Future Urban Scenes Generation Through Vehicles Synthesis

Alessandro Simoni, Luca Bergamini, Andrea Palazzi, Simone Calderara, Rita Cucchiara
University of Modena and Reggio Emilia, Modena, Italy
{alessandro.simoni, luca.bergamini24, andrea.palazzi, simone.calderara, rita.cucchiara}@unimore.it

***Abstract*—In this work we propose a deep learning pipeline to predict the visual future appearance of an urban scene. Despite recent advances, generating the entire scene in an end-to-end fashion is still far from being achieved. Instead, here we follow a two stages approach, where interpretable information is included in the loop and each actor is modelled independently. We leverage a per-object *novel view synthesis* paradigm; i.e. generating a synthetic representation of an object undergoing a geometrical roto-translation in the 3D space. Our model can be easily conditioned with constraints (e.g. input trajectories) provided by state-of-the-art tracking methods or by the user itself. This allows us to generate a set of diverse realistic futures starting from the same input in a *multi-modal* fashion. We visually and quantitatively show the superiority of this approach over traditional end-to-end scene-generation methods on CityFlow, a challenging real world dataset.**

I. Introduction

In the near future, smart interconnected cities will become reality in various countries worldwide. In this scenario, vehicles – both autonomous and not – will play a fundamental role thanks to key technologies developed to connect them (e.g. 5G) and advanced sensors (e.g. lidars, radars) enabling a deeper understanding of the scene. Explainability is expected to be a mandatory requirement to ensure the safeness of all other actors (including pedestrians, cyclists, . . .). However, the current approach to autonomous driving related tasks is still end-to-end, which greatly obscures the learned knowledge. Despite that, recent works [1], [2] have moved from this framework – where raw inputs are transformed into the final outputs/decision – to a more interpretable one, where an intermediate high level representation is employed. Those representations can be easily understood by human operators and provide an effective parallelism between human and autonomous decision taking.

In this work we take a step forward and present a pipeline where the final output is produced by applying a sequence of operations that mimic those of a human operator for a specific task related to the autonomous driving. In particular, we focus on generating realistic visual futures for urban scenes where vehicles are the main actors. In more details, starting from one or multiple RGB frames, the final output is a clip of images where all the actors in the scene move following a plausible path. In doing so, as depicted in Fig. 1, we rely heavily on information a user can easily understand, such as bounding boxes, trajectories and keypoints. Moreover, we wish to easily condition the output on that information; in particular, given a set of trajectories for the same vehicle (either by a state-of-the-art trajectory predictor or a user's input), we would like to generate a set of realistic visual representation of the vehicle following these trajectories.

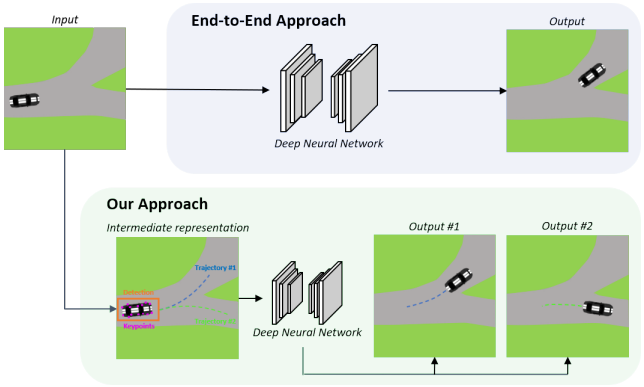

Fig. 1. The difference between a black box end-to-end method and our approach, which exploits intermediate interpretable information to synthesise each vehicle individually.

Explainability is not the only setting where future visual scene generation is expected to be applied. In fact, forensic studies often require to simulate realistic scenes based on high level information provided by an user (e.g. vehicles detections and trajectories). Moreover, state-of-the-art methods working on images could be immediately applied to the final output of our pipeline. In the following, we focus on vehicles only and leave the analysis of other agents as future work.

It is worth noting how the same task can be tackled as an image-to-image problem, where a deep neural network transforms past frame/s into future ones, as depicted in figure 1. While many end-to-end methods [3], [4], [5], [6] can in fact be applied to visual scene generation, they all share some intrinsic drawbacks. In particular: *i)* because they start from raw inputs (i.e. RGB images), it is not always clear which is the best way to include user's or geometric constraints; *ii)* despite recent advances in model explanation [7], [8], end-to-end methods are difficult to investigate either before or after critical faults, which is required for critical applications; *iii)* these methods do not focus on the actors but instead transform the entire image, including the static background: this wastes computational time while limiting the maximum resolution that these methods can handle; and *iv)* they can hardly leverage any established state-of-the-art method for

additional information, such as vehicle detection or trajectory prediction.

Contrarily, we frame the task as a two stages pipeline where only vehicles are individually transformed. First, we extract interpretable information from raw RGB frames, including bounding boxes and trajectory estimations. Second, we employ it to produce visual intermediate inputs. Finally, these inputs condition a deep convolutional neural network [9], [10] to generate the final visual appearance of the vehicle in the future. We argue that this approach is closer to the human way of thinking and, as such, better suits a human-vehicle interactions setting. Similarly to what [1], [2] devise for autonomous planning, our method offers an interpretable intermediate representation a user can naturally understand and interact with. Finally, the input resolution does not represent a limit in our proposal. In fact, as only individual vehicles are processed in our pipeline, the input resolution is typically much lower than the full frame one.

To sum up, we:

- Provide a novel pipeline that leverage interpretable information to produce a deterministic visual future grounded on those constraints;
- prove that our method is not limited to a uni-modal output, but allows to generate *"alternative futures"* by acting on the intermediate constraints;
- show how this approach outperforms end-to-end image-to-image translation solutions both visually and quantitatively.

II. Related works

We first introduce here the current state of the art for image-to-image translation, where one or multiple images are produced starting from a single or a set of input frames. These methods focus on the entire scene, without acknowledging specific elements in the scene. We then focus on approaches based on view synthesis, where the attention is placed instead on the actor solely, with the aim of producing a novel view of it from a different point of view.

Image to Image Translation. Generative Adversarial Networks (GANs) [3], [11], [12], [13], [14] have been widely used to perform image transformations with impressive results. They exploit an *adversarial loss* to constrain generated images to be as similar as possible to the real ones. This supervision signal generates sharper results when compared with standard maximum estimation based losses, and allows these methods to be employed for image generation and editing tasks associated with computer graphics.

Recent works [15], [4], [5] prove that GANs can help solving conditioned image generation, where the network yields an output image conditioned on an observed input image x and an optional random noise z. This can be applied for example to transform a segmentation map into image, or a picture taken at day time into one acquired at night time as presented by [4].

Wang et al. [6] propose a framework able to synthesise high-resolution images (*pix2pixHD*), while Zhu et al. [5] define the concept of *cycle consistency loss* to supervise GANs training without the need for coupled data; their goal is to define a function *G*, which maps from the first domain to the second, and a function *F*, which performs the opposite. The two domains are bound to be consistent with each other at training time.

With the aim of predicting multiple frames, several works [16], [17], [18], [19] extend the image-to-image approach by including time. Authors of PredNet [16] propose a network based on Long Short Term Memory (LSTM) [20] combined with convolutional operations to extract features from input images. In [17], an LSTM based network is trained without any additional information (e.g. optical flow. segmentation masks,...) by leveraging the concept of "network capacity maximisation". Qi et al. [18] decompose the task of video prediction into ego and foreground motion, and leverage RGBD input for 3D decomposition. Finally, authors from [19] address the issue of low quality predictions for distant future by training a network to predict both future and past frames and by enforcing retrospective consistency.

View synthesis. In the last few years, deep generative models have been applied also to novel view generation, i.e. synthesising the aspect of an object from different points of view. Many works [21], [22] achieve impressive results on human pose appearance generation. Among them, VUnet [9] is based on a U-Net architecture [23] which combines a GAN mapping an estimated shape y to the target image x with a Variatonal AutoEncoder (VAE) [24] that conditions on the appearance z. This network aims to find the maximum a posteriori $p(x|y,z)$, i.e. the best object synthesis conditioned on both appearance and shape constraints. Yang et al. [25] propose a recurrent encoder-decoder network to learn how to generate different views of the same object from different rotations. The initial object appearance is encoded into a fixed low dimensional representation, while sequential transformations act on a separate embedding. Finally, the decoder combines both vectors and yields the final prediction.

In the automotive field, Tatarchenko et al. [26] train a CNN to estimate the appearance and the depth map of an object after a viewpoint transformation. The transformation is encoded as azimuth-elevation-radius and is concatenated to the appearance embedding after being forwarded through fully connected layers. By combining multiple predicted depth maps, their approach can generate reconstructed 3D models from a single RGB image. Again, Zhou et al. [27] extract appearance flow information to guide pixels locations after an arbitrary rotation. Their model leverages a spatial transformer [28] to output a grid of translation coefficients. Contrarily, Warp&Learn [10] first extracts 2D semantic patches from the vehicle input image and warps them to the output viewpoint by means of an affine transformation. Then, an image completion network is employed to seamlessly merge the warped patches and produce the final result. Park et al. [29] draw inspiration from [27] to relocate pixels visible both in the input and target view before using an image completion network based on adversarial training to refine the intermediate result.

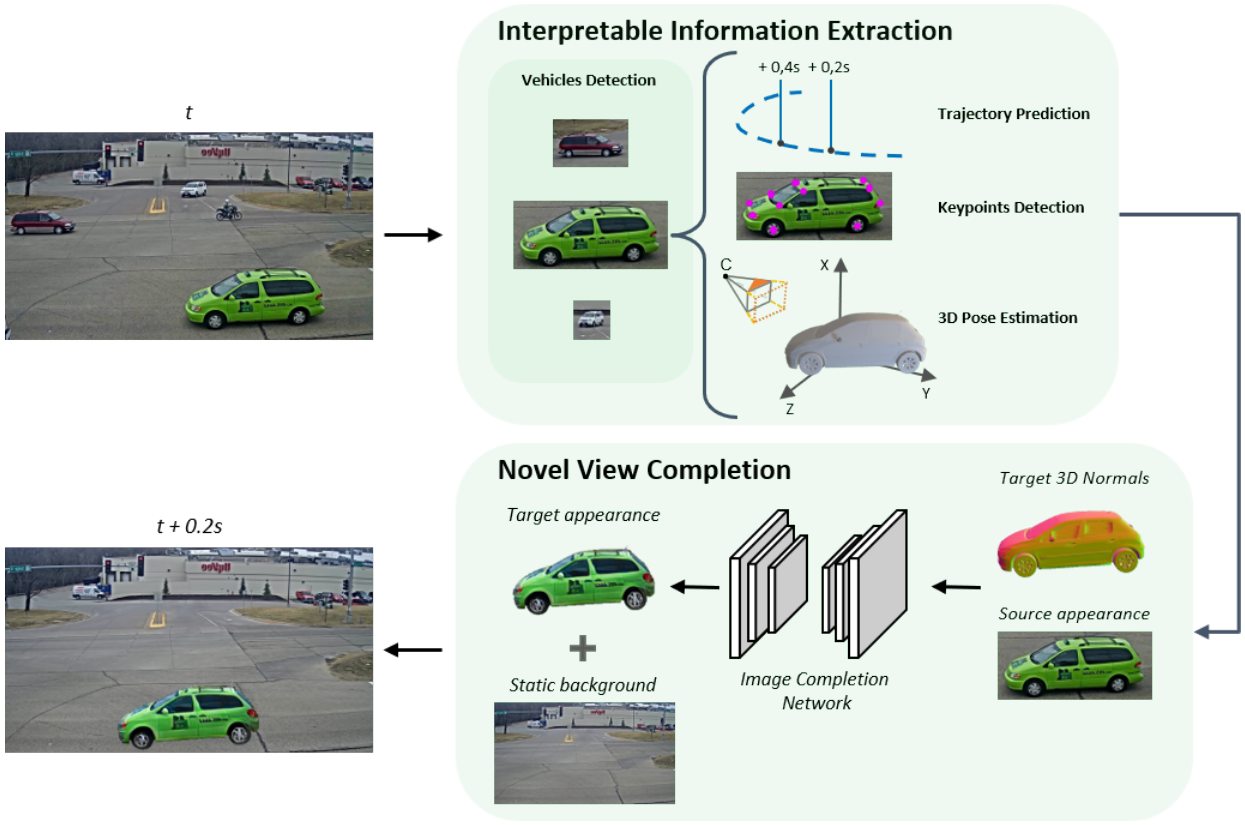

Fig. 2. Our model pipeline composed by two stages: (i) *interpretable information extraction* for each vehicle (detection & tracking), and (ii) *novel view completion* process exploiting the 3D projected rendering of the object (**target 3D normals**) and its appearance from the cropped image (**source appearance**).

III. Model

We present here the two fundamental stages of our approach, as illustrated in Fig. 2. In the first one *(interpretable information extraction)*, we focus on acquiring high level interpretable information for each vehicle in the scene. That information is then exploited by the second stage (*novel view completion*) to generate the final appearance of each vehicle individually.

A. Interpretable Information Extraction

During this stage, high level interpretable information is gathered from raw RGB frames. Vehicles are first detected and their trajectories predicted. However, these trajectories are bound to the 2D plane, which is not sufficient to produce realistic movements (e.g. a car taking a turn). As such, we also detect vehicle 2D keypoints and align them to 3D ones by means of a perspective-n-point algorithm, obtaining a roto-translation matrix. This way, we can lift both the vehicle and the trajectory from 2D to 3D, and simulate realistic movements.

Components from this stage are not the focus of this work. In fact, we're not interested in advancing the research in any of these tasks here, and we make use of pre-trained state-of-the-art methods when possible.

1) Vehicle Detection: We employ SSD detection network [30] to detect vehicles in the scene. Starting from the input frame, SSD outputs a set of bounding boxes (one for each detected object) in a single forward, along with their class probabilities. We filter the bounding boxes to keep only those associated with a vehicle, and use them to crop the visual appearance of each of them.

2) Trajectory Prediction: We employ TrackletNet [31] as a trajectory predictor; it compares each vehicle tracklet – composed by the detected bounding box and the appearance features – along a time window of 64 consecutive frames. Using a similarity measure between tracklets, a graph is created where vertices under a certain distance threshold represent the same object.

3) Keypoints Localisation: We adapt a state-of-the-art network for human pose estimation, namely *Stacked Hourglass* [32], to localise vehicle keypoints. The network is characterised by a tunable number of encoder-decoder stacks. The final decoder outputs a set of planes (one per keypoint) where the maximum value localises the keypoint location. We change the final output structure to produce 12 keypoints: (i) four wheels, (ii) four lights, and (iii) four front and back windshield corners.

4) Pose estimation: We frame the vehicle pose estimation as a *perspective-n-point* problem, leveraging correspondences between 2D and 3D keypoints. While the former are the outputs of the previous step, the latter come from annotated 3D vehicle models. We exploit the 10 annotated models included

in Pascal3D+, and we train a VGG19-based network [33] to predict the correspondent model given the vehicle crop. We argue these 10 CADs cover the vast majority of urban vehicles, as they have been deemed sufficient to annotate all vehicle images in the Pascal3D+ car set by authors from [34]. Then, we adopt a *Levenberg-Marquardt* [35] iterative optimization strategy to find the best roto-translation parameters by minimizing the reprojection error or *residual* between the 2D original keypoints and the correspondent 3D projections. We follow the stop criteria presented in [35]. Once the source roto-translation matrix V_s is known, the predicted model can follow the 3D lifted trajectory by applying consecutive transformations defined by the vehicle trajectory – i.e. the roto-translation between consecutive trajectory positions converted from pixel to GPS meter coordinates. After each transformation we obtain the target roto-translation matrix V_t.

B. Novel View Completion

Once we know what to move and where to move it, we require a method to condition a reprojected 3D model with the original 2D appearance from the vehicle detection module. Theoretically, any view synthesis approach from Sec. II can be used. In practice, a vast majority of them [28], [25], [26], [29] is only able to handle a specific setting known as *"look-at-camera"*, where the vehicle is placed in the origin and the camera z axis points at it. However, in our setting both V_s and V_t are generic roto-translation matrices. Moreover, some of the methods involve voxel spaces [28], which makes infeasible finding a correspondence.

Because our focus is on real-world data, we also exclude works which require direct training supervision and can thus only been trained on synthetic data [25]. In fact, as of today no real-world vehicles dataset can be exploited for supervised novel view synthesis training, as they all lack multiple views for the same vehicle annotated with pose information. This also prevent us from using any method based on [6] for this task. In the following, we thus employ two approaches [9], [10] that are able to handle generic transformations and can be trained in an unsupervised fashion on real-world data.

Giving as input the crop depicting the vehicle x_s observed by a source camera viewpoint V_s, we project the 3D model with the roto-translation outlined by V_t. To enrich the representation, we render a $2.5D$ sketch with normal information. The newly produced output is then pasted into a static background, and the process repeated for each moving vehicle.

We rely on foreground suppression to generate a static background for the output clip. We also experimented with inpainting networks [36] but found that results were less realistic by visual inspection due to the presence of several artefacts. We leave further investigation and the extension to moving cameras as a future work.

IV. Experiments

In this section we present, both visually and quantitatively, the results of our proposed pipeline and we compare them with those from various end-to-end approaches (referred to as baselines in the following). We also introduce the employed datasets and the metrics of interest, as well as implementation details to ensure experiments reproducibility.

A. Datasets

1) CityFlow [37]: is a multi-target multi-camera tracking and re-identification vehicle dataset, introduced for the 2019 Nvidia AI City Challenge. It comprises more than 3 hours of high resolution traffic cameras videos with more than $200K$ bounding boxes and 600 vehicles identities, split between train and test sets. The dataset also includes homography matrices for bird's eye visualisation. Vehicles detection and tracking have been annotated automatically using SSD [30] and TrackletNet [31] as detector and tracker respectively. All baselines have been trained on the train split of this dataset.

2) Pascal3D+ [34]: is composed of 4081 training and 1024 testing images, preprocessed to guarantee the vehicle is completely visible. Every image is also classified into one of ten 3D models. Both 3D and 2D keypoints are included. Because 2D keypoints localisation is crucial in our pipeline, we extend the Pascal training set by including frames from CarFusion [38]. We train models for our first stage on this dataset to ensure the generalisation of our approach when tested on CityFlow.

B. Evaluation Metrics

We evaluate all methods using both pixel level and perceptual metrics. The former evaluates the exact spatial correspondence between the predicted and the ground truth target and are very sensitive to 2D transformations (e.g. translations). Contrarily, the latter evaluates the matching between the content of the two images. It is worth noting that we only compare a tight crop around each vehicle for all methods, instead of the full generated image. We argue this choice leads to a better understanding of true performance, because it removes a vast portion of the image – with only static background in it – we are not interested in evaluating.

1) Pixel wise Metrics: We employ the Mean Squared Error (MSE) as a measure of pixel distance between the target crop x_t and the predicted one x_p as follows:

$$MSE(x_t, x_p) = \|x_t - x_p\|_2^2 \tag{1}$$

values are then averaged over to compute the final score.

2) Perceptual Metrics: We employ the Structural Similarity Index (SSIM) [39] as a measure of the degradation in the image quality due to image data manipulation, defined as:

$$SSIM(x_t, x_p) = \frac{(2\mu_{x_t}\mu_{x_p} + c_1)(2\sigma_{x_t x_p} + c_2)}{(\mu_{x_t}^2 + \mu_{x_p}^2 + c_1)(\sigma_{x_t}^2 + \sigma_{x_p}^2 + c_2)} \tag{2}$$

As another measure of content similarity, we measure the Fréchet Inception Distance (FID) [40], [41] computed between activations from the last convolutional layer of an InceptionV3 model pretrained on ImageNet [42]. We compute the FID as follow:

$$FID = \|m_t - m_p\|_2^2 + Tr\big(C_t + C_p - 2\big(C_t C_p\big)^{1/2}\big) \tag{3}$$

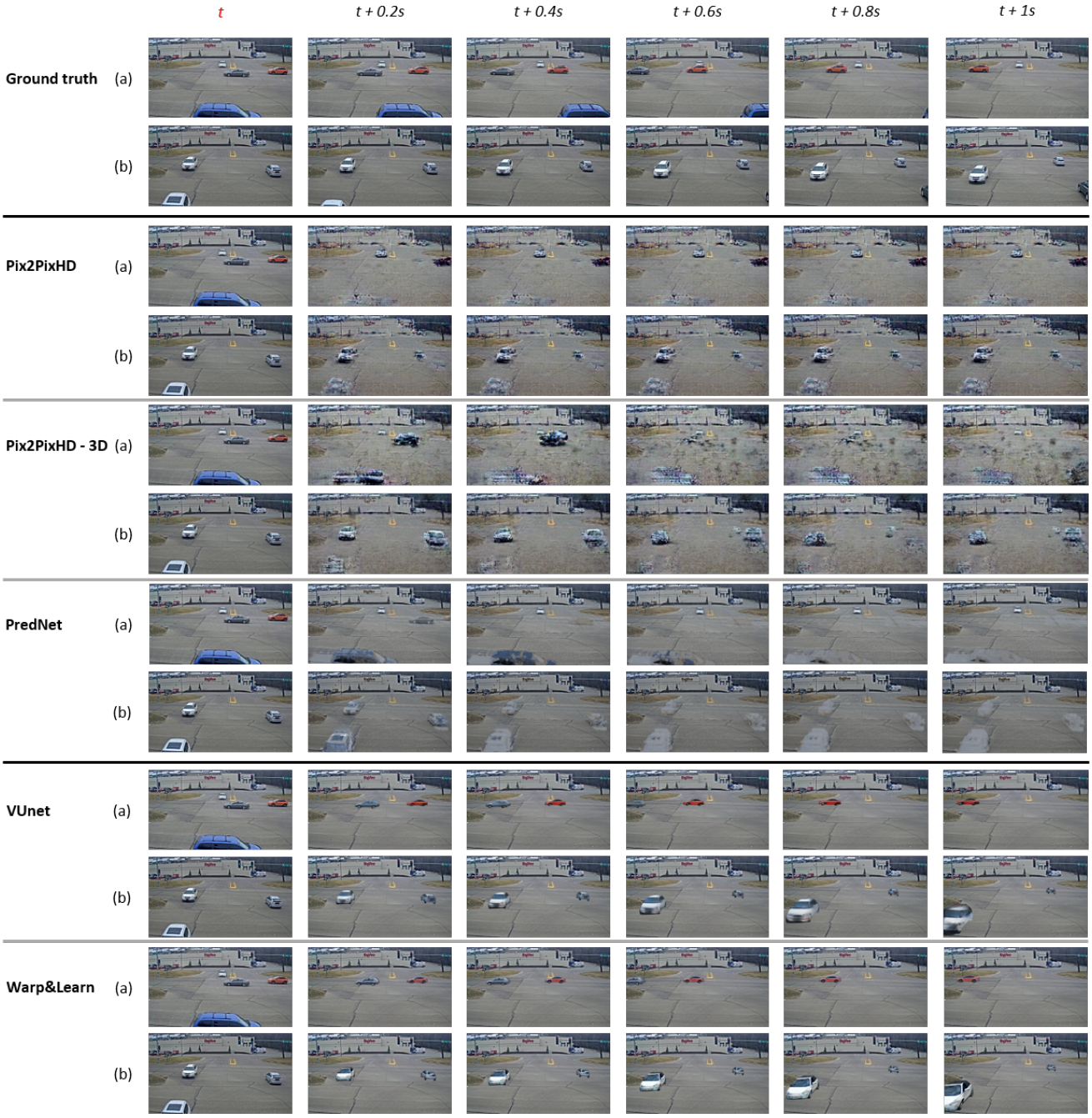

Fig. 3. Visual results using the methods (Pix2PixHD, Pix2PixHD-3D, PredNet, VUnet, Warp&Learn) on two ground truth video sequences (a) and (b) with different vehicles behaviour. Images at time t refer to the ground truth, while images within 1 second in the future represent a method prediction.

where m, C refer to the mean and covariance and follow the same notation as above for target and predicted image.

Finally, we also compute the Inception Score (IS) [13] to measure the generated images variety as:

$$IS(G) = exp(\frac{1}{N}\sum_{i=1}^{N} D_{KL}(p(y \mid x^{(i)}) \| \hat{p}(y))) \quad (4)$$

where x is an image, N the total number of samples and $p(\hat{y})$ an empirical marginal class distribution.

C. Baselines

1) Pix2Pix: We adapt Pix2PixHD [6] for future frame prediction. Because it is trained in an end-to-end fashion, we can trivially include any high level information in the input. Still, we need to condition somehow the output to generate a specific frame (e.g. 0.2 seconds in the future) given the input image. As such, we stack the input with a set of binary maps along the channels dimension. During training, a random frame in the future is selected and the correspondent map is

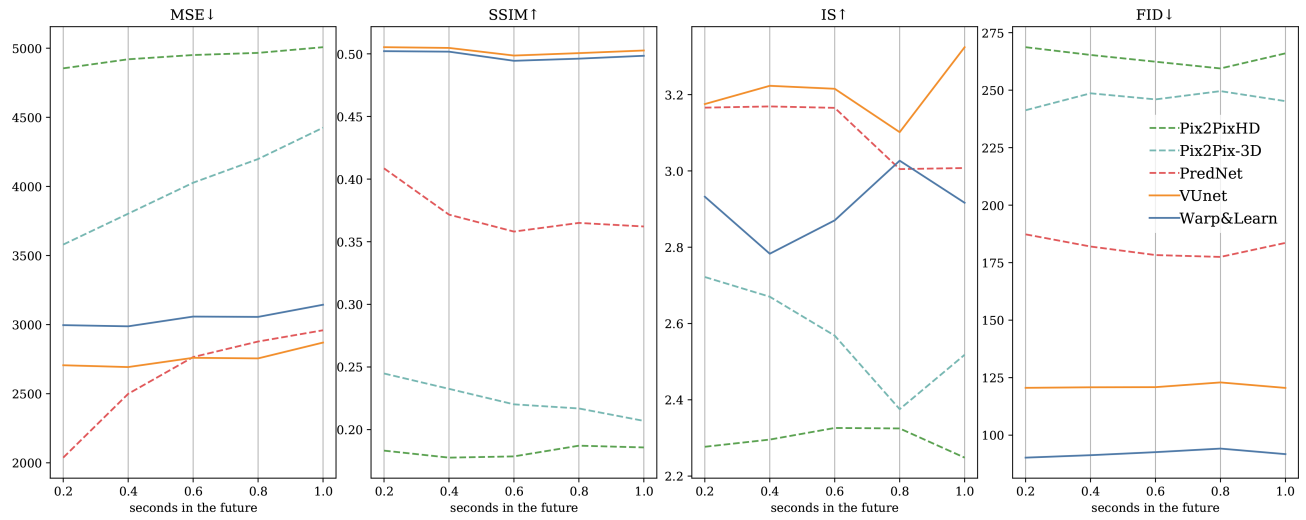

Fig. 4. Comparison between our approach with two different types of image completion network (solid lines) and multiple baselines (dash lines) using Mean Squared Error (MSE) (lower is better), Structural Similarity Index (SSIM) (higher is better), Inception Score (IS) (higher the better) and Frechet Inception Distance (FID) (lower is better).

TABLE I
COMPARISON ON THE TEST SET USING MEAN SQUARED ERROR (MSE). EACH COLUMN REFERS TO A FUTURE DISPLACE. LOWER IS BETTER.

Method	*+0.2s*	*+0.4s*	*+0.6s*	*+0.8s*	*+1.0s*
Pix2PixHD [6]	4854	4919	4950	4966	5007
Pix2Pix-3D	3579	3802	4026	4198	4424
PredNet [16]	**2037**	**2499**	2765	2877	2959
Our(VUnet [9])	2705	2692	**2759**	**2755**	**2870**
Our(Warp&Learn [10])	2996	2987	3058	3055	3153

TABLE II
COMPARISON ON THE TEST SET USING STRUCTURAL SIMILARITY INDEX (SSIM). EACH COLUMN REFERS TO A FUTURE DISPLACE. HIGHER IS BETTER.

Method	*+0.2s*	*+0.4s*	*+0.6s*	*+0.8s*	*+1.0s*
Pix2PixHD [6]	0.18	0.17	0.17	0.18	0.18
Pix2Pix-3D	0.24	0.23	0.22	0.21	0.20
PredNet [16]	0.40	0.37	0.35	0.36	0.36
Our(VUnet [9])	**0.50**	**0.50**	**0.49**	**0.50**	**0.50**
Our(Warp&Learn [10])	**0.50**	**0.50**	**0.49**	0.49	0.49

TABLE III
COMPARISON ON THE TEST SET USING INCEPTION SCORE (IS). EACH COLUMN REFERS TO A FUTURE DISPLACE. HIGHER IS BETTER

Method	*+0.2s*	*+0.4s*	*+0.6s*	*+0.8s*	*+1.0s*
Pix2PixHD [6]	2.27	2.29	2.32	2.32	2.24
Pix2Pix-3D	2.72	2.67	2.56	2.37	2.51
PredNet [16]	3.16	3.16	3.16	3.00	3.00
Our(VUnet [9])	**3.17**	**3.22**	**3.21**	**3.10**	**3.32**
Our(Warp&Learn [10])	2.93	2.78	2.87	3.02	2.91

TABLE IV
COMPARISON ON THE TEST SET USING FRECHET INCEPTION DISTANCE (FID). EACH COLUMN REFERS TO A FUTURE DISPLACE. LOWER IS BETTER

Method	*+0.2s*	*+0.4s*	*+0.6s*	*+0.8s*	*+1.0s*
Pix2PixHD [6]	274.2	268.6	265.3	262.3	259.4
Pix2Pix-3D	240.6	241.2	248.6	245.9	249.5
PredNet [16]	197.1	197.2	196.4	193.4	196.3
Our(VUnet [9])	192.8	187.3	182.0	178.3	177.49
Our(Warp&Learn [10])	**90.4**	**90.22**	**91.2**	**92.6**	**94.1**

set to 1, while the others are set to 0. It is worth noting that predicting movements given a single input image is clearly an ill-posed task. For this reason, we also include another Pix2Pix baseline – referred as Pix2Pix-3D in the following – which is time aware. We provide this baseline with a set of past frame, and replace 2D convolution in the encoder with 3D ones. As such, this baseline version has access to past frames and can therefore exploit temporal information to determine if and how a vehicle is moving. However, this comes with an increase in memory footprint.

2) PredNet: We also adapt PredNet [16] as a recurrent-based approach to the task. Differently from the previous two, this baseline generates frames in the future via a recurrent structure. However, generated frames have to be forwarded as part of the input to produce frames further in the future, causing errors to propagate and performance to degrade in the long run.

D. Implementation Details

All baselines are trained for 150 epochs on frames from the Cityflow train set, resized to 640x352 pixels. Pix2PixHD-based models employ batch size equal to 4 with initial learning rate of $2e^{-4}$ and linear decay as defined in [4]. PredNet is trained according to the original paper parameters.

As for our pipeline, the Keypoints localisation network is trained for 100 epochs employing batch size 10 and a learning rate of $1e^{-3}$ halved every 20 epochs, while VUnet and Warp&Learn models are trained following the policies described respectively in [9], [10]. All models except those

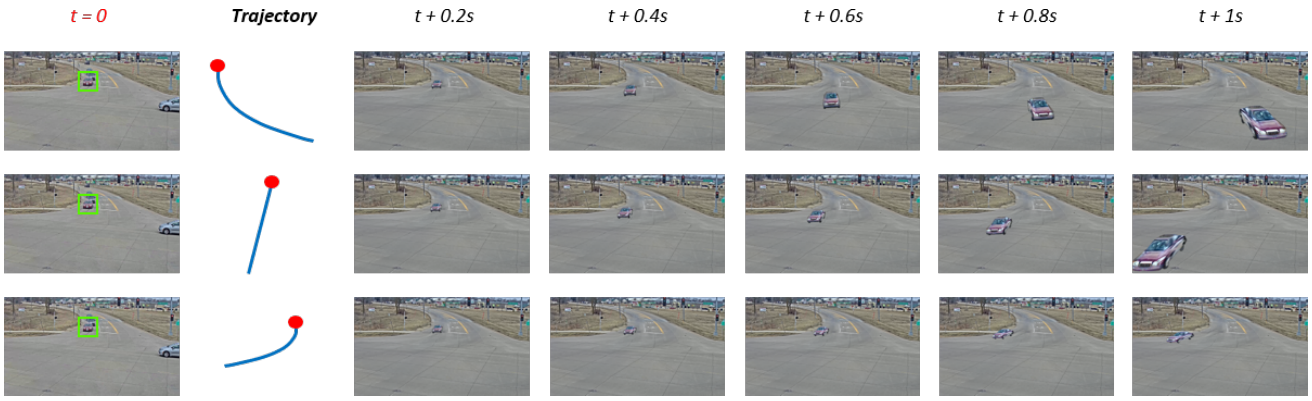

Fig. 5. Visual results from our approach for three plausible futures constrained over different trajectories. The detected vehicle follows the indicated trajectory while preserving the original appearance. Best viewed in color.

for detection and tracking are trained on Pascal3D+ vehicle images resized to 256x256 pixels.
Code has been developed using the PyTorch [43] framework and the Open3D library [44] has been employed to manipulate and render the 3D CAD in the scene. Inference is performed on Cityflow test set videos resized to 1280x720 pixels. Code is available at https://github.com/alexj94/future_urban_scene_generation.

E. Results

Comparisons of the different methods are reported in Tables I, II, III, IV and in Figure 4. Our proposed approach outperforms the baselines for all metrics in the long run, while scores second behind PredNet for the first two predictions according to the MSE.

However, it is worth noting how Prednet is not capturing movements in an effective way, as shown in Figure 3. While first outputs looks realistic, performance degrades quickly when predictions are employed as inputs for the LSTM. Our approach proves to be superior for all the metrics that reward the content realism (i.e. FID, IS and SSIM) and to suffer less performance degradation for long time predictions. This highlights how focusing on individual vehicles is crucial in visual future scene prediction. Between the two novel view synthesis methods, [9] achieves better performance for 2 (IS, MSE) out of 4 metrics, with comparable results for the SSIM. On the other hand [10] outperforms all other methods by a consistent margin for the FID metric.

Figure 3 reports a visual comparison between different methods on two ground truth sequences. It can be appreciated how our approach produces higher quality results, both for the static background and the foreground. On the other hand, baseline methods struggle to produce crisp images, often resulting in extremely blurry images. As expected, Pix2PixHD fails completely to predict vehicles movement and collapses into a static image output. While Pix2Pix-3D partially solves this issue, it still focus mostly on the background. Finally, PredNet is able to guess correctly the evolution of the scene, but performance degrade in the long run, with vehicles progressively fading away. It is worth noting that for the first sequence the vehicle closer to the camera is not modelled by our method, and thus disappears immediately. This is due to an SSD miss-detection. Even though our final output depends on many modules we argue this is not a weakness in the long run. In fact, it's trivial to replace a single component with another with better performance, while the same consideration does not hold true for end-to-end approaches.

F. Constrained Futures Generation

Thanks to its two-stages pipeline our methods can be trivially constrained using high-level interpretable information. Figure 5 illustrate an example of this process where the constraint is provided in the form of trajectories. Starting from the same input frame, three futures are generated by providing different trajectories. It can be appreciated how the vehicle closely follows the designated path, which can be easily drawn by a non-expert user. Other constrains that can be provided out-of-the-box include a different CAD model or a different appearance. The same interaction is not well-defined for end-to-end methods.

V. Conclusions

In this work we presented a novel pipeline for predicting the visual future appearance of an urban scene. We propose a novel approach as an alternative to end-to-end solutions, where human interpretable information is included into the loop and every actor is modelled independently. Existing state-of-the-art methods or the user can both be sources for that information. Furthermore, the final visual output is conditioned onto that by design. We demonstrate the performance superiority of our pipeline with respect to traditional end-to-end baselines through an extensive experimental section. Moreover, as shown in Fig. 5, we visually illustrate how our method can generate diverse realistic futures starting from the same input by varying the provided interpretable information.

References

[1] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," *arXiv preprint arXiv:1812.03079*, 2018.

[2] J. Hong, B. Sapp, and J. Philbin, "Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8454–8462.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[4] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[5] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[6] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8798–8807.

[7] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[8] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

[9] P. Esser, E. Sutter, and B. Ommer, "A variational u-net for conditional appearance and shape generation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8857–8866.

[10] A. Palazzi, L. Bergamini, S. Calderara, and R. Cucchiara, "Warp and learn: Novel views generation for vehicles and other objects," *arXiv preprint arXiv:1907.10634*, 2019.

[11] E. L. Denton, S. Chintala, R. Fergus *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, 2015, pp. 1486–1494.

[12] M. F. Mathieu, J. J. Zhao, J. Zhao, A. Ramesh, P. Sprechmann, and Y. LeCun, "Disentangling factors of variation in deep representation using adversarial training," in *Advances in neural information processing systems*, 2016, pp. 5040–5048.

[13] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in neural information processing systems*, 2016, pp. 2234–2242.

[14] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 214, 223.

[15] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[16] W. Lotter, G. Kreiman, and D. Cox, "Deep predictive coding networks for video prediction and unsupervised learning," *arXiv preprint arXiv:1605.08104*, 2016.

[17] R. Villegas, A. Pathak, H. Kannan, D. Erhan, Q. V. Le, and H. Lee, "High fidelity video prediction with large stochastic recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 81–91.

[18] X. Qi, Z. Liu, Q. Chen, and J. Jia, "3d motion decomposition for rgbd future dynamic scene synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7673–7682.

[19] Y.-H. Kwon and M.-G. Park, "Predicting future frames using retrospective cycle gan," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1811–1820.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool, "Pose guided person image generation," in *Advances in Neural Information Processing Systems*, 2017, pp. 406–416.

[22] B. Zhao, X. Wu, Z.-Q. Cheng, H. Liu, Z. Jie, and J. Feng, "Multi-view image generation from a single-view," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 383–391.

[23] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[24] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[25] J. Yang, S. E. Reed, M.-H. Yang, and H. Lee, "Weakly-supervised disentangling with recurrent transformations for 3d view synthesis," in *Advances in Neural Information Processing Systems*, 2015, pp. 1099–1107.

[26] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Multi-view 3d models from single images with a convolutional network," in *European Conference on Computer Vision*. Springer, 2016, pp. 322–337.

[27] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros, "View synthesis by appearance flow," in *European conference on computer vision*. Springer, 2016, pp. 286–301.

[28] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in neural information processing systems*, 2015, pp. 2017–2025.

[29] E. Park, J. Yang, E. Yumer, D. Ceylan, and A. C. Berg, "Transformation-grounded image generation network for novel 3d view synthesis," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2017, pp. 3500–3509.

[30] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[31] G. Wang, Y. Wang, H. Zhang, R. Gu, and J.-N. Hwang, "Exploit the connectivity: Multi-object tracking with trackletnet," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 482–490.

[32] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European conference on computer vision*. Springer, 2016, pp. 483–499.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[34] Y. Xiang, R. Mottaghi, and S. Savarese, "Beyond pascal: A benchmark for 3d object detection in the wild," in *IEEE winter conference on applications of computer vision*. IEEE, 2014, pp. 75–82.

[35] M. Darcis, W. Swinkels, A. E. Güzel, and L. Claesen, "Poselab: A levenberg-marquardt based prototyping environment for camera pose estimation," in *2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. IEEE, 2018, pp. 1–6.

[36] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi, and M. Ebrahimi, "Edgeconnect: Generative image inpainting with adversarial edge learning," *arXiv preprint arXiv:1901.00212*, 2019.

[37] Z. Tang, M. Naphade, M.-Y. Liu, X. Yang, S. Birchfield, S. Wang, R. Kumar, D. Anastasiu, and J.-N. Hwang, "Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8797–8806.

[38] N. Dinesh Reddy, M. Vo, and S. G. Narasimhan, "Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1906–1915.

[39] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[40] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017, pp. 6626–6637.

[41] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, "Are gans created equal? a large-scale study," in *Advances in neural information processing systems*, 2018, pp. 698–707.

[42] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[43] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[44] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3d data processing," *arXiv preprint arXiv:1801.09847*, 2018.