Dynamic-to-Static Image Translation for Visual SLAM

Berta Bescos, Cesar Cadena, Jose Neira

***Abstract*—We present a method for improving vision-based localization and mapping tasks under appearance changes due to dynamic objects. We introduce an end-to-end deep learning framework to turn urban images that show dynamic content, such as vehicles or pedestrians, into realistic static frames which are suitable for localization and mapping. Such images would increase the autonomous vehicles understanding of their stationary surroundings. This objective encounters two main challenges: detecting the dynamic objects, and inpainting the static occluded background. The former challenge is addressed by the use of a convolutional network that learns a multi-class semantic segmentation of the image. The second challenge is approached with a generative adversarial model that, taking as input the original dynamic image and the computed dynamic/static mask, is capable of generating the final static image. This framework makes use of two new losses, one based on image steganalysis techniques, useful to improve the inpainting quality, and another one based on ORB features, designed to enhance feature matching between real and synthesized image regions.**

To validate our approach, we perform SLAM and place recognition experiments with the synthesized images, showing the benefits of our proposal for vision-based localization tasks.

I. Introduction

In this work we present a manner of improving the performance of vision-based localization and mapping tasks in environments with changes in appearance due to dynamic objects such as cars, bikes, pedestrians, *etc*, using images synthesized by a Convolutional Neural Network (CNN).

Most vision-based localization systems are conceived to work in static environments. Such systems compute the dynamic objects motion as camera ego-motion. Thus, their performance is compromised. Building stable maps is also of key importance for long-term autonomy. Mapping dynamic objects prevents vision-based robotic systems from recognizing already visited places and reusing pre-computed maps.

The standard approach to deal with dynamic objects consists in detecting them in the images so as not to use the information belonging to them for neither localization nor mapping. However, we propose to instead modify these images so that the dynamic content is converted realistically into static. We consider that the combination of experience and context allows us to hallucinate, *i.e.*, inpaint, a geometrically and semantically consistent appearance of the static structure behind dynamic objects. Turning images that contain dynamic objects into realistic static frames reveals several challenges:

1) Detecting such dynamic content in the image.
2) Inpainting the space left with areas that are semantically and geometrically consistent with the rest of the image.

The former challenge can be addressed with geometrical approaches if an image sequence is available –usually studying the optical flow consistency along the images [29, 1]–. If only one frame is available, deep learning is the approach that excels at this task by the use of CNNs [13, 22]. These frameworks are trained with the knowledge of what classes are dynamic, but recent works show that it is possible to acquire this knowledge in a self-supervised way [2, 32].

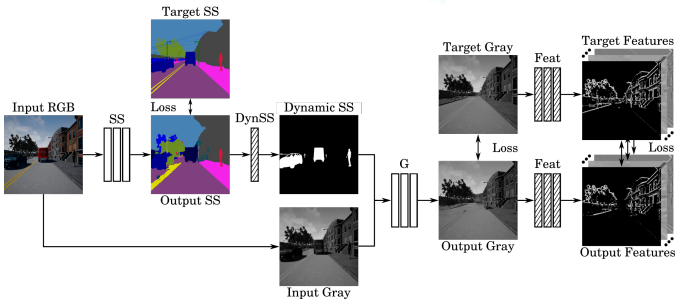


Fig. 1: Block diagram of our proposal. We first compute the segmentation of the dynamic objects of the input RGB image, and then obtain the static image with both the dynamic/static binary mask and the dynamic image. A loss based on ORB features together with an appearance and an adversarial loss are used during training. The striped blocks are differentiable layers with pre-defined weights that are not modified during training.

As for the second challenge, image inpainting approaches that do not use deep learning use image statistics of the remaining image to fill in the holes [27, 4]. Such approaches generally produce smooth results, but are limited by the available image statistics and have no concept of visual semantics. However, neural networks learn semantic priors and meaningful hidden representations in an end-to-end fashion, which have been used for recent image inpainting efforts [17, 20, 14].

Both challenges can also be seen as one single task: translating a dynamic image into a corresponding static image. In this direction, Isola et al. [15] propose a general-purpose solution for image-to-image translation. Our previous work Empty Cities [6] builds on top of this idea and re-formulates the framework objectives to take advantage of a pre-computed dynamic objects' mask, seeking a more inpainting-oriented framework. In this work, we follow this idea of transforming images with dynamic content into realistic static frames, and want to optimize them to be used for both localization and mapping tasks for autonomous vehicles. These images provide a richer understanding of the stationary scene, and could also be of interest for the creation of high-detail road maps, as well as for street-view imagery suppliers as a privacy measure to replace faces and license plates blurring. For such task, we introduce a new loss that, combined with the integration of a semantic segmentation network achieves the final objective of creating a dynamic-object-invariant space. This loss is based on steganalysis techniques and on ORB features detection, orientation and descriptor maps [24]. Such loss allows the inpainted images to be realistic and suitable for localization and mapping tasks.

II. Related Work

Dynamic Objects Detection The vast majority of SLAM systems assume a static environment. Therefore, they can only manage small fractions of dynamic content by classifying them as outliers to such static model, as in ORB-SLAM [18, 19] or PTAM [16]. There are several SLAM systems that more explicitly address the dynamic scene content. Tan et al. [26] detect scene changes by projecting the map features into the current frame for appearance validation. Wang and Huang [29] and Alcantarilla et al. [1] segment the dynamic objects in the scene using RGB optical flow. More recently, thanks to the boost of deep learning, integrating semantics information into SLAM has allowed to deal with dynamic content in a different manner [25, 5]. It allows clustering of map points belonging to independent objects with different dynamics, as well as the possibility of detecting dynamic objects in just one shot.

Sequence-based Inpainting Our previous work on SLAM in dynamic scenes attempted to reconstruct the background occluded by dynamic objects in the images with information from previous frames [5]. It needs per-pixel depth information and makes use of the static content of the pre-built map to create the inpainted frames, but does not add semantic consistency. Granados et al. [12] remove marked dynamic objects from videos by aligning other candidate frames in which parts of the missing region are visible, assuming that the scene can be approximated using piecewise planar geometry. The recent work by Uittenbogaard [28] uses a Generative Adversarial Network (GAN) to learn to use imagery information from different viewpoints to generate a plausible inpainting.

Image-based Inpainting Among the non-learning approaches to inpainting, propagating appearance information from neighboring pixels to the target region is the usual procedure [27]. These methods succeed in dealing with narrow holes, but fail when handling big holes, resulting in over-smoothing. Differently, patch-based methods [9] iteratively search for relevant patches from the rest of the image. Yet, they do not make semantically aware patch selections. Deep learning methods usually initialize the image holes with a constant value, and pass it through a CNN. Context Encoders [20] successfully used a standard pixel-wise reconstruction loss, as well as an adversarial loss for image inpainting tasks. Iizuka et al. [14] extend their work by defining global and local discriminators. Differently, Yu et al. [31] added a refinement network powered by contextual attention layers. The recent work of Liu et al. [17] make use of partial convolutions.

Our work bins the image sequences and treats the frames independently. Therefore, we make use of deep learning to segment out the *a priori* moving objects –vehicles and pedestrians–, and also of image-based "inpainting". Our system does not perform pure inpainting but image-to-image translation with the help of a dynamic objects' mask, outcome of a semantic segmentation network. This choice is justified by the fact that the dynamic objects' mask might be inaccurate or may not include their shadows, and the adoption of an image-to-image translation framework allows us to slightly modify the image non-hole regions for better accommodation of the reconstructed areas. Differently to inpainting methods, we do not initialize the "holes" with any placeholder values, and our inpainting network input consists of the dynamic original image with the dynamic/static mask concatenated. Concisely, utilizing an image-to-image translation approach allows us to have the image hole regions inpainted, and the non-hole regions slightly modified for better accommodation of the reconstructed areas to cope with imprecise masks, or with dynamic objects possible shadows and reflections.

III. Our Proposal

Our proposal is to turn images of an urban environment that show dynamic content, such as vehicles or pedestrians, into realistic static frames which are suitable for navigation. For this purpose, we first obtain the pixel-wise dynamic/static semantic segmentation of the dynamic image –see Fig. 1–. Once we have this segmentation, we convert the RGB dynamic image to gray scale and we compute the static image with the use of the generator G, which has been trained in an adversarial way –for simplicity, the discriminator is not shown–. To fully exploit the capabilities of this framework for navigation, inpainting is enriched with a loss based on ORB features detection, orientation and descriptors. We perform the inpainting in gray-scale rather than in RGB. This is motivated by the fact that many visual localization applications only need gray-scale information, and that learning a mapping from 1D→1D instead of from 3D→3D is simpler and therefore leads to having less room for wrong reconstructions.

A. From Image-to-Image Translation to Inpainting

For our objective, dynamic objects masks are specially considered to re-formulate the training objectives of the general purpose image-to-image translation work by Isola et al. [15]. We adopt a variant of the conditional GAN (cGAN) that learns a mapping from observed image x and dynamic/static mask m, to y, $G:\{x,m\}\rightarrow y$. Also, the discriminator D learns to classify $\hat{y}=G(x,m)$ patches as "fake" from $\hat{y}$, m and x, and the patches of y as "real" from y, m and x, $D:\{x,y/\hat{y},m\}\rightarrow real/fake$. In most of our training dataset images, the relationship between the static and dynamic regions sizes is unbalanced, *i.e.*, static regions occupy usually a much bigger area. This leads us to believe that the influence of dynamic regions on the final loss is significantly reduced. As a solution to this problem, we propose to re-formulate the $cGAN$ and $L1$ losses so that there is more emphasis on the main areas that have to be inpainted, according to Eqs. 1 and 2. The weights w are computed as $w=\frac{N}{N_{dyn}}$ if $m=1$ (dynamic object), and as $w=\frac{N}{N-N_{dyn}}$ if $m=0$ (background). N stands for the number of elements in the binary mask m, and N_{dyn} means the number of pixels where $m=1$.

$$\mathcal{L}_{L1}(G)=\mathbb{E}_{x,y}[w\cdot||y-G(x,m)||_1], \tag{1}$$

$$\mathcal{L}_{cGAN}(G,D)=\mathbb{E}_{x,y}[w\cdot\log D(x,y,m)]+ \mathbb{E}_{x}[w\cdot\log(1-D(x,G(x,m),m))], \tag{2}$$

An important feature of our framework is the computation of our images "noise", motivated by its use for steganalysis to detect if an image has been tampered. Fig. 2 shows an example of why working in the noise domain is helpful for detecting "fake" images. While the generated image (Fig. 2b) looks visually similar to its target image (Fig. 2d), their computed noises (Figs. 2c and 2e) are very different. The discriminator can more easily learn to distinguish "real" from

(a) (b) (c) (d) (e)

Fig. 2: (b) shows the image generated by our framework when taking (a) as input. (d) is the static objective image. (b) and (d) are visually similar, but their computed noise ((c) and (e) respectively) clearly show what image and what parts of it have been modified the most.

"fake" images by taking as input their noise. To the best of our knowledge, steganalysis features have not been used before in the context of GANs. This idea is further explained in subsection III-B.

This GAN training setup leads to having good inpainting results. However, despite the efforts of the discriminator to catch the high frequency of the "real" images, the outputs of our framework are still a little bit blurry. One of the objectives of this work is to use our images for localization tasks, therefore, if the inpainted regions are somewhat blurry, features would not be extracted in these areas. We have proved with our localization experiments (subsection IV-C) that not using dynamic objects' features leads to worse tracking results than working with fully static images. Therefore, we want to exploit our framework to obtaining both high-quality inpainting results, and to succeeding in generating reliable features for visual localization tasks. Luckily, these two assignments are highly related. Solving one of them leads to having the other one tackled. Therefore, we have implemented a new loss based on ORB features [24]. That is, we want the output of our generator G to have the same ORB features than its target image, while keeping it realistic and close to its target in a $L1$ sense. This procedure is further described in subsection III-C.

B. Steganalysis-Based Loss

With the advances of image editing techniques, manipulated image generation processes have become widely available. As a result, distinguishing authentic images from tampered images has become increasingly challenging. Our framework is actually trying to eliminate certain regions from an authentic image followed by inpainting, *i.e.*, *removal*, one of the most common image manipulation techniques. It is the discriminator's job to classify the generated images as tampered –fake– or real.

Recent work on image forensics utilizes clues such as local noise features [10, 33] to classify a specific patch or pixel in an image as tampered or not, and localize the tampered regions. The intuition behind this idea is that when an object is removed from one image (source) and the gap is inpainted (target), the noise features between the source and target are unlikely to match. To utilize these features, we transform our images –target y and output $\hat{y}$– into the noise domain, and use the local noise features n and $\hat{n}$ as the input to our network's discriminator –$D(x,y,m,n)$ and $D(x,\hat{y},m,\hat{n})$–. There are many ways to produce noise features from an image. Inspired by recent progress on Steganalysis Rich Models (SRM) for image manipulation detection [10], we use SRM kernels to extract the local noise features from the static images. The SRM use statistics of neighboring noise residual samples as features to capture dependency changes caused by embedding. Zhou et al. [33] use SRM residuals, together with the RGB image to localize corrupted regions in images. Following their results, we use the same three SRM kernels they utilize.

C. ORB-Features-Based Loss

We use ORB features because they allow real-time performance, and provide good invariance to changes in viewpoint and illumination, and therefore are useful for visual SLAM and place recognition, as demonstrated in the popular ORB-SLAM [19] and its binary bag-of-words [11]. The following sections summarize how we have adapted this new loss based on ORB features detector, descriptors and orientation.

1) Detector: ORB Detector is based on FAST algorithm [23]. It takes one parameter, the intensity threshold t between the center pixel p, I_p, and those in a circular ring around the center. If there exists a set of contiguous pixels in the circle which are all brighter than I_p+t, or all darker than I_p-t, the pixel p would be a keypoint candidate. Then the Harris corner measure is computed for each of these candidates, and the target N keypoints with the highest Harris measure are finally selected. ORB uses a scale pyramid of the image and extracts FAST features at each level in the pyramid.

Bringing this to a differentiable solution, we have approximated the FAST corner detection and have used instead a convolution with the kernels shown in Fig. 3. Convolving the image with these kernels for different kernel sizes yields its corner response for the different image pyramid levels. We keep the maximum score per pixel and per level, and we then raise each element to its 2nd power –to leverage positive and negative responses– and subtract a value, equivalent to the FAST threshold t. This, followed by a sigmoid operation, can be seen as the probability of a pixel of being a FAST feature. The features for the output and target images are computed following this procedure. We define this network as det, and the corresponding loss $\mathcal{L}_{det}(G)$ can be expressed as

$$\mathcal{L}_{det}(G) = -\mathbb{E}_{x,y}[w_{det}\cdot(det(y)\cdot\log(det(\hat{y}))+ (1-det(y))\cdot\log(1-det(\hat{y})))], \quad (3)$$

where $\hat{y}=G(x,m)$ and w_{det} is calculated following Eq. 4. This weights definition allows us to leverage the uneven distribution of non-features and features pixels, and follows a Hamming distribuion in order to affect only those image regions with a wrong feature response. N stands for the number of pixels in the features map, and N_f represents the number of pixels in the response map $det(y)$ where $det(y)>0.5$, *i.e.*, the number of FAST features in the current objective frame.

$$w_{det} = \begin{cases} \frac{N}{N_f}, & det(y)>0.5 \quad \text{and} \quad det(\hat{y})\leq 0.5 \\ \frac{N}{N-N_f}, & det(y)\leq 0.5 \quad \text{and} \quad det(\hat{y})>0.5 \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

According to our results, the optimum number of image pyramid levels for this objective is 1. More levels lead to a greater training time and the results are barely influenced. This is coherent with the idea that we want to maximize the sharpness of small features rather that of the big corners. These convolutions have been applied

Fig. 3: Some of the kernels used to obtain corner responses. The 12 black pixels have a value of $-1/12$, the gray pixels are set to 0, and the white pixel is set to 1. A very positive or negative response is obtained when convolving them with a corner area in an image.

with a stride of 5, offering a good trade-off between training time and good results.

Other approaches have tried before to include a similar loss inside a CycleGAN framework [21]. The work by Porav *et al.* uses the SURF detector [3], which is already differentiable, but does not compute a binary loss. They compute a more traditional L1 loss between the blob responses of both output and ground-truth images. Computing a binary loss as in Eq. 4 allows us to have more emphasis on the high-gradient areas.

2) Orientation: Once FAST features have been detected, the original ORB work extracts their orientation to provide them with rotation invariance. This is done by computing its orientation $\theta = \text{atan2}(m_{01}, m_{10})$ and its intensity centroid $C = (\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}})$, where $m_{pq} = \sum_{x,y} x^p y^q I(x,y)$ are the moments of an image patch. We have created three 14-pixel-radius circular kernels with the values x, y and 1 (centered in 0), so that when convolving the image with them, we obtain their respective patch moments m_{01}, m_{10} and m_{00}. We define this network as ori, and the objective of its corresponding loss is that the "fake" static image detected features, $det(G(x,\, m))$, have the same orientation parameters m_{01}, m_{10} and m_{00} than the ground-truth static image detected features, $det(y)$. This loss $\mathcal{L}_{ori}(G)$ can be expressed as $-\mathbb{E}_{x,y}[w_{ori} \cdot ||ori(y) - ori(\hat{y})||_1]$. Even though these convolutions are applied to the whole image with a stride of 5 –as in the detection loss–, the weighting term w_{ori} has a value of 1 if a FAST feature has been detected in either the ground-truth static image or the output image. Otherwise, the weighting term w_{ori} is set to 0.

3) Descriptor: The ORB descriptor is a bit string description of an image patch constructed from a set of binary intensity tests. Consider a smoothed image patch, **p**, a binary test $\tau(\mathbf{p};x,y)$ is defined by 1 if $\mathbf{p}(x) < \mathbf{p}(y)$, and by 0 if $\mathbf{p}(x) \geq \mathbf{p}(y)$, where $\mathbf{p}(x)$ is the intensity of **p** at a point x. The feature is then defined as a vector of n binary tests, following the equation $f_n(\mathbf{p}) = \sum_{1 \leq i \leq n} 2^{i-1} \tau(\mathbf{p};x_i,y_i)$. As in Rublee et al. [24]'s work, we use a Gaussian distribution around the center of the patch and a vector length $n = 256$. This can be achieved in a convolutional manner by creating n kernels with all values set to 0 except for those in the positions x and y, whose values would be $+1$ and -1. Convolving an image with these n kernels yields each pixel's ORB descriptor –a negative output corresponds to the bit value 0 and a positive one to 1–. This convolution is followed by a sigmoid activation function. We define this network as $desc$, and the corresponding loss $\mathcal{L}_{desc}(G)$ can be expressed as

$$\mathcal{L}_{desc}(G) = -\mathbb{E}_{x,y}[w_{desc} \cdot (desc(y) \cdot \log(desc(\hat{y})) + (1 - desc(y)) \cdot \log(1 - desc(\hat{y})))], \quad (5)$$

where w_{desc} follows a Hamming distribution, *i.e.*, $w_{desc} = 1$ if $desc(y) > 0.5$ and $desc(\hat{y}) \leq 0.5$, or if $desc(y) \leq 0.5$ and $desc(\hat{y}) > 0.5$. Otherwise, $w_{desc} = 0$. This loss is back-propagated to the whole image, whether a feature has been detected or not, since it helps keeping the image statistics.

All these losses are combined into one loss $\mathcal{L}_{ORB}(G) = \lambda_{det}\mathcal{L}_{det}(G) + \lambda_{ori}\mathcal{L}_{ori}(G) + \lambda_{desc}\mathcal{L}_{desc}(G)$. The values for the weights of the different losses λ_{det}, λ_{ori} and λ_{desc} have been chosen experimentally, and are 10, 0.1 and 1 respectively.

The features detection, orientation and descriptor maps can be computed in a parallel way to decrease the training time, since their computation is not necessarily sequential.

Finally, the generator's job can be expressed as $G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda_1 \cdot \mathcal{L}_{L1}(G) + \mathcal{L}_{ORB}(G)$.

D. Dynamic Objects Semantic Segmentation

Deep neural networks excel at semantic segmentation, as they can be trained end-to-end to accurately classify multiple object categories in an image at pixel level. However, few architectures have a good quality/computation trade-off. ERFNet, the recent work of Romera et al. [22] runs in real time while providing accurate semantic segmentation.

We have fine tuned the ERFNet model with encoder and decoder both trained from scratch on Cityscapes train set [7] to adjust it to our inpainting approach by back-propagating the loss of the semantic segmentation and the adversarial loss of our final inpainting model $\mathcal{L}_{cGAN}(G,D)$. Its objective is to produce an accurate semantic segmentation y_{SS}, but also to fool the discriminator D. The latter objective might occasionally lead the semantic segmentation network to not only recognize dynamic objects but also their shadows.

Once the semantic segmentation of the RGB image is done, we can select those classes known to be dynamic (vehicles and pedestrians). This has been done by applying a *SoftMax* layer, followed by a convolutional layer with a kernel of $n \times 1 \times 1$, where n is the number of classes, and with the weights of those dynamic and static channels set to w_{dyn} and w_{stat} respectively. w_{dyn} and w_{stat} are calculated following $w_{dyn} = \frac{n - n_{dyn}}{n}$ and $w_{stat} = -\frac{n_{dyn}}{n}$, where n_{dyn} stands for the number of dynamic existing classes. A positive output corresponds to a dynamic object, where as a negative one corresponds to a static one.

The consequent output passes through a $Tanh$ layer to obtain the wanted dynamic/static mask. Note that the defined weights w_{dyn} and w_{stat} are not changed during training time.

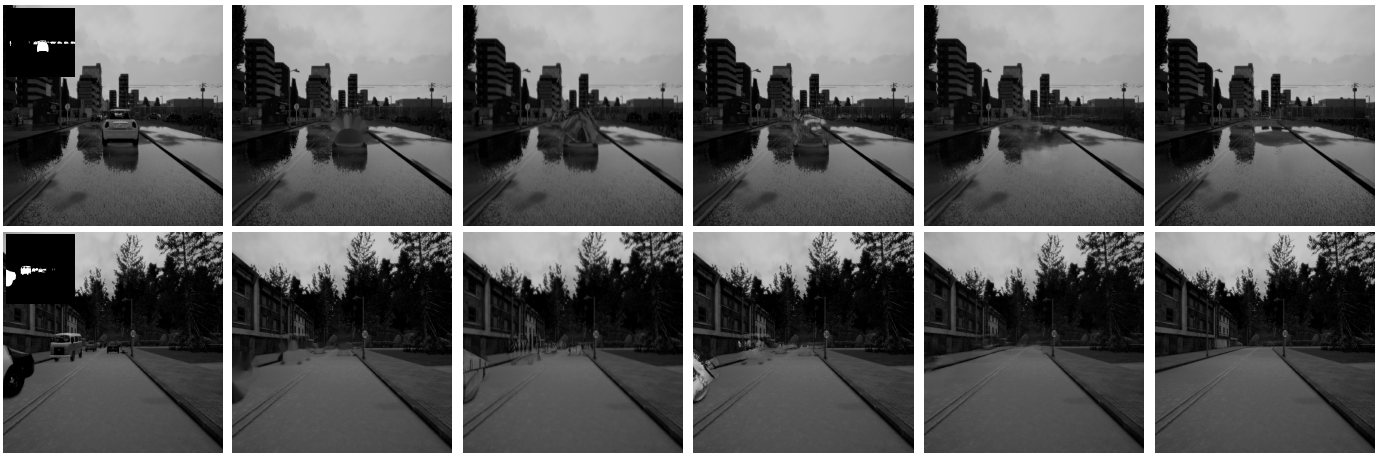
IV. Experiments

A. Data Generation

We have explored our method using CARLA [8]. CARLA is an open-source simulator for autonomous driving research, that provides open digital assets (urban layouts, vehicles, pedestrians, *etc.*), and supports flexible specification of sensor suites and environmental conditions. We have generated over 12000 image pairs consisting of a target image captured with neither vehicles nor pedestrians, and a corresponding input image captured at the same pose with the same illumination conditions, but with vehicles and people moving around. Their semantic segmentation has also been captured.

Experiment	$G(x)$ $D(x,y)$	$G(x,m)$ $D(x,y)$	$G(x,m)$ $D(x,y,m)$	$G(x,m)\|^w$ $D(x,y,m)\|^w$	$G(x,m)\|^w$ $D(x,y,m,n)\|^w$	$G(x,m)\|^w_{ORB}$ $D(x,y,m,n)\|^w$
$L_1(\%)$ - / In / Out	1.98 / 10.04 / 1.70	2.13 / 7.70 / 1.95	3.03 / 7.37 / 2.92	2.00 / 5.73 / 1.88	**0.96** / **4.99** / **0.80**	1.46 / 5.16 / 1.32
$Feat$ - / In / Out	1.29 / 8.75 / 1.04	1.95 / 7.48 / 1.77	1.44 / 7.92 / 1.26	1.45 / 5.75 / 1.33	0.80 / 5.84 / 0.62	**0.70** / **5.14** / **0.53**
$PSNR$ - / In / Out	29.9 / 17.4 / 32.8	29.9 / 20.3 / 31.2	28.3 / 20.6 / 29.1	30.5 / 22.6 / 31.4	**34.0** / 23.3 / **36.5**	33.0 / **23.3** / 34.7
$SSIM$ - / In / Out	0.99 / 0.15 / 0.99	0.99 / 0.33 / 0.99	0.98 / 0.49 / 0.99	0.99 / 0.61 / 0.99	**1.00** / 0.65 / **1.00**	1.00 / **0.66** / 1.00

TABLE I: Quantitative evaluations of the achievements carried out by our contributions in the inpainting task on the test synthetic images. The best results for almost all the inpainting metrics (L_1, $PSNR$ and $SSIM$) are obtained with the generator $G(x,m)|^w$ and the discriminator $D(x,y,m,n)|^w$. More correct features ($Feat$ metric) are detected though when adding the features based loss $G(x,m)|^w_{ORB}$.



(a) Input (b) Geo [27] (c) Lea1 [31] (d) Lea2 [14] (e) Ours (f) Ground-truth

Fig. 4: Qualitative comparison of our method (e) against other inpainting techniques (b), (c), (d) on our synthetic dataset.

B. Inpainting

Here we report the improvements achieved by our framework for inpainting. Table I shows our ablation study for the different reported inputs and losses. We follow previous inpainting works and report the L_1, $PSNR$ and $SSIM$ errors [30], as well as a feature-based metric $Feat$. This last metric compares the FAST features detection as explained in subsection III-C for the output and ground-truth images.

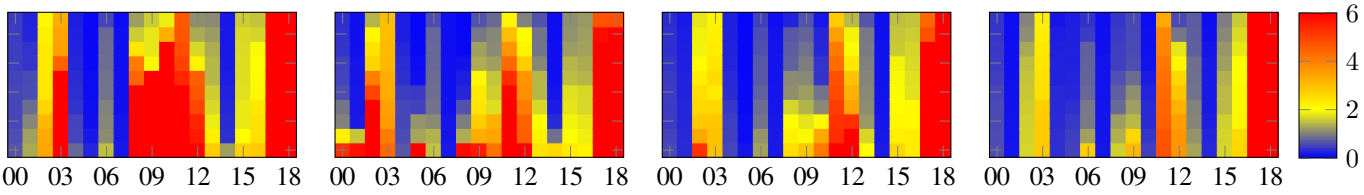
Adding the dynamic/static mask as input for both the generator and discriminator, and leveraging the unbalanced quantity of static and dynamic data within the dataset helps obtaining better inpainting results within the images hole regions –In–, at the expense of having slightly worse quality results in the non-hole regions –Out–. Providing the discriminator with the images noise makes it learn better to distinguish between real and fake images, and therefore the generator learns to produce more realistic images. The ORB-based loss leads to slightly worse inpainting results –according to L_1, PSNR and SSIM metrics–, but renders this approach more useful for both localization and mapping tasks since more correct features are created.

Baselines for Inpainting We compare qualitatively and quantitatively our method with four other approaches, two non-learning approaches **Geo1** [27] and **Geo2** [4], and two deep-learning based methods **Lea1** [31] and **Lea2** [14]. Since both **Lea1** and **Lea2** are methods conceived for general inpainting purposes, we directly use their released models [31, 14]. We provide them with the same mask than to our method to generate the holes in the images. We evaluate qualitatively on the 3000 images from our synthetic test dataset (Fig. 4). Note that results generated with both **Lea1** and **Lea2** have been generated with the color images and then converted to gray scale for visual comparison. Visually, we see that our method obtains a more realistic output –these results are computed without the ORB loss for an inpainting oriented comparison–. Also, it is the only one capable of removing the shadows generated by the dynamic objects even though they are not included in the dynamic/static mask (Fig. 4 row 1). The utilized masks are included in the images in Fig. 4a.

Table II shows the quantitative comparison of our method against **Geo1**, **Geo2**, **Lea1** and **Lea2** on our CARLA dataset. For a fair comparison, we only report the different errors within the images hole regions since the other methods only significantly modify such parts. By following these results, we can claim that our method outperforms both qualitatively and quantitatively the other approaches in such task.

	Geo1 [27]	Geo2 [4]	Lea1 [31]	Lea2 [14]	Ours
$L_1(\%)$	6.55	6.69	12.35	9.50	**4.99**
$Feat$	6.24	6.33	11.19	10.63	**5.84**
$PSNR$	21.03	20.90	15.56	18.23	**23.25**
$SSIM$	0.479	0.467	0.213	0.316	**0.646**

TABLE II: Quantitative comparison of our inpainting method.



(a) Dynamic images (b) Dynamic images w/ mask (c) Our images w/ the ORB loss (d) Static images

Fig. 5: The vertical axis shows the 10 repetitions of every test on the 19 sequences (horizontal axes) in account for the non-deterministic nature of ORB-SLAM. (a) shows the ORB-SLAM absolute trajectory RMSE [m] for the dynamic images. (b) shows DynaSLAM ATE [5]. (c) shows the ORB-SLAM ATE when using our inpainted frames, and (d) shows the ORB-SLAM ATE for the ground-truth static images.

C. Localization

We have conducted some experiments to measure how much dynamic objects affect localization systems accuracy. SLAM systems conceived for static environments are usually robust to a small portion of images dynamic content. When dynamic objects occupy a significant part of the images, the motion of such objects is incorrectly computed as camera ego-motion.

Baselines for Odometry For these experiments we have chosen the state-of-the-art feature-based system ORB-SLAM [19], since it is ideal to test the influence of our features loss. Fig. 5 displays the ORB-SLAM absolute trajectory RMSE [m] computed for 19 CARLA sequences that do not contain loop closures. Fig. 5a shows the results when many vehicles and pedestrians are moving, and Fig. 5d shows the odometry results for the ground-truth static sequence. We can see that dynamic objects have in many sequences a big influence on ORB-SLAM's performance –03, 08, 09, 10 and 12–. Fig. 5b shows the trajectory error obtained with our previous system DynaSLAM [5]. This system is based on ORB-SLAM and uses the semantic segmentation network Mask R-CNN [13] to detect the moving objects and not extract ORB features within them. Even though better odometry results are obtained –compared against using the raw dynamic images–, this experiment shows that using static images leads to a more accurate camera tracking. One reason for this is that dynamic objects might occlude the nearby regions in the scene, which are the most reliable for camera pose estimation. Another reason might be that using dynamic objects masks does not yield anymore an homogeneous features distribution within the image, and pose optimization could be degraded and drifting could increase if the features do not follow such distribution. Finally, Fig. 5c shows the ORB-SLAM error when using our inpainted images. Our odometry results show that better results are usually obtained when using our inpainted images, even though there is still room for improvement regarding the inpainting task. The inpainting is realistic enough to provide the SLAM system with consistent features that are useful for localization.

	#1 Input	#1 Output	#1 Target	#2 Input	#2 Output	#2 Target
BoW	1.29 %	4.46 %	5.40 %	17.41 %	29.57 %	29.80 %

TABLE III: Percentage of frames in which place recognition is successful for the dynamic, inpainted and ground-truth static sequences.

D. Visual Place Recognition

Visual place recognition is an important task for visual-based navigation. As well as having an accurate visual odometry, SLAM systems show algorithms for place recognition. They are useful when revisiting places in order to perform loop closure and correct the accumulated drift along long trajectories. Bags of visual words (BoW) is the common approach to perform such task, as can be seen in ORB-SLAM [19].

Baseline for Visual Place Recognition Here we show a place recognition experiment performed with the ORB-SLAM's bag of words module based on the work by Gálvez-López and Tardos [11]. We have generated two CARLA sequences with loop closures with and without dynamic objects. To test the BoW algorithm, we have ran monocular ORB-SLAM with the dynamic raw frames, with the ones processed by our generative model and with the ground-truth static images ten times each. We have computed the number of frames in which place recognition is successful and show the percentage with respect to the total amount of frames in which place recognition should be successful. Table III shows the mean results for each trajectory and type of input. Even though the inpainting algorithm is not perfect and might introduce false appearance, it brings closer images from the same place with different dynamic objects while pulling apart images from different places but with similar dynamic objects. Reporting the results for the ground-truth static images allows us to to see to what extent our framework could report an improvement for place recognition, since failure cases can be also due to view-point change or different weather conditions.

V. Conclusions

We have presented an end-to-end deep learning framework that translates images that contain dynamic objects within a city environment, such as vehicles or pedestrians, into a realistic image with only static content. These images provide a more informative understanding of the stationary surroundings, and are suitable for visual localization tasks thanks to a new loss based on steganalysis techniques and ORB features maps. We motivate this extra complexity by showing quantitatively that ORB-SLAM obtains a higher accuracy when utilizing the images synthesized with this loss. Finally, an architectural nicety is that our system processes the image streams outside of the localization pipeline, either offline or online, and hence can be used naturally as a front end to many existing systems.

References

[1] Pablo F Alcantarilla, José J Yebes, Javier Almazán, and Luis M Bergasa. On combining visual SLAM and dense scene flow to increase the robustness of localization and mapping in dynamic environments. In *ICRA*, pages 1290–1297. IEEE, 2012.

[2] Dan Barnes, Will Maddern, Geoffrey Pascoe, and Ingmar Posner. Driven to distraction: Self-supervised distractor learning for robust monocular visual odometry in urban environments. In *ICRA*, pages 1894–1900. IEEE, 2018.

[3] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[4] Marcelo Bertalmio, Andrea L Bertozzi, and Guillermo Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[5] Berta Bescos, José M Fácil, Javier Civera, and José Neira. DynaSLAM: Mapping, Tracking and Inpainting in Dynamic Scenes. *IEEE Robotics and Automation Letters*, 3:4076 – 4083, 2018.

[6] Berta Bescos, Roland Siegwart, José Neira, and Cesar Cadena. Empty Cities: Image Inpainting for a Dynamic-Object-Invariant Space. *IEEE International Conference on Robotics and Automation*, 2019.

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual CoRL*, 2017.

[9] Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In *CVPR*, 2001.

[10] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, 2012.

[11] Dorian Gálvez-López and Juan D Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, 2012.

[12] Miguel Granados, Kwang In Kim, James Tompkin, Jan Kautz, and Christian Theobalt. Background inpainting for videos with dynamic objects and a free-moving camera. In *ECCV*, 2012.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[14] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (TOG)*, 36(4):107, 2017.

[15] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017.

[16] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, pages 225–234, 2007.

[17] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *ECCV*, 2018.

[18] Montiel J. M. M. Mur-Artal, Raúl and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. doi: 10.1109/TRO.2015.2463671.

[19] Raul Mur-Artal and Juan D Tardós. ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras. *IEEE T-RO*, 2017.

[20] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016.

[21] Horia Porav, Will Maddern, and Paul Newman. Adversarial training for adverse conditions: Robust metric localisation using appearance transfer. *IEEE International Conference on Robotics and Automation*, 2018.

[22] Eduardo Romera, José M Alvarez, Luis M Bergasa, and Roberto Arroyo. ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 2018.

[23] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *ECCV*. Springer, 2006.

[24] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE, 2011.

[25] Martin Runz, Maud Buffier, and Lourdes Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 10–20. IEEE, 2018.

[26] Wei Tan, Haomin Liu, Zilong Dong, Guofeng Zhang, and Hujun Bao. Robust monocular SLAM in dynamic environments. In *ISMAR*, pages 209–218, 2013.

[27] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of graphics tools*, 9(1): 23–34, 2004.

[28] Ries Uittenbogaard. Moving object detection and image inpainting in street-view imagery. 2018.

[29] Youbing Wang and Shoudong Huang. Motion segmentation based robust RGB-D SLAM. In *WCICA*. IEEE, 2014.

[30] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[31] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative Image Inpainting with Contextual Attention. *arXiv preprint arXiv:1801.07892*, 2018.

[32] Guoxiang Zhou, Berta Bescos, Marcin Dymczyk, Mark Pfeiffer, José Neira, and Roland Siegwart. Dynamic objects segmentation for visual localization in urban environments. *arXiv preprint arXiv:1807.02996*, 2018.

[33] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Learning rich features for image manipulation detection. *arXiv preprint arXiv:1805.04953*, 2018.