNet, CD is defined as:

$$CD(S,T) = \frac{1}{|S|} \sum_{x \in S} \min_{y \in T} \|x - y\|_2 + \frac{1}{|T|} \sum_{x \in T} \min_{y \in S} \|y - x\|_2,$$
(2)

where, S and T are the source and template point clouds, along with their respective 3D points defined by x and y.

The smoothness of a modeled GP surface is quantified in $loss_2$ using the Laplacian operator applied to S. The indicator $LP(S_i)$ gives us a measure of the surface's smoothness around point i in the source cloud S:

$$LP(S_i) = \frac{1}{N} \sum_{j=1}^{N} S_j,$$
 (3)

where, N is the number of neighboring points around i and S_j is the position of the j-th neighbor.

We use a value of 0.7 for α in Eq. 1, since we are more interested in calculating shapes similar to the template.

E. Dataset preparation

To train and test our model, we use synthetic CAD shapes from the ModelNet database [11] and real 2.5D shapes from the KITTI database [12]. In the following, we detail the shape classes, statistics and the splitting of the data into training and testing sets.

ModelNet: we consider 939 shapes of 4 object categories: car (297), pedestrian (108), cup (99) and bottle (435). From this pool of shapes, we use 699 (75%) for training and 240 (25%) for testing. In order to avoid overfitting, all point clouds are augmented with an additive Gaussian noise having variance 0.1m and 0.01m for large (cars, pedestrians) and small objects (cups, bottles), respectively.

We consider the 240 testing CAD shapes as ground truth for evaluating the performance of the shape completion apparatus. To produce incomplete shapes needed for the testing procedure we generate partial point clouds from the ground truth shapes. For each shape, we chose 4 randomly distributed view points for generating a depth image. We then back-project these images in 3D to obtain 4 incomplete 3D representation of the initial shape [29]. Due to the fact that this strategy produces point clouds closer to real-world sensor data, it is by far more efficient than using subsets of points from the complete shapes.

KITTI: we have extracted shapes from KITTI's Velodyne point clouds using the provided ground truth 3D bounding boxes. We thus avoid taking into consideration points from nearby objects, such as the street, walls, or vegetation. In total, 500 incomplete shapes depicting 2 categories were collected (386 for cars and 114 for pedestrians). We then split the shapes into training and testing sets using the same schema as for the ModelNet database.

The shapes extracted from the KITTI dataset have missing regions caused by the fact that the laser sensor is not able to image occluded surfaces. This limitation makes incomplete shapes unsuitable for usage as full ground truth data in testing. In [19], the authors propose a technique to generate partial ground truth shapes that can be used for evaluation purposes. Based on [19], we accumulate 3D point clouds of 10 future and 10 past frames around each object in order to reduce occlusion.

F. Training

The objective of the training process is to learn a deformation model by correlating the source with the modeled cloud, where the model is encoded within the layers of the DNN from Fig. 4. The labels in the training set represent modeled versions of the source point cloud.

Considering the GFPNet architecture, the input data is represented by a tuple of source and template point clouds. To

produce such data, we take the training shapes presented in Section II-E and register a GP over them using PCRNet [25]. Further, for each point i in the GP, we define the source point cloud as a spherical region around i containing neighboring points.

From the 699 and 375 ModelNet and KITTI training shapes, we have extracted approx. 750k and 350k training samples, respectively. To produce labels for these samples, we use the 3D active contour modeling technique from our previous work [10], enhanced by applying smoothing and re-sampling via MLS on the GP points. For GP regions where the modeling process was erroneous, we manually redistribute each point using a 3D tool.

GFPNet has been trained using the Adam optimizer [30], while the weights have been initialized according to the scheme in [31]. The Adam optimizer was chosen due to its adaptive learning rates. The training is based on a batch size of 256 for 1000 epochs, using a learning rate of 10^{-4} and a weight decay value of 0.92. We have performed all training and testing operations on computing unit equipped with a single NVIDIA GeForce GTX 1080Ti GPU and an Intel Core i7 CPU, running at 4.2 GHz.

III. EXPERIMENTAL RESULTS

A. Evaluation Metrics

We evaluate the GFPNet's performance on the ModelNet test set using the Chamfer Distance (CD). This distance provides a quantitative measure of similarity between the modeled GP and the ground truth shape defined in Section II-E. The similarity is determined as the average closest point distance between the modeled GP and the ground truth cloud. The CD formula is the same as the one used in the loss function 2.

Due to the fact that the CD metric can only be used to compare full shapes, as in the case of comparing modelled GFPNet shapes with the CAD models in ModelNet, we have evaluated GFPNet's performance on the KITTI test set using the approach from [9], where the authors proposed to use the following three metrics:

- **Fidelity** (F): the average distance from each point of the modeled GP to its nearest neighbor in the ground truth;
- Minimal Matching Distance (MMD): the CD between the modeled GP and the ModelNet object point cloud closest to the GP's points in terms of CD;
- Consistency (C): the average CD between the modeled GPs of the same instance in consecutive frames.

B. GFPNet vs GFS

In this section, we describe the evaluation of GFPNet against our previous implementation [10], referred here as Generic Fitted Shapes (GFS). To understand better the relevance of the modeling step, we also provide as baseline the performance results obtained solely by registering a GP on each O using the PCRNet registration method [25]. The GFPNet results summarized in Table I have been obtained using our iterative approach with 5 iterations. An illustration

	Car				Pedestrian				Cup	Bottle
	ModelNet	KITTI			ModelNet	KITTI			ModelNet	ModelNet
Method	CD	Fidelity	MMD	Consistency	CD	Fidelity	MMD	Consistency	CD	CD
	[m]	[m]	[m]	[m]	[m]	[m]	[m]	[m]	[m]	[m]
PCRNet baseline [25]	0.252	0.311	0.275	0.0	0.387	0.411	0.305	0.0	0.109	0.144
GFS [10]	0.040	0.089	0.12	0.043	0.057	0.069	0.049	0.048	0.027	0.029
GFPNet	0.011	0.027	0.035	0.018	0.017	0.032	0.02	0.027	0.011	0.009

TABLE I

PERFORMANCE OF GFPNET AGAINST THE BASELINE PCRNET [25] AND OUR PREVIOUS WORK ON GFS [10]

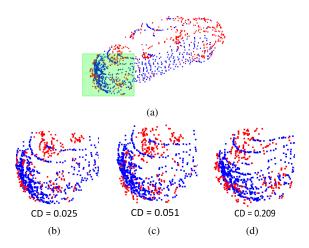


Fig. 7. Shape completion results over a front body part of a car object. The computed CD for each completion approach is given bellow each picture. (a) GP registration (red) over the observation cloud Ok (blue). (b) Iterative GFPNet. (c) Generic Fitted Shapes (GFS). (d) Baseline registration using PCRNet [25]

of shape completion results using the three considered approaches is shown in Fig. 7 for the case of a car's frontal part. The least accurate results were obtained using the baseline registration method, which is not sufficient when the requirement is to produce a highly detailed reconstructed shape.

As shown in Table I, GFPNet outperforms the baseline registration algorithm and GFS in all object categories. It is important to note that the results were highly accurate for the cases of cups and bottles. This is because the shapes of these objects depict regular surfaces. For these surface types, the GFS solution to deform along the normal direction is sufficient to capture the particularities of observation O. However, this is not the case for the car and pedestrian object classes, since they contain highly deformed and irregular surfaces. The GFS method failed here because it had only one degree of freedom to deform the GP points along the surface normal direction. GFPNet succeeded in these cases because the DNN was trained with labels that were refined, smoothed, resampled and in some cases manually adjusted to capture as accurate as possible the template surface.

On the KITTI test dataset we have obtained a slightly higher value for the CD metric, compared to the ModelNet test dataset. This is because of the scattered distribution of the LiDAR points in KITTI, compared to the synthetic CAD models from ModelNet.

C. Comparison to other methods

In the following, we briefly describe the shape completion methods used as competing algorithms in our evaluation. Here, we compare our model against shape completion methods that work on objects from multiple classes with different levels of occlusions.

- PCN: Point Completion Network [9], which is a network that uses a raw object point cloud as input and reconstructs its shape via an encoder-decoder DNN architecture. One advantage over GFPNet is that it does not require any structural or semantic a-priori knowledge about the shape of the object.
- **3D-SC**: 3D Shape Completion under weak supervision [19], where a DNN learns a shape prior on synthetic data together with a maximum likelihood fitting objective. Similar to GFPNet, it uses prior knowledge about the semantics of the shape.
- Folding [18], which is a network that also uses a similar encoder as GFPNet, but the decoder is purely foldingbased, deforming a 128 × 128 2D grid into a 3D point cloud.
- **3D-EPN**: 3D Encoder-Predictor Network [8], representing a data-driven approach to complete partial 3D shapes through a combination of volumetric DNNs and 3D shape synthesis. For comparing the outputs, we have converted the output of 3D-EPN into point clouds by extracting the isosurface around a small area and then uniformly resampling the obtained cloud.
- **SCRG**: Shape Completion enabling Robotic Grasping [6], which is a convolution neural network trained to complete an object's mesh representation. It does not use any prior structural and semantic information. In order to calculate CD, we have converted the resulted voxel into point clouds.

For all the above approaches, we have used already-trained models from the authors, since their extensive experiments were conducted on the same or similar datasets as ours (KITTI [12], ModelNet [11] or ShapeNet [5]).

D. Ablation study

The goal of the ablation study is to show the importance of the different components in our architecture. As illustrated in Fig. 9, we have varied the structures of the CNN layers in the encoder and decoder network, as well as the number of iterations.

For this study, the sizes and number of convolutional layers in the GFPNet's encoder and decoder architecture from Fig. 4 have been varied. We have observed that using only two convolutional layers affects the quality of the generated feature vector, determining the overall modeled surface to be more rigid and thus unable to fold over deformed surfaces. On the other hand, using more than four convolutional layers produces overfitting and increased run-time.

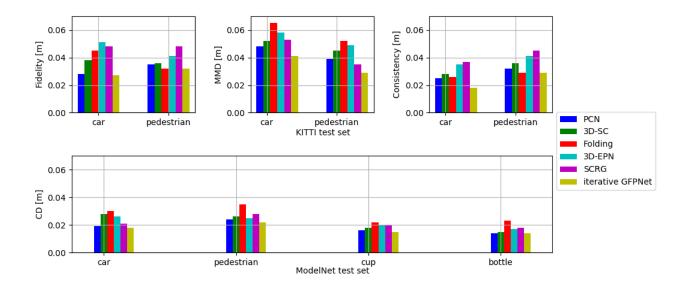


Fig. 8. Quantitative comparison on KITTI (first row) and ModelNet (second row) test datasets. The optimal results correspond to low values for the Chamfer Distance (CD), Fidelity (F), Minimal Matching Distance (MMD) and Consistency (C).

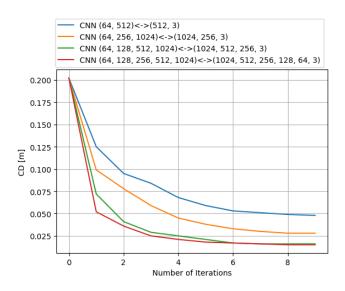


Fig. 9. The value of CD for different numbers and sizes of the CNN layers in the GFPNet architecture. The outcome of the experiment corresponds to an encoder-decoder DNN architecture composed of 4 convolutional layers of size 64, 128, 512 and 1024 and an optimal value of 5 iterations.

To identify the optimal number of convolution layers, we have applied the evolutionary approach described [32]. Aiming to obtain a balance between run-time and precision, we have identified a DNN architectures composed of 4 convolutional layers of size 64, 128, 512 and 1024 for the encoder and decoder, respectively.

In order to determine the optimal number of iterations, we have measured the evolution of CD over an evaluation subset of shapes. The optimal value of 5 iterations was empirically determined.

E. Discussion

The performance of the competing shape completion methods is shown in Fig. 8, taking into account the evaluation metrics presented in Section III-A. The iterative GFPNet,

configured to perform 5 iterations, provided the better results on the majority of shapes from both test sets.

GFPNet obtained similar results to PCN for the cup, bottle and car test shapes, while outperforming it by a considerable margin for the pedestrian shapes. This is because the PCN Folding based multistage decoder considers weak constraints on the local densities. On inputs where the object structure has an occlusion ratio above 60% (e.g. a vehicle seen only from the back), the coarse PCN output fails to produce a correct global shape.

On the other hand, the Folding approach often produces outliers that are not consistent with the global shape. This reflects in high CD, F and MMD values for all compared methods. Similar to GFPNet, the Folding network can be applied also on local surfaces. Nevertheless, in order to achieve an accuracy similar to GFPNet, it requires a considerable higher number of iterations.

The SCRG network has a good performance on regular shapes, such as cups and bottles, but fails on cars and pedestrians. Especially in the completed part, the network produces a very rough approximation mainly due to the lack of prior information on the object class. The same behavior has also been obtained on the 3D-EPN volumetric approach. Also in this case, GFPNet performs better in both the CD, F and MMD measures.

IV. CONCLUSIONS

In this paper, we have introduced GFPNet, which is a 3D volumetric object modeling approach for objects described by incomplete point clouds. Its main goal is to deliver precise 3D shape models and pose estimation for objects present in different real-world scenes. The main novelty of the algorithm is represented by the usage of a deep learning technique for deforming a Generic Primitive (GP) with the purpose of 3D shape completion.

As future work, we plan to apply GFPNet for modeling a larger category of object shapes, as well as to non-point cloud data such as images, where the shape completion operation should be performed solely on 2D visual information.

ACKNOWLEDGMENT

The authors would like to thank Elektrobit Automotive for the infrastructure and research support.

REFERENCES

- J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin, and D. Hoiem, "Completing 3D object shape from one depth image," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015, pp. 2484–2493.
- [2] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. Reid, "Dense reconstruction using 3D object shape priors," in 2013 IEEE Conference on Computer Vision and Pattern Recognition, June 2013, pp. 1288–1295.
- [3] M. Pauly, N. J. Mitra, J. Giesen, M. Gross, and L. J. Guibas, "Example-based 3D scan completion," in *Proceedings of the Third Eurographics Symposium on Geometry Processing*, ser. SGP '05. Eurographics Association, 2005.
- [4] M. Sung, V. G. Kim, R. Angst, and L. Guibas, "Data-driven structural priors for shape completion," ACM Transactions on Graphics (Proc. of SIGGRAPH Asia), 2015.
- [5] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An information-rich 3D model repository," Tech. Rep. arXiv:1512.03012 [cs.GR], 2015.
- [6] J. Varley, C. DeChant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," 09 2017, pp. 2442–2447.
- [7] D. Stutz and A. Geiger, "Learning 3D shape completion from laser scan data with weak supervision," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [8] A. Dai, C. Ruizhongtai, and M. NieBner, "Shape completion using 3D encoder predictor cnns and shape synthesis," 2017, pp. 6545–6554.
- [9] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "PCN: Point completion network," in 2018 International Conference on 3D Vision (3DV), 2018, pp. 728–737.
- [10] T. T. Cocias, F. Moldoveanu, and S. M. Grigorescu, "Generic fitted shapes (GFS): Volumetric object segmentation in service robotics," *Robotics and Autonomous Systems*, vol. 61, no. 9, pp. 960–972, 2019.
- [11] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D ShapeNets: A deep representation for volumetric shapes," in *Proceedings* of 28th IEEE Conference on Computer Vision and Pattern Recognition (CVPR2015), 2015.
- [12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *International Journal of Robotics Research*, 2013.
- [13] S. Thrun and B. Wegbreit, "Shape from symmetry," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2, 2005, pp. 1824–1831 Vol. 2.
- [14] D. T. Nguyen, B. Hua, M. Tran, Q. Pham, and S. Yeung, "A field model for repairing 3D shapes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5676–5684.
- [15] F. Engelmann, J. Stckler, and B. Leibe, "SAMP: Shape and motion priors for 4D vehicle reconstruction," in 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), 2017, pp. 400–408.
- [16] J. Ahlberg, "Active contours in three dimensions," 1996.
- [17] T. T. Cocias, S. M. Grigorescu, and F. Moldoveanu, "3D structure estimation from a single view using generic fitted primitives (GFP)," in *Computer Vision, Imaging and Computer Graphics. Theory and Application*. Springer Berlin Heidelberg, 2013, pp. 369–382.
- [18] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Interpretable unsupervised learning on 3D point clouds," *CoRR*, vol. abs/1712.07262, 2017.
- [19] D. Stutz and A. Geiger, "Learning 3d shape completion under weak supervision," *International Journal of Computer Vision*, 10 2018.
- [20] F. Engelmann, J. Stückler, and B. Leibe, "Joint object pose estimation and shape reconstruction in urban street scenes using 3D shape priors," in *Proc. of the German Conference on Pattern Recognition (GCPR)*, 2016.
- [21] Q.-H. Pham, M.-K. Tran, W. Li, S. Xiang, A. Liu, Y. Su, M.-T. Tran, N.-M. Bui, T.-L. Do, T. V. Ninh, T.-K. Le, A.-V. Dao, V.-T. Nguyen, M. N. Do, A.-D. Duong, B.-S. Hua, L.-F. Yu, D. T. Nguyen, and S.-K. Yeung, "RGB-D object-to-CAD retrieval," in *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2018.

- [22] F. Steinke, B. Schlkopf, and V. Blanz, "Support vector machines for 3D shape processing," *Computer Graphics Forum*, vol. 24, no. 3, pp. 285–294, 2005.
- [23] M. Yumer and N. Mitra, "Learning semantic deformation flows with 3D convolutional networks," in *Computer Vision – ECCV 2016*, 2016, pp. 294–311.
- [24] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, June 16-20, 2019*, pp. 770–779.
- [25] V. Sarode, X. Li, H. Goforth, Y. Aoki, R. A. Srivatsan, S. Lucey, and H. Choset, "PCRNet: Point cloud registration network using PointNet encoding," *CoRR*, vol. abs/1908.07906, 2019.
- [26] P. Besl and N. McKay, "A method for registration of 3D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 239–256, 1992.
- [27] I. Dryden and K. Mardia, Statistical shape analysis, ser. Wiley series in probability and statistics. Wiley, 1998.
- [28] D. Levin, "Mesh-independent surface interpolation," in *Geometric Modeling for Scientific Visualization*. Springer Berlin Heidelberg, 2004, pp. 37–49.
- [29] F. Gney and A. Geiger, "Displets: Resolving stereo ambiguities using object knowledge," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 4165–4175.
- [30] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," International Conference on Learning Representations, 12 2014.
- [31] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, vol. 9, 2010, pp. 249–256.
- [32] S. M. Grigorescu, B. Trasnea, L. Marina, A. Vasilcoi, and T. Cocias, "Neurotrajectory: A neuroevolutionary approach to local state trajectory learning for autonomous vehicles," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3441–3448, Oct 2019.



Tiberiu Cocias received the Ph.D. degree in System Engineering in 2013 and the Dipl.-Eng. degree in Control Engineering and Computer Science in 2009 both from Transilvania University of Brasov, Romania. He is a member of the ROVIS team, as well as team manager in the Artificial Intelligence Tech Centre at Elektrobit Automotive Romania. His main areas of research are 3D perception, 3D object reconstruction, object segmentation, volumetric shape modelling and object volume approximation. Since 2015 he is a lecturer at the Department of

Automation, Transilvania University of Brasov, Romania, where he teaches Robotics, Expert Systems and Computer Programming.



Alexandru Razvant received his Dipl.-Eng. degree in Information Technology in 2019 from Transilvania University of Brasov, Romania. He is also affiliated with the Artificial Intelligence Tech Centre at Elektrobit Automotive Romania, where he is a software developer. His main areas of research are 3D perception, 3D object reconstruction, object segmentation, volumetric shape modelling and object volume approximation.



Sorin Grigorescu received the Ph.D. degree in Robotics from the University of Bremen, Germany, in 2010 and the Dipl.-Eng. degree in Control Engineering and Computer Science from Transilvania University of Brasov, Romania, in 2006. Sorin is an associate professor at the Department of Automation, Transilvania University of Brasov, Romania. Since June 2013, he is also affiliated with Elektrobit Automotive, where he is the global Head of Artificial Intelligence (AI), coordinating an international team of AI researchers working on deep learning

solutions for the automotive industry. His research interests are Computer Vision, Artificial Intelligence for Autonomous Vehicles, Learning Controllers, Rehabilitation Robotics and Feedback Control in Computer Vision.