Static Mesh Enrichment with Dynamic Entities for Training Sets Generation

Vivian Chiciudean, Radu Beche, Florin Oniga and Sergiu Nedevschi
Computer Science Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
FirstName.LastName@cs.utcluj.ro

***Abstract*—The 3D reconstruction of a real-world scene, usually represented as a textured mesh, supports the fusion of various information and can solve complex problems. For example, the fusion of the 3D textured mesh with a relatively small set of semantically annotated input images can generate a supplementary semantic mesh. The two meshes can be used, based on a set of consecutive camera positions, to generate novel RGB, depth, or semantic images. However, these meshes are not able to represent the dynamic objects, as these objects tend to vanish in the 3D textured mesh construction process. The main goal of this work is to provide a solution to generate training sequences including dynamic entities. Therefore, for a given camera pose and time instance, a new RGB, depth or semantically annotated image can be generated using a mesh instance that inserts the dynamic entities corresponding to the given timestamp. The proposed solution utilizes Blender, an open-source software tool, to place and animate the photorealistic mesh models of the different dynamic entities in the 3D mesh representation of the real-world scene. We use this tool to manage environment occlusions and shadows, and to mimic the properties of the scene. After the photorealistic rendering, we obtain a set of images of the dynamic objects and their shadows. We propose a technique to transfer the generated dynamic information into RGB, depth and semantic images. The method focuses on entities such as moving cars and pedestrians, but any dynamic entity is suitable. We used a subset of the UAVid dataset to test the practical viability of the solution for supervised semantic segmentation training. Experimental results show that using the enriched images, as opposed to the initial images, increases the performance on the semantic segmentation task by 10.77% mIoU.**

***Index Terms*—image enrichment, static scene, dynamic scene, semantic segmentation, 3D reconstruction, dynamic mesh, image inpainting**

I. Introduction

The limited availability of comprehensive labeled datasets poses a significant challenge in training and evaluating deep learning models for semantic segmentation, hindering the ability to capture the full range of scenarios. Annotating scenes involving small, fast-moving objects, as observed in UAV applications, adds complexity to the creation of such datasets. To overcome these limitations, we propose a methodology that builds upon prior research and focuses on enhancing static scenes by seamlessly integrating photorealistic dynamic objects into the acquired images. We address the underrepresented classes like vehicles and pedestrians by utilizing dense 3D reconstruction as a guiding framework to place and animate 3D dynamic entities within a scene.

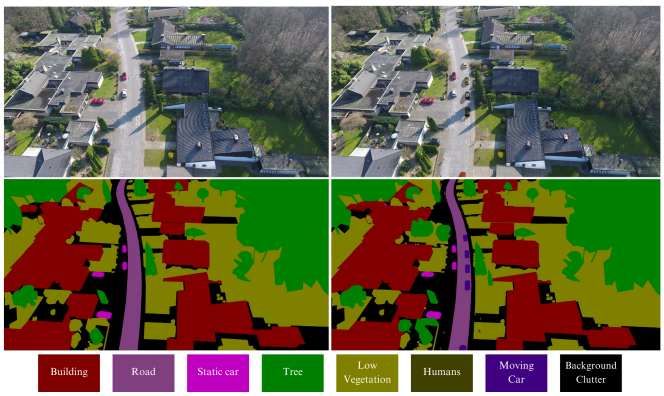


Fig. 1: Initial image (left) compared to enriched and corrected image (right).

In dynamic scenarios, traditional techniques that rely on static 3D mesh models for label propagation encounter limitations such as the presence of noise artifacts and potential disappearance of entities. These issues does not allow to consistently generate image sequences that are consistent in terms of realism, semantics and geometry. For example, Fu et al. [1] successfully employed coarse 3D panoptic annotations and environment reconstruction to produce panoptic masks, however, the camera movements being very limited. Recently, Chiciudean et al. [2] introduced a semantic mesh-based method that allows a virtual camera to capture and generate images from novel views, however, the dynamic objects were omitted altogether. In this context, dealing with small dynamic objects in 3D reconstruction can present significant challenges. Instead of relying solely on real data trajectories, we propose a novel approach that involves removing existing trajectories from the reconstruction and introducing synthetic ones. By incorporating synthetic trajectories, we aim to address the limitations and overcome the issues associated with accurately capturing and reconstructing small dynamic objects in the scene.

Expanding on prior work, we propose a new methodology for augmenting static scenes by inpainting photorealistic dynamic objects into preexisting imagery, maintaining the capabilities of novel view synthesis and label propagation. We leverage the capabilities of dense scene reconstruction

as a guide for placing and animating 3D dynamic entities, including examples such as vehicles and pedestrians. We conduct an iterative refinement of the global illumination, texture, and geometry of the entire scene, focusing our efforts on the objects of interest. By making use of the intrinsic and extrinsic parameters obtained from the dense photogrammetry reconstruction, we are able to render and transfer these objects of interest onto real-world images. To facilitate the smooth integration of these dynamic entities, we employ a modified version of the Alpha Blending technique. This allows us to transfer both the objects and their respective shadows to RGB frames, update the semantic masks and maintain a high level of fidelity in the representation of the augmented scene as seen in Fig. 1. Having access to all the aforementioned information, we are able to generate the corresponding depth images. We conducted experiments on low level aerial imagery collected from UAV as a case study for showing the benefits of this method. However, they can easily be extrapolated to other domain of applications such as automotive or surveillance.

To prove the effectiveness of our enrichment, we conducted experiments using the UAVid [3] dataset. Initially, we employed classical photogrammetry approaches [4] to reconstruct the dataset. Then, we applied our proposed pipeline to generate multiple potential traffic scenarios for each video sequence. To assess the performance of our method, we employed a semantic segmentation model and compared its training and evaluation outcomes with and without the synthetic data. Through extensive experimentation in real-world scenarios, we observed notable improvements. The main contributions of this paper can be summarized as follows:

- we implement a scalable method for enriching video-based datasets, which involves animating synthetic objects based on map constraints and integrate them into a geo-located preexisting 3D scene
- for the rendered images we ensure a high level of photo-realism while considering factors such as environmental occlusions, shadows and global illumination
- we developed a method for transferring the generated information that does not affect unwanted areas of the image
- we conduct a comprehensive study employing quantitative and qualitative methodologies to assess the proposed pipeline in the context of semantic segmentation task

II. Related Work

A. Aerial Datasets

The recent surge in UAV application development has created a demand for acquiring and annotating aerial datasets, focusing on areas such as semantic segmentation [5], object detection [6], [7], tracking [8], and more [9], [10]. For this study, we will specifically focus on datasets collected at relatively low altitudes with available semantic annotations considering a large number of classes. The selected dataset for our work is the UAVid [3] dataset, which comprises 42 sparsely annotated aerial video sequences obtained from both static and dynamic real-world environments targeting eight semantic classes. It is worth noting that our approach can be easily extended to other datasets [11]–[13], provided they offer video sequences with corresponding segmentation masks.

B. 3D Scene Reconstruction

3D reconstruction through classical photogrammetry approaches involves analyzing multiple images captured from different viewpoints and extracting spatial information to reconstruct the geometry and appearance of the scene. Notable examples in this field include COLMAP [4] [14] and OpenDroneMap [15]. While these are general-purpose pipelines that require significant larger running time, tailored SLAM applications like ORBSlam3 [16] can also be used to recover desired information. The reconstruction of dynamic scenes using these methods can be noisy, as moving objects tend to disappear from the static reconstruction and need to be modeled separately. The output representation of these methods usually consists of 3D meshes textured with associated RGB colors obtained from the dense reconstruction process. Additional texture information, such as semantic segmentation, can also be added. Utilizing these methods has the benefit of providing camera intrinsic and extrinsic parameters, enabling us to generate synthetic images from the exactly the same pose and add the objects over the real ones, guiding the animation making in the process.

C. Image Composing and Inpainting

The inclusion of new elements in an image, known as inpainting, has become a prominent subject in the field of artificial intelligence. Recent advancements in generative AI , particularly diffusion-based approaches [17] and GANs [18] (Generative Adversarial Networks), have facilitated the realistic addition of objects into scenes. However, these methods have a limitation: they cannot generate a sequence of images that adhere to geometric constraints. In order to overcome this constraint, we employ a three-dimensional framework as a foundational concept and depend on conventional rendering engines such as Blender [19] throughout our workflow. Following the rendering and illumination modeling of the scenes, we will utilize a modified alpha blending technique to merge the artificially generated entities with the RGB images.

D. Semantic Segmentation

Research in the domain of semantic segmentation of images captured from drones falls in two main categories: multi-scale networks and transformers. Multi-scale networks are designed to handle large scale variations present in drone imagery, where the sizes of objects can vary dramatically due to oblique view and different camera distances [3], [20], [21]. On the other hand, transformers are used to model the complex relationships between objects and capture the global context of the scene by leveraging self-attention mechanisms [22]–[24].

III. Overview of the Proposed Approach

The input of the proposed pipeline is a 3D mesh reconstruction of the real-world scene, the camera parameters of each image and a set of models that define the dynamic entities. Therefore, the main steps for obtaining an enriched set of images with dynamic entities from a set of images acquired from static scenes are as follows:

1) Import the inputs
2) Define places
3) Animate actions
4) Set the global illumination
5) Render the images
6) Transfer the dynamic entities

IV. Detailed Methodology

For a significant visual support and to be able to demonstrate the usefulness of the proposed method, we use the UAVid dataset. More precisely, we focused on a scene with little to no dynamic objects.

A. Import the Inputs

We started from the set of video sequences and used them to build a 3D mesh reconstruction. This was done using the COLMAP open-source software tool, but any photogrammetry based reconstruction tool is suitable. COLMAP can obtain both the geometry of the scene in form of a mesh, as well as the camera parameters of each video sequence frame used in the reconstruction. Fig. 2 shows the obtained 3D mesh reconstruction and the camera fly-path of the video sequences, where each black pyramid represent a camera pose. The camera orientation is given by the base of the pyramid, its top being the camera translation. For a better visualization, only the frames divisible by 250 are shown.

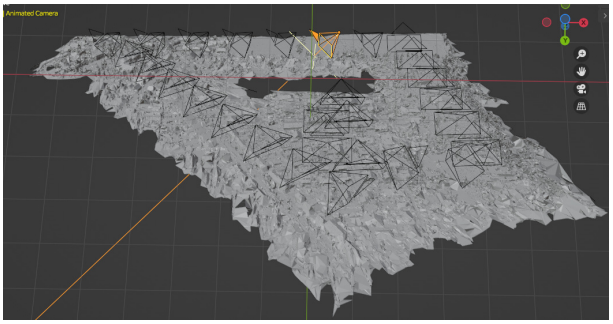


Fig. 2: The 3D mesh reconstruction and the camera poses using UAVid (seq29, seq13, seq15, seq36, seq19, seq31 and seq38). Only the frames divisible by 250 are displayed.

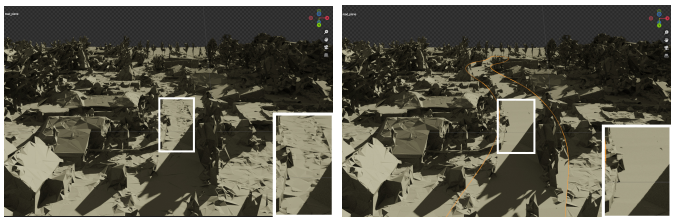
The mesh models for the dynamic entities can be manually modeled using a 3D model tool, like Blender, or can be found for free, online. The goal of this study is to add entities such as moving cars and humans. Therefore, Fig. 3 shows some of the photorealistic models used as dynamic entities.

Once we have these inputs, we can import them into Blender using a script, taking into account the change of the coordinate system, or using a specific add-on. For the ease of development, we made use of [25] for loading the COLMAP information, while the mesh models were loaded using the standard Blender method. Nevertheless, we have to take into consideration the scale of the models, for both the reconstruction mesh and the dynamic entities. The safest method is to recover the scale in the reconstruction process, and then, based on a reference from the mesh, to proportionally scale the entities.

Fig. 3: Dynamic entities mesh models.

B. Define Places

At this point, we can define the places where the entities should move or appear. Because we wanted to enrich the images with moving cars and pedestrians, the suitable places are the road or the pedestrian walking area. Due to the fact that the video sequences cover a single camera pass over the scene, and not a view from several angles of it, the reconstruction can have significant noises. These type of noises are presented in Fig. 4a. For a better visualization, we added a sun-type light source.



(a) Noises (b) Smoothed

Fig. 4: Road reconstruction - UAVid seq13.

In order to obtain a seamlessly transfer of the generated information, we need a global illumination that reflects the real-world environment and also capture the shadows of the dynamic entities. Therefore, the surface on which we capture the shadows must be as smooth as possible. To alleviate this problem, we added a new plane that follows the road path and remove almost all the reconstruction noises. This is presented in Fig. 4b. The reconstruction noises presented above can negatively influence the projection of the shadows and generate visual artifacts as presented in Fig. 5a and corrected in Fig. 5b.

(a) Shadows on reconstruction (b) Shadows on smooth surface

Fig. 5: Road shadows

After the surface of interest is smoothed, we can define various paths for the model entities to follow. For example, in Fig. 6 we define a main road path and two sidewalk paths that will be attached to the road plane. One step to automate this process is to transfer information from OpenStreetMap [26] to Blender using a specialized add-on. This will automatically add Bezier curves that define the main roads of the scene. However, fine-tuning the control points may be required depending on the 3D reconstruction.

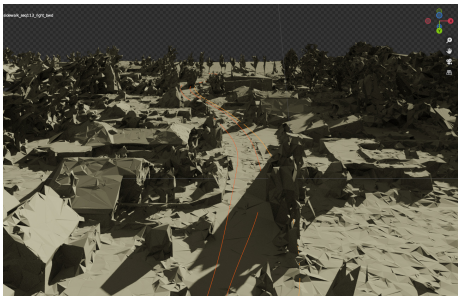


Fig. 6: Defined animation paths.

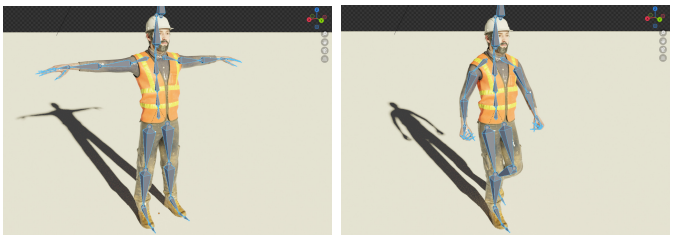
C. Animate Actions

Once we have defined the paths where various entities will be placed and made sure that the place where their shadow will be projected is smooth so that visual artifacts won't be produced, we can define a specific movement for each model.

Regarding the dynamic cars, we set a constraint in Blender for each mesh model to follow the defined road path, without animating the wheel movement. The main reason that motivated this choice was the nature of the dataset used, since wheel movement is almost imperceptible in aerial images. Therefore, the car animation is defined by a simple translation of the model.

As far as moving pedestrians are concerned, this translation would seem much less natural, and would not reflect a normal movement, regardless of the distance from which the images were acquired. We focused on walking pedestrians, without various complex actions. Therefore, it is appropriate to define a fixed walking-cycle animation in which the pedestrian is translated at a constant speed, on the sidewalk defined in the previous step, and performs the defined animation in a loop. For this we use rigged models. These models have an associated armature, meaning a certain skeleton, that will be linked to various parts of the model. Fig. 7a shows the initial pose of the armature for a rigged model.

A walking animation may be created manually or automatically. The first option involves the placing of each bone of the armature at different positions in successive key frames, then interpolating the movement and deforming the mesh accordingly. While the second options involves automated online tools[1] that take a common rigged model and provide a unique animation. Fig. 7b shows a specific pose of the walking cycle.



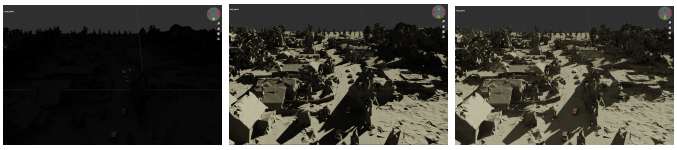
(a) Initial pose (b) Walk pose

Fig. 7: Rigged model.

After the animations are defined for each dynamic entity, we can constraint each model to follow one of the three defined paths and set a specific speed. We carefully place each car-type entity on the road path, and each pedestrian-type entity on one of the sidewalk-type paths.

D. Set the Light Properties

At this point we will have a fully functional dynamic mesh, with moving entities and specific actions. However, in the previous figures we did not mentioned the global illumination settings that will help us to reflect the real-world acquisition environment. Fig. 8a presents the rendering result if we do not use any light source in the previous scene.



(a) No sun (b) Sun (c) Sun and area light

Fig. 8: Global illumination of the dynamic mesh.

To solve this, we add a sun-type light source, and the results change significantly. This is shown in Fig. 8b. In this step it is very important to analyze the type and color of the light in the acquisition environment and chose a light source with similar properties. It is also important to approximate from which direction the light comes, in order to obtain shadows similar to those in the initial images. This will help to correctly project the shadows of the dynamic entities, as well as the shadows of the reconstructed environment over the dynamic entities.

[1]https://www.mixamo.com

However, in the rendering result, the shadowed dynamic entities have a very big difference in illumination compared to the models on which environmental shadows are not projected. To cope with this problem, we add an area light source that does not cast shadows. The results are shown in Fig. 8c.

E. Render the Images

After defining multiple paths and populating the mesh with more dynamic entities, Fig. 9 shows the initial generated RGB image using the UAVid camera parameters of seq13, frame 800.

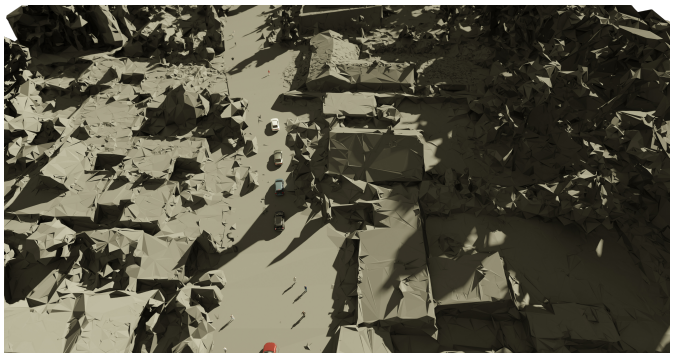


Fig. 9: Rendered RGB image, without shadow catcher.

In order to be able to further transfer only the dynamic entities and their shadows, we do not want to render the 3D mesh reconstruction but only its occlusions and shadows that influence the entities. This can be done using a specific Blender setting, namely marking the 3D reconstruction a shadow catcher. The same settings are used for the smoothing plan placed above the road. Therefore, the initially generated image, using the same camera parameters, is presented in Fig. 10. Note that this image is an RGBA type image.

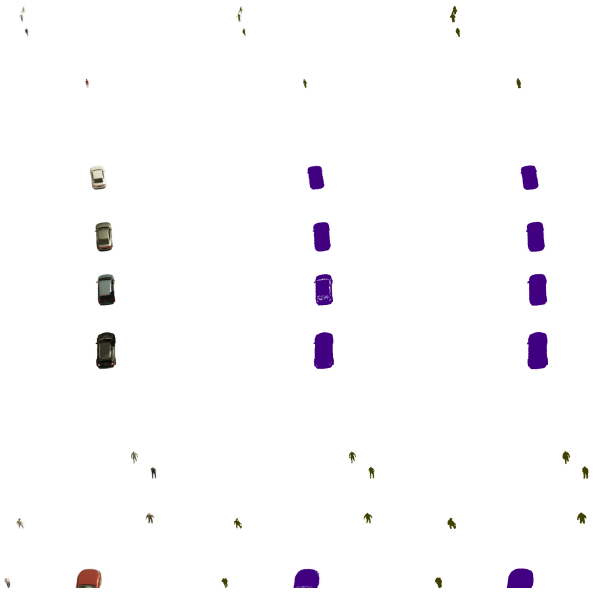
Fig. 10: Rendered RGBA image, final results.

Depending on the initial images, specific Blender parameters should be changed. For example, in foggy conditions there should be a mist pass added to the generated image, to better reflect the real-world global illumination and environment. In the case of unfocused images, the focal length of the camera should be changed.

So far we focused on generating the RGB information. Regarding the semantic information, we have to render the image on two layers, a foreground layer and a background one. The dynamic entities will be a part of the foreground layer, while the 3D reconstruction and road plane will be part of the background layer. In this way we will separate the dynamic object from their shadows and obtain the mask of each object from the foreground. Fig. 11a shows the generated RGBA image obtained after the rendering of the foreground layer. An alpha greater than zero means that there is a dynamic entity, while the R, G and B channels have been set to the color of the entity.

To differentiate between masks from different types of entities, we took advantage of another Blender specific setting, the Object Index pass. Therefore, we set an index for each dynamic entity placed in the scene. Because we have only two types of dynamic entities, we choose index one for cars and index two for pedestrians. Fig. 11b shows the obtained indices image. For visualization purposes, the car index was marked as purple and the pedestrian index was marked as brown, as were the moving car and human semantic classes of UAVid.

However, there are some problems with this approach, specifically, the edges and translucent parts of the dynamic entities. In this case, these problems manifest themselves especially at the level of pedestrian contours and at the level of car windows and windshields. Therefore we proposed a combination of the mask generated at the previous step (Fig. 11a) and the Object Index Pass image (Fig. 11b). We applied a morphological dilation on the index image, then extracted the relevant information based on the mask image. Thus, we obtained the final semantic mask (Fig. 11c). The subfigures in Fig. 11 present, for a better visualization, only the region of interest with dynamic entities from the generated images.



(a) No shadows (b) Object Index Pass (c) Semantic mask

Fig. 11: Obtaining semantic masks, only the region of interest is displayed.

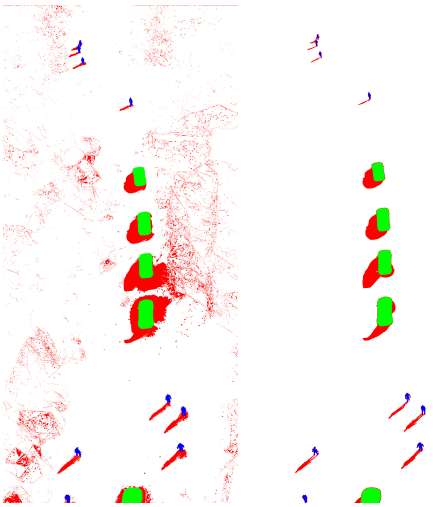
F. Transfer the Dynamic Entities

At this point, we have all the desired dynamic information in certain images, but we want to seamlessly transfer this information into the initial images.

$$Image = \alpha * Foreground + (1 - \alpha) * Background \quad (1)$$

For the initial RGB images, we will use the generated RGBA image (Fig. 10) and apply the Alpha Blending technique (1). However, using this method, due to ambient occlusions and shadows specific to the 3D reconstruction, a series of unwanted noises may appear. Fig. 12a highlights these kind of noise, using red for the shadows presents in the RGBA image (Fig. 10), blue for pedestrians and green for cars. Although we want to transfer the shadows of the dynamic entities, we do not want to alter the original images in wrong places in ways that are imperceptible to the human eye, but possibly perceptible by a neural network. Another problem is that an Alpha Blending operation can make distant entities fade out or even vanish. Therefore, we propose a transfer process based on the opacity level in the RGBA image (Fig. 10) and the semantic class in the mask image (Fig. 11c).

The proposed method involves the application of the Alpha Blending technique for each pixel with an opacity greater than a threshold (T1), and then overwriting, without blending, all car or pedestrian pixels that have a lower value than another threshold specific to their class (T2 - car and T3 - pedestrian). For this case, we experimentally found that T1 = 30, T2 = 200 and T3 = 96 work best (Fig. 12b). In this way, we will transfer the shadows and the outline of the dynamic entities using the blending technique, while their interior will be overwritten pixel by pixel.



(a) Problem (b) Solution

Fig. 12: Transfer the dynamic entities in RGB images, only the areas with alpha greater than 0 are displayed.

For the semantic information, since there are no shadows in the generated images, we can simply overwrite the pixels of interest with the corresponding classes. The final results are shown in the second column of Fig. 14.

As a final remark, choosing the position of the light and the textural properties of the materials can drastically influence the way mesh models, shadows and ambient occlusions are rendered (Fig. 13). At this point, the 3D reconstruction of the scene also plays a very important role. If we use a poor quality reconstruction, the results will be wrong in terms of occlusions and shadows.

(a) Without fine-tuning (b) With fine-tuning

Fig. 13: Illumination and texture properties differences - UAVid seq31, frame 316.

V. Experiments

A. Dataset

UAVid [3] is a dataset acquired from two countries and consists of 42 aerial video sequences at a 4K resolution. Each video has 900 frames, and for each frame whose number is a multiple of 100 a manual semantic annotation was made. A total of eight classes are defined for the semantic segmentation task, namely: building, road, tree, low vegetation, static car, moving car, human, and background clutter. With a total of 420 images, this dataset is divided into 200 images for training, 70 for validation and 150 for testing. The semantic images used for testing are not publicly available, therefore there are only 270 semantic images.

After a detailed analysis of the areas from which these images were acquired, we noticed that the images from a closed-loop subset have very few dynamic objects. Therefore, we chose seven video sequences (seq29, seq13, seq15, seq36, seq19, seq31 and seq38) to reconstruct the 3D mesh of a closed-loop area of 600 x 800 meters. The mesh reconstruction was obtained through the COLMAP pipeline. Regarding the semantic information, these seven video sequences has only 50 semantic images publicly available, because seq29 and seq38 are being used for testing by the UAVid official benchmark. After another detailed analysis of the 50 annotated images, we noticed a series of errors and inconsistencies (i.e., different labels between frames) in the UAVid manual labeling. These issues were comprehensively described by Chiciudean et al. [2]. In this work, we manually corrected all these problems

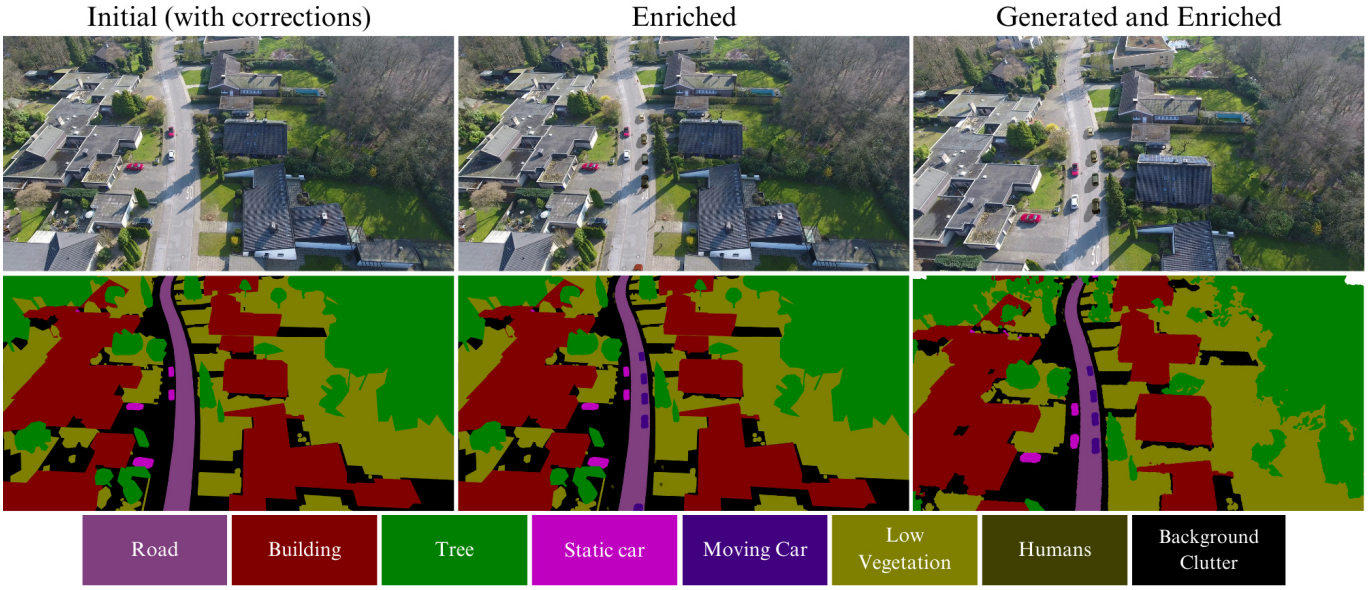


Fig. 14: Results of the proposed method. The first column shows the initial RGB image of seq13, frame 800, and the corresponding corrected semantic image. The second column shows the enriched RGB and semantic image of the same frame. The third column shows the generated and enriched images for frame 850, seq13.

Fig. 15: Before (left) and after (right) corrections. Overlaid semantic image on RGB frame - UAVid seq36, frame 0.

for the 50 images of interest. Fig. 15 displays the corrections (right) of the initial semantic image (left) of frame 0, seq36.

Moreover, based on the augmenting method developed in [2], we generated 902 new semantic images using the 50 initial manually annotated images. For a visual understanding, the images generated by the augmentation method [2] and enriched with dynamic entities by the proposed solution are presented in the last column of Fig. 14. These images are generated with 50 frames ahead of the existing ones.

B. Semantic Segmentation Results

In this section, we present the methodology employed to assess the usefulness of an enriched dataset for the task of semantic segmentation.

TABLE I: Semantic Class Distribution (%)

Dataset	Background Clutter	Moving Car	Static Car	Human	Building	Ground Vegetation	Tree	Road
E2	12.352	0.018	0.297	0.007	21.673	37.652	23.286	4.592
E3	12.341	0.233	0.297	0.017	21.673	37.650	23.256	4.410

We use UNetFormer [23] which is composed of a ResNet18 encoder, and a transformer based decoder. The network is implemented with PyTorch, using AdamW optimizer, with base learning rate of 6e-4, employing cosine annealing learning rate scheduler. For loss, we use a combination of cross-entropy and Dice loss. Each image is cropped into 1024x1024 patches, with random brightness, vertical and horizontal flips. We train for 40 epochs, with batch size of 8, on a NVIDIA 3090 GPU. The following data splits are employed, which can also be seen in Table III:

- Experiment 0 (E0) - From the 50 official UAVid images (seq13, seq15, seq19, seq31, and seq36), we use 45 for training and five for testing. We also add half of seq14 for training and the other half for testing. Resulting in a total of 50 images for training and 10 images for testing.
- Experiment 1 (E1) - Almost the same setup as E0, but we used the corrected semantic images instead.
- Experiment 2 (E2) - We add to E1 the 902 images generated for training and keep the same 10 for testing.
- Experiment 3 (E3) - The quantity and data distribution is the same as for E2, with the inclusion of synthetic cars and pedestrians in the images of seq13, seq15 and seq31.

The test images were chosen so that there were no real-world areas that also appeared in the training images.

The mean pixel-level distribution of the semantic classes is presented in Table I. Compared to the corrected set of images (E2), the proposed method (E3) increased the number of pixels for the ”Moving Car” semantic class by a factor of 12.9, while the number of pixels for the ”Human” class increased 2.4 times.

We report the IoU results in Table II, from which we notice a mIoU improvement of +0.63% when we used the corrected images (E1), compared to the same dataset used without any corrections (E0). An additional +4.11% improvement when we increase the quantity of training data from 50 images

TABLE II: Quantitative Semantic Segmentation Results using UNetFormer.

Experiment	Clutter	Moving Car	Static Car	Human	Building	Vegetation	Tree	Road	mIoU
E0	55.55	0	41.32	0	84.00	63.03	67.64	68.28	47.48
E1	56.88	0	41.34	0	84.59	65.37	66.87	69.81	48.11
E2	62.02	11.11	49.07	0	85.90	68.80	65.67	75.21	52.22
E3	61.68	**64.15**	50.99	**35.81**	86.26	67.73	65.16	76.69	**62.99**
E4	59.95	57.59	50.13	30.06	86.42	67.00	65.32	74.28	61.34

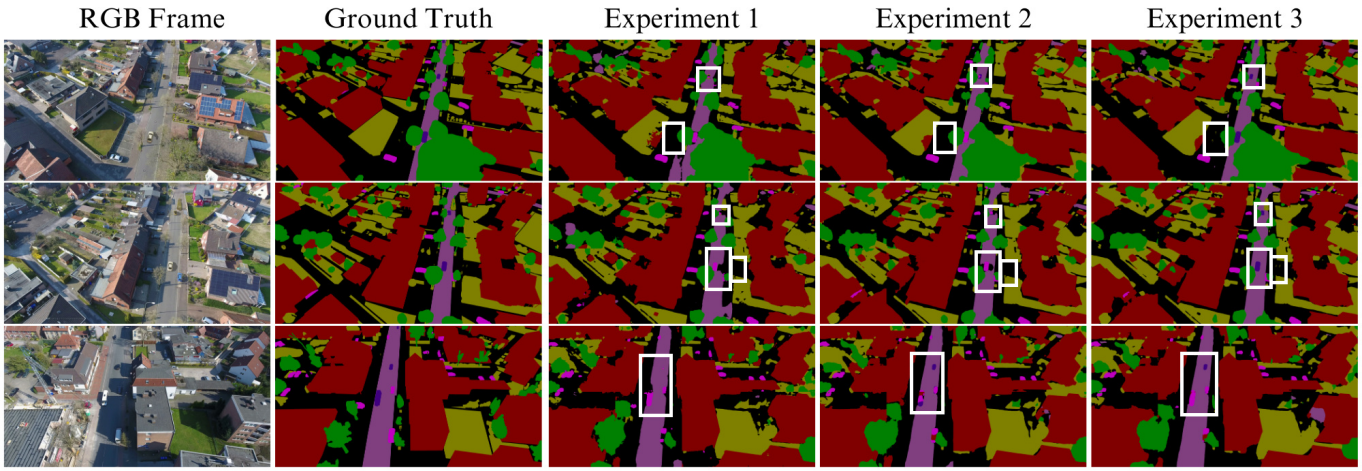


Fig. 16: Qualitative semantic segmentation results using UNetFormer. The columns from left to right contain: the RGB frame, the semantic segmentation ground truth, and the three outputs for each training dataset. Experiment 1 contained 50 training images, while experiments 2 and 3 used 952 images for training. Experiment 3 also contained synthetic pedestrians and moving cars. The white rectangles highlight various elements of interest, such as cars and humans.

TABLE III: Data Split for Semantic Segmentation Experiments.

Seq	#img	Purpose	Generated			
13	10	Train	no	**E1**	**E2**	**E3**
15	10	Train				
19	10	Train				
36	10	Train				
31	5	Train				
	5	Test				
14	5	Train				
	5	Test				
13	261	Train	yes			
15	261	Train				
19	90	Train				
36	174	Train				
31	116	Train				

to 952 images, and another improvement of +10.77% when synthetic 3D models for pedestrians and cars are introduced in the training data. In Figure 16 we present results for three test cases. We observe that the network has difficulties segmenting small-sized objects such as cars and humans. In the case of cars, when multiple examples are provided, the system begins to distinguish between static and dynamic cars. However, since certain car models have not been introduced synthetically, such as buses or minivans, the network has problems in extracting them. For the human semantic class their distribution is still smaller compared to the other classes. Therefore, the network learns to correctly extract the pedestrians only in the last experiment, using the synthetic data enrichment.

C. Ablation Study

To prove the importance of the proposed transfer method for dynamic entities and their shadows in RGB images, we performed an ablation study. For this, we compared the aforementioned results from E3 with a new experiment, E4. In this new experiment, we employed the same dataset used for E3, but we chose to transfer all the information from the generated RGBA images (Fig. 10) without taking into account the alpha channel thresholds (i.e., using the original Alpha Blending technique). The last two rows of Table II shows that if we had not applied the special transfer of synthetic pedestrians and moving cars, the overall performance of the network would have been 1.65% mIoU lower, and the least represented classes (i.e., Moving Car and Human) would have had a major segmentation decrease.

VI. Conclusions

In this paper, we proposed a processing pipeline for animating a 3D static mesh with dynamic entities in order to enrich a set of images and obtain complex training sets. The pipeline aims to create a virtual scene that reflects the real-world

environment from which the images were acquired, taking into account the global illuminations, occlusions and shadows. At the end of it, we transferred the obtained information to RGB and semantic images.

In order to prove the practical utility of the proposed approach and evaluate its capabilities, the UAVid dataset [3] was used. This work focused on a subset of 50 images from this dataset, which was improved by various manual corrections of errors and inconsistencies. The results for the semantic segmentation task show an improvement of 10.77% mIoU, significantly increasing the detection of poorly represented dynamic object classes.

In terms of future work, we plan to develop a rendering method based on semantic labels, which is not limited by the 3D reconstruction of the scene and allows the rendering of occluded objects even for wrong reconstructions. We also plan to generate depth masks for the dynamic entities and analyze how much this can help the task of depth estimation.

VII. Acknowledgment

This work was supported by the "DeepPerception - Deep Learning Based 3D Perception for Autonomous Driving" grant funded by Romanian Ministry of Education and Research, code PN-III-P4-PCE-2021-1134.

References

[1] X. Fu, S. Zhang, T. Chen, Y. Lu, L. Zhu, X. Zhou, A. Geiger, and Y. Liao, "Panoptic nerf: 3d-to-2d label transfer for panoptic urban scene segmentation," *arXiv preprint arXiv:2203.15224*, 2022.

[2] V. Chiciudean, H. Florea, B.-C.-Z. Blaga, F. Oniga, and S. Nedevschi, "Data augmentation for environment perception with unmanned aerial vehicles," June 2023, manuscript submitted for publication.

[3] Y. Lyu, G. Vosselman, G.-S. Xia, A. Yilmaz, and M. Y. Yang, "UAVid: A semantic segmentation dataset for UAV imagery," *ISPRS journal of photogrammetry and remote sensing*, vol. 165, pp. 108–119, 2020.

[4] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys, "Pixelwise view selection for unstructured multi-view stereo," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*. Springer, 2016, pp. 501–518.

[5] Y. Pi, N. D. Nath, and A. H. Behzadan, "Detection and semantic segmentation of disaster damage in uav footage," *Journal of Computing in Civil Engineering*, vol. 35, no. 2, p. 04020063, 2021.

[6] G. Cheng and J. Han, "A survey on object detection in optical remote sensing images," *ISPRS journal of photogrammetry and remote sensing*, vol. 117, pp. 11–28, 2016.

[7] Z. Li, A. Namiki, S. Suzuki, Q. Wang, T. Zhang, and W. Wang, "Application of low-altitude uav remote sensing image object detection based on improved yolov5," *Applied Sciences*, vol. 12, no. 16, p. 8314, 2022.

[8] L.-Y. Lo, C. H. Yiu, Y. Tang, A.-S. Yang, B. Li, and C.-Y. Wen, "Dynamic object tracking on autonomous uav system for surveillance applications," *Sensors*, vol. 21, no. 23, p. 7888, 2021.

[9] B. Chen, Z. Chen, L. Deng, Y. Duan, and J. Zhou, "Building change detection with rgb-d map generated from uav images," *Neurocomputing*, vol. 208, pp. 350–364, 2016.

[10] H. Ling, H. Luo, H. Chen, L. Bai, T. Zhu, and Y. Wang, "Modelling and simulation of distributed uav swarm cooperative planning and perception," *International Journal of Aerospace Engineering*, vol. 2021, pp. 1–11, 2021.

[11] H. Florea, V.-C. Miclea, and S. Nedevschi, "Wilduav: Monocular uav dataset for depth estimation tasks," *2021 IEEE 17th International Conference on Intelligent Computer Communication and Processing (ICCP)*, 2021.

[12] I. Bozcan and E. Kayacan, "Au-air: A multi-modal unmanned aerial vehicle dataset for low altitude traffic surveillance," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8504–8510.

[13] ——, "Au-air: A multi-modal unmanned aerial vehicle dataset for low altitude traffic surveillance," 2020.

[14] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.

[15] OpenDroneMap, "A command line toolkit to generate maps, point clouds, 3d models and dems from drone, balloon or kite images." https://github.com/OpenDroneMap/ODM, 2020.

[16] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, dec 2021. [Online]. Available: https://doi.org/10.1109%2Ftro.2021.3075644

[17] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," 2021.

[18] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, "Alias-free generative adversarial networks," in *Proc. NeurIPS*, 2021.

[19] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

[20] Y. Su, J. Cheng, H. Bai, H. Liu, and C. He, "Semantic segmentation of very-high-resolution remote sensing images via deep multi-feature learning," *Remote Sensing*, vol. 14, no. 3, p. 533, 2022.

[21] M. Y. Yang, S. Kumaar, Y. Lyu, and F. Nex, "Real-time semantic segmentation with context aggregation network," *ISPRS journal of photogrammetry and remote sensing*, vol. 178, pp. 124–134, 2021.

[22] L. Wang, R. Li, D. Wang, C. Duan, T. Wang, and X. Meng, "Transformer meets convolution: A bilateral awareness network for semantic segmentation of very fine resolution urban scene images," *Remote Sensing*, vol. 13, no. 16, p. 3065, 2021.

[23] S. Yi, X. Liu, J. Li, and L. Chen, "UAVformer: A Composite Transformer Network for Urban Scene Segmentation of UAV Images," *Pattern Recognition*, vol. 133, p. 109019, 2023.

[24] L. Wang, R. Li, C. Zhang, S. Fang, C. Duan, X. Meng, and P. M. Atkinson, "UNetFormer: A UNet-like transformer for efficient semantic segmentation of remote sensing urban scene imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 190, pp. 196–214, 2022.

[25] S. Bullinger., C. Bodensteiner., and M. Arens., "A photogrammetry-based framework to facilitate image-based modeling and automatic camera tracking," in *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021) - GRAPP*, INSTICC. SciTePress, 2021, pp. 106–112.

[26] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.