Computer Vision and Image Understanding 189 (2019) 102803

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

IterGANs: Iterative GANs to learn and control 3D object transformation[☆]

Ysbrand Galama[a,1], Thomas Mensink[b,*,1]

[a] TomTom, Amsterdam, The Netherlands
[b] Google Research, Amsterdam, The Netherlands

ARTICLE INFO

MSC:
68T45
68U05

Keywords:
Generative Adversarial Networks
Object Transofoormation
Novel Viewpoint Estimation

ABSTRACT

We are interested in learning visual representations which allow for 3D manipulations of visual objects based on a single 2D image. We cast this into an image-to-image transformation task, and propose Iterative Generative Adversarial Networks (IterGANs) which iteratively transform an input image into an output image. Our models learn a visual representation that can be used for objects seen in training, but also for never seen objects. Since object manipulation requires a full understanding of the geometry and appearance of the object, our IterGANs learn an implicit 3D model and a full appearance model of the object, which are both inferred from a single (test) image. Two advantages of IterGANs are that the intermediate generated images can be used for an additional supervision signal, even in an unsupervised fashion, and that the number of iterations can be used as a control signal to steer the transformation. Experiments on rotated objects and scenes show how IterGANs help with the generation process.

1. Introduction

In this paper we are interested in manipulating visual objects and scenes, without resorting to external provided (CAD) models or advanced (3D/depth) sensing techniques. To be more specific, we focus on rotating objects and rotating the camera viewpoint of scene from a single 2D image. Manipulating objects require an expectation about the appearance and the geometrical structure of the *unseen* part of the object. Humans clearly have such an expectation based on an understanding of the physics of the world, the continuity of objects, and previously seen (related) objects and scenes. We aim to learn such an 3D understanding, inferred from single 2D images.

In order to learn a representation for object manipulation, we cast this problem into an image-to-image transformation task, with the goal to transform an input image following a given 3D transformation to an target image. For this kind of object manipulation, often either stereoscopic cameras (Ko et al., 2007; Bruno et al., 2010) or temporal data streams (Pollefeys et al., 2008; Gibson et al., 2003) have been used to infer depth cues, while our aim is to obtain the target image from a single input image only. Similarly as humans are able to do so with one eye closed (Vishwanath and Hibbard, 2013), there has also been works that aim to reconstruct 3D from a single image (Saxena et al., 2009; Rematas et al., 2017), however these typically require external provided 3D object models, *e.g*. balloon shapes (Vicente and Agapito, 2013), or focus on a single class of objects only (Park et al., 2017). Our aim, on the other hand, is to learn a general transformation model, which can transform many classes of objects, even objects never seen at train time, based on the fact that appearance and geometrical continuity are (mostly) not object specific but generally applicable.

In this paper, we focus on a specific instance of object manipulation: the object in the target image is a fixed rotation of the input image. For this task, we propose the use of Iterative Generative Adversarial Networks (IterGANs). GANs have been used for many (image) generation tasks (Reed et al., 2016; Denton et al., 2015; Radford et al., 2015), including image-to-image prediction (Isola et al., 2017; Zhu et al., 2017b). Our proposed IterGANs are a special kind of GANs, where the input image is fed to the generator, and the output of the generator is iteratively fed back into the generator for a fixed number of iterations to generate the output image. The number of iterations is either predefined, or used as a control mechanism to steer the degree of image rotation.

The underlying hypothesis of the IterGAN is, it is easier to rotate an object for a few degrees, than for a large rotation. This has experimentally been shown by Shepard and Metzler (1971), by measuring the reaction time of humans to identify whether two rotated objects are the same. The study shows that there exists a linear dependence between the reaction time and the degree of rotation between the two objects, see Fig. 1.

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.cviu.2019.102803.

* Corresponding author.
E-mail addresses: ysbrand.galama@tomtom.com (Y. Galama), mensink@google.com (T. Mensink).

1 The main body of this research was performed while both authors where with the Computer Vision group, University of Amsterdam, the Netherlands.

https://doi.org/10.1016/j.cviu.2019.102803
Received 16 July 2018; Received in revised form 14 June 2019; Accepted 21 August 2019
Available online 28 August 2019

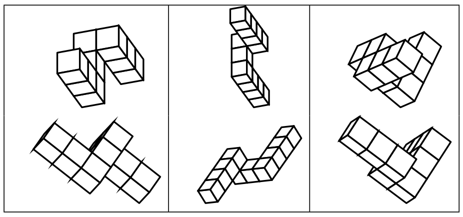

Fig. 1. Shepard and Metzler (1971) demonstrated the time needed to identify matching pairs of objects depends on the degree of rotation, more rotated objects take longer to identify. We introduce IterGANs, which *iteratively* rotate objects for a few degrees each time to reach a target rotation.

The iterative nature of IterGANs has two particular advantages over a single image-to-image GAN for image manipulation. First, IterGANs break long range dependencies between the pixels of the input image and the pixels of the output image. A fundamental difference between image manipulation and the image-to-image tasks explored in Isola et al. (2017) is, that when translating a map into an aerial image there exists a one-to-one pixel relation between the input and the output image. In the case of object manipulation, however, pixels have long range dependencies, depending on the geometry and the appearance of the object and the required degree of rotation. IterGANs break these long dependencies into a series of shorter dependencies. Second, IterGANs allow to use intermediate loss functions measuring the quality of the series of intermediate generated images to improve the overall transformation quality.

This paper is an extended version of our ICLR Workshop paper (Galama and Mensink, 2018). This extended version, includes an supervised intermediate loss function, a stepwise learning approach, and extensive experimental results, on both the ALOI dataset (Geusebroek et al., 2005) for object rotation, and the VKITTI dataset (Gaidon et al., 2016) for camera viewpoint scene rotation. Our paper is organised as follows. Next we will discuss some of the most related work in image-to-image object manipulation. In Section 3 we introduce IterGANs and propose extended loss functions on the intermediate generated images. We show extensive experimental results in Section 4, on the ALOI dataset (Geusebroek et al., 2005) for object rotation, and the VKITTI dataset (Gaidon et al., 2016) for camera viewpoint scene rotation. Finally, we conclude the paper in Section 5.

2. Related work

There is vast amount of related work in the field of 3D reconstruction and image generation. Here, we only highlight the most relevant methods with respect to our proposed models. For a more detailed overview see for 3D reconstruction (Li et al., 2015) and image generation (Zhu et al., 2017b).

3D reconstruction from 2D images. There are several techniques for generating 3D models from 2D data, differing in both the type of data, and the type of environment. It is possible to create a point-cloud from video using Structure from Motion (Koenderink and Van Doorn, 1991; Nistér, 2005), or to fit or deform polygonal objects to images (Rematas et al., 2017; Suveg and Vosselman, 2004; Vicente and Agapito, 2013).

In recent years, there have also been attempts to use deep learning. These techniques also allow forgoing 3D models, thus using only 2D images to describe the 3D environment. Such an approach has been used to classify objects (Su et al., 2015), generate different viewpoints from descriptors (Dosovitskiy et al., 2015) or creating the frames for 3D movies (Xie et al., 2016).

In contrast to their work, we focus on changing the orientation of an object in the image or the viewpoint of a scene in a given image. Therefore our models not only need to construct the view (Dosovitskiy et al., 2015), but also perceive an input image, and capture more than only a disparity map, *e.g.* Xie et al. (2016). Since our desired output is an image, we use image-to-image GANs to transform the image.

Image manipulation with GANs. Generative Adversarial Networks (GANs, Goodfellow et al., 2014) have been shown to be successful for generating visually pleasing images, *e.g.* Reed et al. (2016), Denton et al. (2015) and Radford et al. (2015). For image-to-image translation, where the goal is to translate an input images (*e.g.* sketch) to an output image (*e.g.* photo), conditional GANs (Mirza and Osindero, 2014) have been used by Isola et al. (2017) in their Pix2Pix paper, where the input image is the conditional. The Pix2Pix paradigm has sparked many image manipulation tasks into image-to-image translation, including ageing a face from a single image (Antipov et al., 2017), or provide it with glasses and a shave (Shen and Liu, 2017), or change the main object of an image, *e.g.* a cat to a dog (Liang et al., 2018). We also use the Pix2Pix image-to-image GANs, but apply and extend it for object manipulation, to generate images of rotated objects/scenes.

Since GANs are notoriously difficult to train, in part due to mode collapse, many variants have been introduced, focusing on the optimisation and the loss functions of the network, *e.g.* Salimans et al. (2016) and Arjovsky et al. (2017). Another line of research is to use the network architecture to guide the learning process, for example by imposing a cyclic or dual GAN architecture (Zhu et al., 2017a; Yi et al., 2017), which exploits the insight that if image A transforms into image $\hat{B}$, then with an inverse transformation image $\hat{B}$ should transform back into A. The same insight of cyclic generation is also used for left–right consistency when depth is generated from a single image, trained on paired left–right images without depth ground-truth (Pilzer et al., 2018). Instead of using two separate generators, one for the transformation and for the inverse transformation, the cyclic reasoning could also be used with a single generator with some control vector (Liang et al., 2018; Choi et al., 2018). We also change the GAN architecture, where we explicitly let the GAN transform the object iteratively, and instead of using a control vector, we rather use the number of iterations of the generator as a control function.

Finally, an important line of research is on improving the quality of the synthesised images. For example by progressively increasing the depth of the generator and the discriminator over the coarse of training (Karras et al., 2018) improves the quality and stability of training in higher resolutions. In Wang et al. (2018) a coarse-to-fine generator is introduced, where first a small image is generated, which is then used as conditional to generate a larger output image. This two step zooming process into the details of a high resolution image, is similar in how our IterGAN model iteratively rotates an object to a final rotation.

Novel viewpoint estimation. A more specific form of image manipulation is *Novel viewpoint estimation*, which has a slightly different goal than 3D reconstruction, since the output is again a 2D image. A new viewpoint can be estimated using voxel projections (Yan et al., 2016), or GANs to estimate the frontal view of a face (Huang et al., 2017), or transforming a single image with viewpoint estimation (Zhou et al., 2016). The quality of the generated image can be improved by using multiple input images (Zhou et al., 2016), or using a second network to refine the results of a flow-network (Park et al., 2017). Existing works on viewpoint transformation have been conducted to synthesise novel views of the same object using synthesised data from CAD models. In our paper, we use a heterogeneous dataset of real world images taken under constraint conditions, instead of synthetic images of a single object class.

3. IterGANs

Iterative GANs are image-to-image GANs, where the generator is called iteratively for k steps:

$$B^k = \mathcal{G}_\theta(\mathcal{G}_\theta(\mathcal{G}_\theta(\mathcal{G}_\theta(\mathcal{G}_\theta(\mathcal{G}_\theta(A)))))) = \mathcal{G}_\theta^k(A), \quad (1)$$

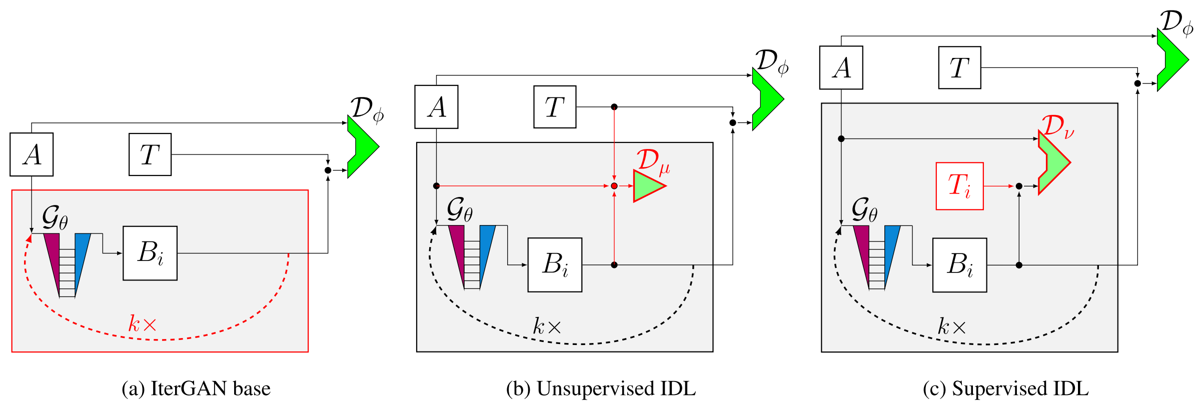

Fig. 2. IterGAN framework: the iterative nature of IterGANs (*left*) allows for additional discriminators on the intermediate generated images to steer the learning. In this paper we introduce two types of intermediate discriminator loss functions (IDL). **Unsupervised IDL** (*middle*) aims to tell apart generated intermediate images (B^i) from real images (A or T). **Supervised IDL** (*right*) aims to discriminate input-generated (A, B^i) image pairs from the real pairs (A, T^i).

where the output image B^k is generated by rotating the input image A in k small steps using the same generator $\mathcal{G}$. While this iterative generation cannot guarantee a specific degree rotation, *e.g.* $6\times5° = 30°$, it breaks the long dependencies between pixels of rotated objects, and therefore the generator should be easier to learn. The number of steps could also be used to control the degree of rotation, by varying the number of iterations, which we explore in Section 3.2. The iterative nature of the IterGAN is illustrated in Fig. 2a, we refer to our IterGAN network as IG.

The iterative nature of IterGANs resembles recurrent neural network modules, like LSTM (Hochreiter and Schmidhuber, 1997) or GRUs (Cho et al., 2014). Typically use cases for RNNs in computer vision include classifying a sequence of inputs, *e.g.* to classify actions in video (Ng et al., 2015), or to produce a sequence conditioned on an input, *e.g.* describe an image with a caption (Vinyals et al., 2015).

In contrast to the first, IterGANs only receive a *single* input, instead of a sequence (sentence or video-frames) and produce a single (image) output. The difference with the latter is more subtle: both the image captioning as well as our IterGANs are conditioned on a single input. However, there are two main differences: (a) LSTMs propagate both the (sampled) output of the previous layer combined with a hidden state, while the proposed IterGANs only propagate the generated image of the previous layer; (b) the quality of the caption is determined by the combination of words, and the quality of an individual word is difficult to measure, while for IterGANs we are mainly interested in the output after iteration k and the quality of each generated image is independently measurable.

Discriminator and generator loss functions. We use the same network architectures as in Pix2Pix (Isola et al., 2017). IterGANs could be seen as an extension of Pix2Pix, and are identical when $k = 1$ is used. Since in IterGANs the same generator is applied iteratively, it also has the same number of parameters as the Pix2Pix models. For training we use the generator and discriminator losses of Isola et al. (2017).:

$$\mathcal{L}_{\mathcal{G}}^{(\text{IG})} = H[\mathcal{D}_\phi(A, B^k), 1] + \lambda_{L_1}\ L_1(B^k, T), \tag{2}$$

$$\mathcal{L}_{D}^{(\text{IG})} = H[\mathcal{D}_\phi(A, T), 1] + H[\mathcal{D}_\phi(A, B^k), 0], \tag{3}$$

where A denotes the input image, T the target image, and $B^k = \mathcal{G}_\theta^k(A)$ the generated output image after step k. The generator ($\mathcal{L}_{\mathcal{G}}$) combines the cross-entropy loss (H) for predicting *real* for the image pair A, B^k, *i.e.* a low loss value is obtained when the discriminator is fooled that B^k is a real image, with the L_1 loss between the generated image B^k and the target image T. The discriminator ($\mathcal{L}_D$) combines the cross-entropy loss for predicting *real* for the input pair A and T and *generated* for the input pair A and B^k.

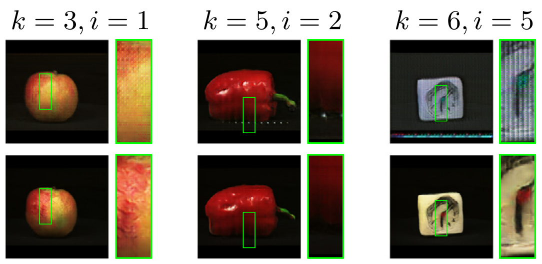

Fig. 3. Examples of generated artefacts in intermediate images (*top row*), which seem independent of specific value for the number of iterations $k = \{3, 5, 6\}$. Interestingly, the artefacts disappear in the final generated image (*bottom row*).

Object mask specific reconstruction loss. The L1 loss equally weights each pixel in the image. For our object rotation, we would like to focus more on the pixels of the rotated object, rather than the (black) background. Therefore, we use a variant of the L1-loss, which uses a binary object mask M:

$$L_1^{\text{M}} = \frac{2}{|M|}\sum_{xy} M_{xy}L_{xy} + \frac{1}{|\neg M|}\sum_{xy} \neg M_{xy}L_{xy}, \tag{4}$$

with $L_{xy} = \sum_c |T_{xyc} - B_{xyc}^k|$,

where $M \in \{0, 1\}$ assigns pixels to object (1) or background (0), these masks are part of the ALOI dataset (Geusebroek et al., 2005). This variant of the L1-loss weights the object twice as important as the background. Models trained with this L_1^{M}-loss are indicated with an ‘M’.

3.1. Intermediate discriminator loss functions

While IterGANs generate intermediate images, the (implicit) assumption that these would be realistic images as well, does not necessarily hold. In fact, preliminary results reveal that, the iterative generator introduces different types of artefacts in the intermediate generated images. Interestingly, these are gone in the final generated image, so repeatedly applying the same generator removes the introduced artefacts. In Fig. 3, we show examples of intermediate generated images when using $k = \{3, 5, 6\}$ and each of these have artefacts, *e.g.* switching colours, adding noise or patterns.

In this section we introduce two types of intermediate discriminator loss functions (IDL), to overcome these artefacts and to improve the final image generation quality. The IDLs aim to guide the learning process to generate realistic intermediate images, by an additional discriminator fed by intermediate generated images. We propose IDLs which do not require additional target images (*unsupervised*), and which do use additional target images of the intermediate images (*supervised*). Both models are illustrated in Figs. 2b and 2c.

Unsupervised intermediate discriminator loss function. The unsupervised IDL requires only the (A,T) image pair, already provided to the GAN, yet it includes an additional discriminator, to tell apart image A or T from any of the generated images $\{B^i\}_{i=1}^k$, unconditioned on the original input image, see Fig. 2b. This additional discriminator guides the GAN to generate intermediate images following the distribution of *real* images A and T. This results in the extended loss functions:

$$\mathcal{L}_{\mathcal{G}}^{(\mathrm{IG_U})} = \mathcal{L}_{\mathcal{G}}^{(\mathrm{IG})} + \lambda_u \; H[\mathcal{D}_\mu(B^i), 1] \tag{5}$$

$$\mathcal{L}_{\mathcal{D}}^{(\mathrm{IG_U})} = \mathcal{L}_{\mathcal{D}}^{(\mathrm{IG})} + \lambda_u \; \left(H[\mathcal{D}_\mu(B^i), 0] + H[\mathcal{D}_\mu(A \vee T), 1]\right) \tag{6}$$

where an additional cross-entropy loss (H) is added for the generator and the discriminator. For each image in a batch, we sample a single B^i uniformly from $\{B^i\}_{i=1}^k$, and either A or T is used, λ_u is an additional hyper-parameter. Since no additional labelled data is required, we call this *unsupervised IDL* and indicate models trained with these losses with 'U'.

Supervised intermediate discriminator loss. The supervised intermediate loss also adds a discriminator, but one conditioned on the input image A, and using the intermediate target images $\{T^i\}_{i=1}^k$ for supervision, see Fig. 2c. It aims to discriminate whether the intermediate images are a generated rotation ($\{B^i\}_{i=1}^k$ or a real rotation ($\{T^i\}_{i=1}^k$) from the object depicted in image A. This discriminator is similar to the existing conditional discriminator used by the GAN, yet the difference is that the goal of the main discriminator is to detect if the output is a real $R°$ rotation of the input A, while the goal of the added discriminator is to accept an arbitrarily rotation of image A. Supervised IDL results in the following extended losses:

$$\mathcal{L}_{\mathcal{G}}^{(\mathrm{IG_S})} = \mathcal{L}_{\mathcal{G}}^{(\mathrm{IG})} + \lambda_s \; H[\mathcal{D}_\nu(A, B^i), 1] \tag{7}$$

$$\mathcal{L}_{\mathcal{D}}^{(\mathrm{IG_S})} = \mathcal{L}_{\mathcal{D}}^{(\mathrm{IG})} + \lambda_s \; \left(H[\mathcal{D}_\nu(A, B^i), 0] + H[\mathcal{D}_\nu(A, T^i), 1]\right) \tag{8}$$

where an additional cross-entropy loss (H) is added on the conditional model, and where (B^i, T^i) are sampled randomly, with $i = \{1, \dots, k-1\}$. Preliminary experiments using also intermediate L_1^M-loss between B^i and T^i did not improve the final quality of the generator, and were therefore not further explored. The proposed losses for the generator and discriminator need the intermediate ground-truth images $\{T^i\}_{i=1}^k$, therefore we call this *supervised IDL*, and indicate models with an 'S'.

3.2. Training iterGANs to control object manipulation

A notable difference between the unsupervised and supervised IDL is that the implicit versus explicit requirement of the generator to rotate an object for a predefined (fixed) degree of rotation. The unsupervised model could rotate any degree as long as the final rotation is correct, while the supervised model, is guided to rotate each iteration for the same degree of rotation. In this section, we follow up on this, by using the number of iterations as a way to control to the desired rotation of the object even more explicit.

The number of iterations could be seen as an explicit control variable, in contrast to learn from implicit control, *e.g.* by adding a control vector to the generator network for a desired output domain/viewpoint, see *e.g.* Zhou et al. (2016), Park et al. (2017) and Choi et al. (2018). Instead of training on input–target pairs with a fixed rotation, we sample a value of $k \in \{1, \dots, 36\}$ and select input–target pairs with corresponding difference of rotation $(5, 10, \dots, 180)$. The generator is repeated k times to generate the desired $5° \times k$ degrees rotation of the input image. When $k > 1$ the IDL can still be used to discriminate the intermediate results sampled from $\{B^i\}_{i=1}^{k-1}$. Models trained using any target rotation are indicated with 'A'.

Table 1
Overview of the IterGANs models used in our experiments, including the naming conventions and the generator and discriminator losses.

Overview of naming of IterGAN models	
IG6	Using fixed iterations $k = 6$ and targets 30°.
IGA	Trained using k as control variable, *c.f.* Section 3.2
IGS	Trained with the stepwise approach, *c.f.* Section 3.2.1
Loss function extensions	
-M	trained with the mask objective, *c.f.* Eq. (4)
-U	trained with **unsupervised** IDL
-S	trained with **supervised** IDL
Generator loss	*Discriminator loss*
$\mathcal{L}_{\mathcal{G}} = H[\mathcal{D}_\phi(A, B^k), 1]$	$\mathcal{L}_{\mathcal{D}} = H[\mathcal{D}_\phi(A, B^k), 0]$
$+\lambda_{L_1} \; L_1 \textit{ or } L_1^{\mathrm{M}}(B^k, T)$	$+H[\mathcal{D}_\phi(A, T), 1]$
$+\lambda_u \; H[\mathcal{D}_\mu(B^i), 1]$	$+\lambda_u \; \left(H[\mathcal{D}_\mu(B^i), 0] + H[\mathcal{D}_\mu(A \vee T), 1]\right)$
$+\lambda_s \; H[\mathcal{D}_\nu(A, B^i), 1]$	$+\lambda_s \; \left(H[\mathcal{D}_\nu(A, B^i), 0] + H[\mathcal{D}_\nu(A, T^i), 1]\right)$

3.2.1. Learning with a stepwise approach

Finally we include a model which learns first to rotate objects by a small angle, before learning to propagate the generated images for a larger rotation angle. On one hand this is inspired on the insight that learning small rotations is easier than larger rotations. On the other hand, on our experiments which shows that IterGANs – even with supervised IDLs – produce artefacts in the intermediate generated images. This can be overcome if the network first learn to produce high quality small rotations, and then adjust these weights when propagating images for larger rotations. Therefore we start training the model using $k = 1$ for a few epochs, in order to learn a 5° rotation. We then increase the range of k from 1 to 36 in 3 steps, to increase the degree of rotation from 5° to the full 180° rotation. Models trained using this stepwise approach are indicated with 'S'.

4. Experiments

Dataset. For most of our experiments we use the Amsterdam Library of Object Images (ALOI) (Geusebroek et al., 2005) dataset, a set of real images of 1000 household objects, photographed under constrained lighting and from different viewing directions using a turntable setup. Each object is rotated 360° in steps of 5° (resulting in 72 images per object), padded and scaled to fit 256 × 256.

For training we have 28.8k images from 800 objects (36 pairs of 30° rotation per object). For testing we use two sets, (i) **seen objects**: 3.6k images from 100 objects from the training set, yet with different start/end rotations; (ii) **unseen objects**: 3.6k images from 100 objects not present during training.

Evaluation measures. Evaluating the quality of generated images is hard by itself (Wang et al., 2002; Salimans et al., 2016), therefore we use different evaluation measures:

1. the pixel-wise L_1 loss (also used in Isola et al., 2017);
2. the object specific mask loss (L_1^{M} loss, Eq. (4));
3. the Kullback–Leibler (KL) Label divergence:

   $$D_{\mathrm{KL}} \left(p(y|B^k) \; \| \; p(y|T) \right), \tag{9}$$

   to measure the similarity of the label distributions $p(y|\cdot)$, obtained from a pre-trained VGG16, between the generated image B and the target image T;
4. Structural Similarity (SSIM) (Wang et al., 2004); and
5. Visual Information Fidelity on the pixel domain (VIFp) (Sheikh and Bovik, 2006).

The KL-Label measure is inspired on the KL measure used to measure specificity and diversity in Salimans et al. (2016), yet in our case the generated image should be realistic and therefore have a similar label distribution to the target image. Note that for L_1, L_1^{M} and D_{KL} better performing models obtain a lower (↓) score, while for SSIM and VIFp better performing models obtain a higher (↑) score.

Table 2
Comparison of several IterGAN variants and baselines, including supervised and unsupervised intermediate discriminators and training using the L_1^{M} loss. While the performance differences between some of the models are small, the results are statistical significant for IG6 over P2P-30, and IG6-MU and IG6-MS over P2P-M30, see text for details. *We conclude, that the iterative nature of IterGANs are beneficial for learning object rotation.*

		Identity	Projective	P2P-6×5	P2P-30	IG6	IG6-U	IG6-S	P2P-M30	IG6-M	IG6-MU	IG6-MS
L_1	↓	.022 ± .020	.138 ± .032	.021 ± .009	.014 ± .009	.014 ± .008	.016 ± .010	.016 ± .010	.013 ± .009	.013 ± .009	**.012** ± **.008**	**.012** ± **.008**
L_1^{M}	↓	.298 ± .154	.457 ± .156	.299 ± .092	.210 ± .092	.200 ± .084	.239 ± .099	.247 ± .094	.157 ± .092	.162 ± .060	**.147** ± **.055**	.152 ± .058
D_{KL}	↓	***.480*** ± ***.520***	1.407 ± .810	1.488 ± 1.333	1.329 ± 1.333	1.234 ± 1.261	1.567 ± 1.301	1.555 ± 1.306	1.324 ± 1.333	1.219 ± 1.240	**1.019** ± **1.134**	1.057 ± 1.125
SSIM	↑	.915 ± .068	.547 ± .082	.910 ± .056	.938 ± .047	.941 ± .045	.930 ± .049	.931 ± .048	.937 ± .052	.940 ± .047	**.946** ± **.043**	.943 ± .046
VIFp	↑	.446 ± .122	.154 ± .050	.424 ± .098	.504 ± .097	.513 ± .097	.476 ± .096	.469 ± .095	.492 ± .107	.503 ± .102	**.522** ± **.097**	.517 ± .101

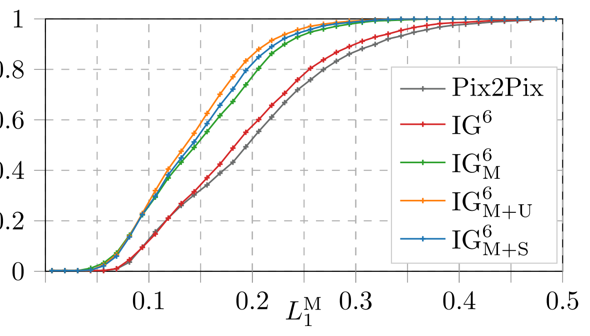

Fig. 4. Evaluation of the L_1^{M}-objective and Intermediate Discriminator Losses, showing the data (%) for a specific loss value. The L_1^{M} objective increase performance significantly, and combined with unsupervised IDL it performs best.

Training procedure. For training all models, we followed the training setup of Pix2Pix (Isola et al., 2017) as closely as possible. All models are trained for 20 epochs, alternating between one gradient descent step on $\mathcal{L}_{\mathcal{G}}$ and one step $\mathcal{L}_D$. For $\mathcal{L}_{\mathcal{G}}$ we maximise $\log(H)$ instead of minimising $\log(1-H)$, while for $\mathcal{L}_D$ *is* we divide the loss by 2. As hyperparameters we use $\lambda_{L_1} = 100$ (cf. Pix2Pix) and we set $\lambda_u = \lambda_s = 0.1$.

The final losses used for the generator and discriminator, combined from Eqs. (2)–(8), are summarised in Table 1. To counteract bad initialisation, each model was trained multiple (3) times, and the best were used for comparison. Source code, the trained models, and the train and test splits used are available on GitHub.[2]

Models and baselines. In the experiments below, we use the following baselines, models and naming conventions:

Id the identity mapping (B = A);
Proj a non-learning projective transformation, which rotates the image plane assuming a pinhole camera to compute point-pairs for the transformation matrix;
P2P direct image rotation using Pix2Pix, we compare two variants: P2P-30 where a 30° rotation is learned directly (identical to IterGAN with k = 1), and P2P-6×5 where a 5° rotation is learned, following the insight that smaller rotations are easier and applied 6 times to obtain the target 30° rotations; and
IG6 the proposed IterGAN models using $k = 6$, including models with the unsupervised intermediate discriminator (IG6-U) and the supervised (IG6-S) variant. We use $k = 6$ for most IG models, since the targets are available at 5° intervals. We also compare training with all available rotations IGA and *stepwise* approach IGS, see also Table 1.

For the learning based models we include models trained with the L1 and L_1^{M} objective (Eq. (4)), the latter denoted with **M**, *e.g.* IG6-MU for the IG6 model with the unsupervised intermediate discriminator, trained using the L_1^{M} loss.

[2] Code available at: https://github.com/tomaat/itergan.

4.1. IterGANs on ALOI

In the first experiment we compare the baseline methods to several IterGANs variants. The target is to rotate an input image for 30° and we evaluate all measures on the *seen-objects* test set.

The results of this experiment are shown in Table 2. From these results, we observe that for evaluation measures L_1, L_1^{M}, SSIM, and VIFp the learning methods outperform the non-learning baselines, while for D_{KL} the identity projection is very strong. The strong performance of the identity projection for D_{KL} is explained by the VGG16 network, which is (partly) trained to be invariant to object viewpoint, while subtle differences in local image statistics can have a large impact. In Fig. 5 we show some qualitative results of 30° rotations of images from the seen objects test set (top three rows).

In general the IG models improve over P2P baselines. Albeit these differences are small, they are significant according to the non-parametric Friedman test where each image-pair judges different models. The test finds models which are significantly ranked higher or lower than the others, *e.g.* IG6 is significantly better than P2P-30 with a $p \leq 0.01$ based on the L_1^{M} metric, and the same holds for IG6-MU and IG6-MS over P2P-M30.

Object mask objective (L_1^M). Learning with the L_1^{M} objective shows a clear increase in performance of any model and any evaluation measure, see Table 2. This holds especially for the L_1^{M} and D_{KL} evaluation measures. Probably because the learner is now informed about the important region of the target image, which yields higher quality of the final generated image.

Intermediate discriminator losses. Here we look in more detail at the performance when the Intermediate Discriminator Losses (IDL) are added. The results from Table 2 are detailed in Fig. 4, where we show the cumulative data percentage for a given loss, *e.g.* for the IG6 about 60% of all images have a loss < .2, while for IG6-MU that is about 85%. From Table 2, we observe that IDL is only beneficial when combined with the L_1^{M}-objective. Apparently there is an interplay between the objective and the generator, that helps in getting a strong enough discriminator to guide the generator. From the detailed figure (Fig. 4), we observe that IDL is almost only beneficial in the middle regime, for the loss around .2, the IDL models obtain 10% more examples with such a loss.

Finally from the fact that unsupervised IDL outperforms supervised IDL, see Table 2 and Fig. 4, we conclude that the models either already perform at their best (which is unlikely given the artefacts), or that the extra ground-truth data is underused by the current training paradigm. This leaves room for improvement by using a different discriminator in supervised IDL.

Stepwise learning: Control object rotation. Since IterGANs generate the final image in steps, each step could be interpreted as a partial rotation. In this experiment we use the number of iterations k to control the amount of rotation, *i.e.* by varying k we inter- and extrapolate the generator to generate different angles. Note that P2P-30 can only rotate in steps of 30° and that the IG6-MU has no guarantee to rotate for 5° per step, it has just been trained on performing 30° rotations in 6 steps. We include three additional training strategies, each using different target rotations and losses:

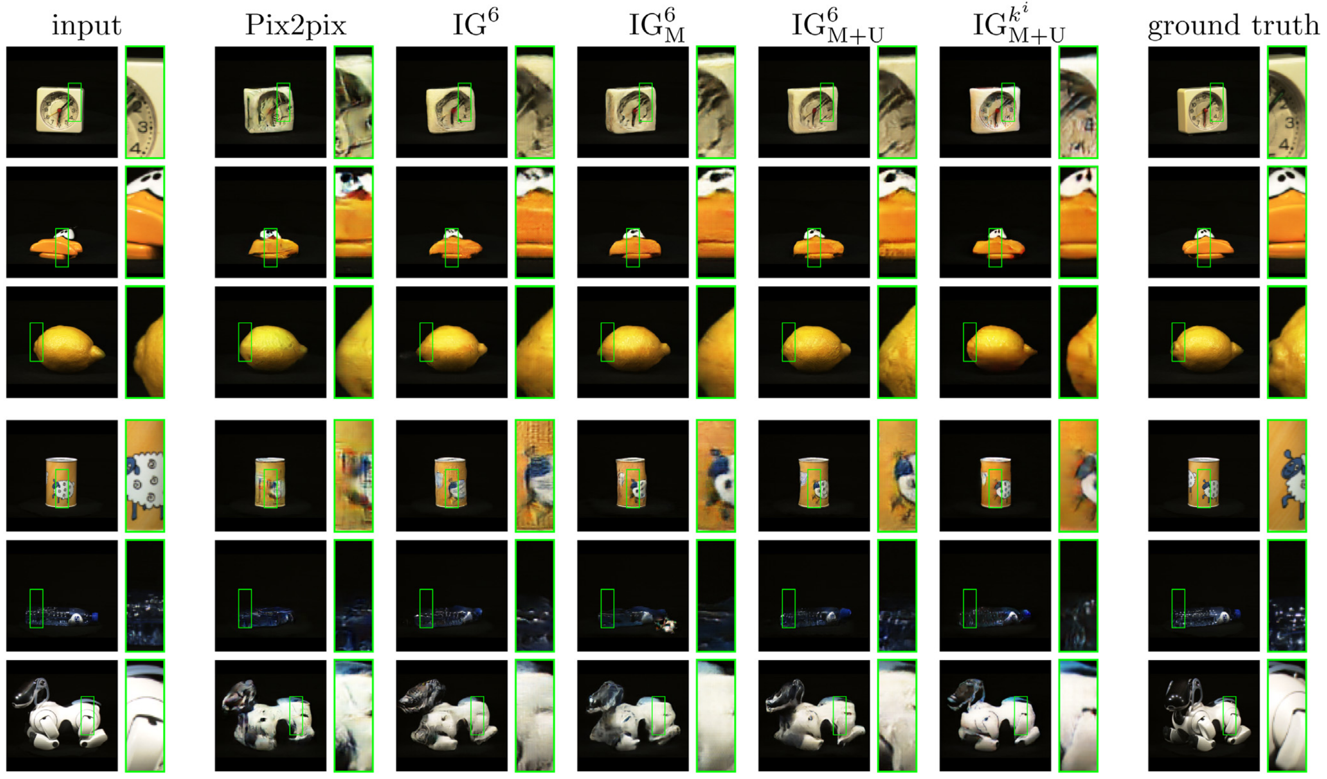

Fig. 5. Qualitative comparison of the models (*columns*). The top three rows show a 30° rotation of seen objects, and the bottom three for unseen objects. The green rectangle is magnified to better compare details. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

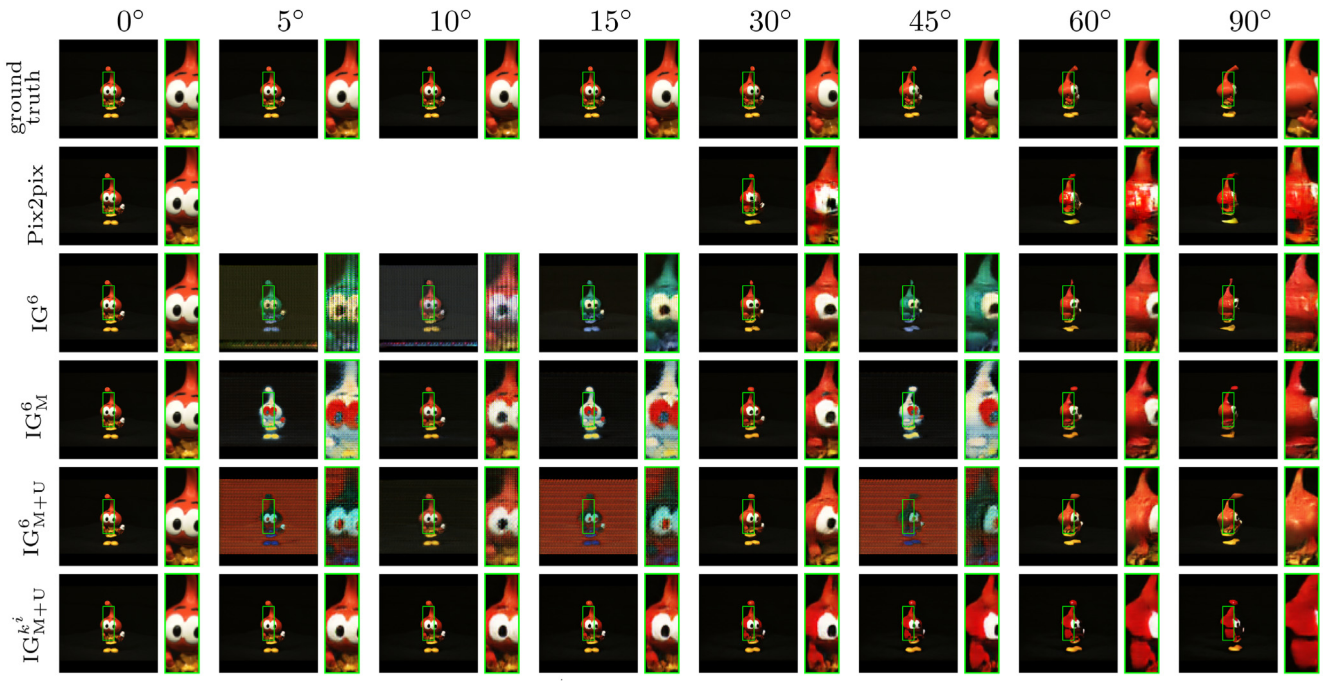

Fig. 6. Inter- and extrapolation of the different models (*rows*) with the ground truth at top. The columns show the rotation from the input. The IterGANs show more realistic generated images for a wide range of angles. *(Best viewed in colour, zoom in for details)*.

1. IGA-MU: sampling k from the full range for each epoch;
2. IGS-M: following the stepwise learning (Section 3.2.1); and
3. IGS-MU the same as above, with the unsupervised IDL.

In Fig. 8 we show the L_1^{M} performance of several models for different angles, thus varying values of target k at evaluation time. We also provide qualitative results for different models, see Fig. 6, and for different objects (both seen and unseen) for IGS-MU, see Fig. 7. From the stepwise learning variants, the graph shows that training with incrementing k (IGS) outperforms training with all values for k from the start (IGA). This indicates that learning small rotations first helps

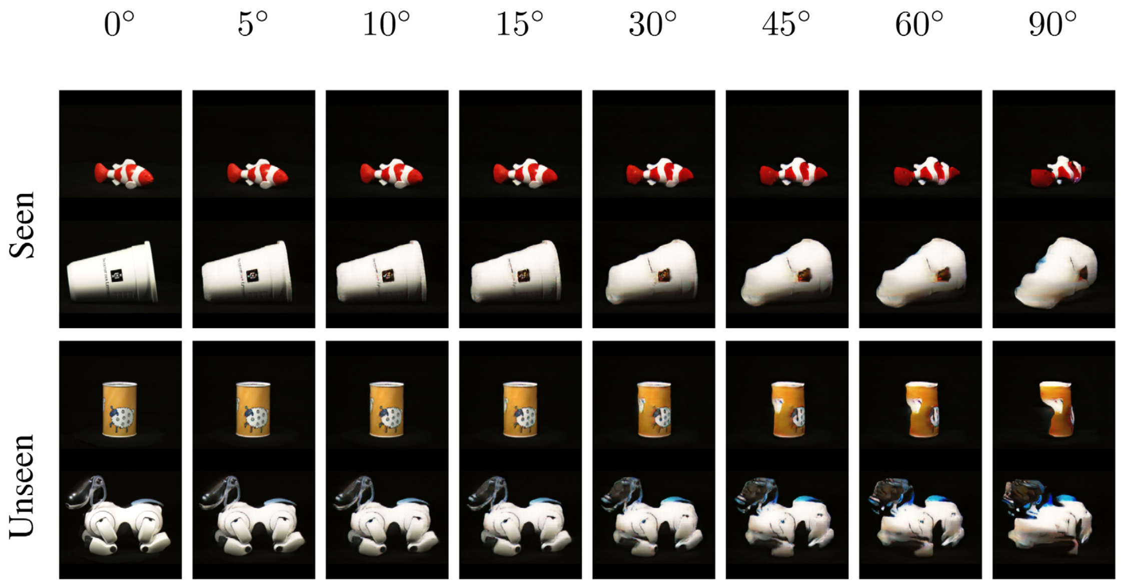

Fig. 7. Illustration of generated rotated objects from IGS-MU for both *seen* and *unseen* objects for different rotation angles. *(Best viewed in colour)*.

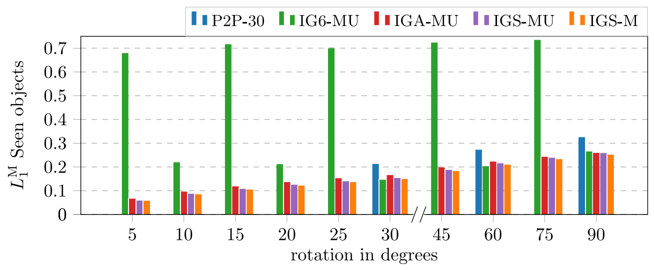

Fig. 8. Showing the performance versus the target rotation. Note that P2P-30 only rotates in steps of 30°, missing most angles. Smaller rotations are easier than large rotations, and iterative training performs best for almost any rotation.

the training process, it also indicates that IDL is of little beneficial value when such an incremental learning approach is used.

Discussion on quality of intermediate images. A clear phenomenon when using IG6 IterGANs are the artefacts introduced in the intermediate generated images, see row 3–5 in Fig. 6, which disappear at the final iteration. The most obvious artefact is the red background added to every other image generated by IG6-MU. This results in a phase-two periodic pattern in Fig. 8, where IG6-MU is either one of the best scoring models (for 30, 60, and 90), or the worst scoring models (*e.g.* for 5, 15, and 25).

This phenomenon shows that even with IDL, it is hard to train the generator to produce both good final images and high quality intermediate images. We have evaluated performance of a 30° rotation as a function of the rotation of the object on the input image, the standard deviation is low (0.012/0.013 on L_1^M). Based on this evaluation we conclude that the colour tint is not caused by the angle of the object on the input image, it is caused by suboptimal parameters in the network We believe this is due to the complex parameter space, where the set of parameters which satisfies both objectives is smaller than the set of parameters which only produce good final images.

We have investigated this by learning multiple times with the same settings and we observed that this pattern already occurs at an early stage, *e.g.* the red background is visible after a few epochs and the model does not escape from this local maximum to produce higher quality intermediate images. One way to overcome these artefacts is by first learning for 5° rotations before propagating generated images for multiple iterations, the strategy we follow in our stepwise approach (IGS).

The discriminator in the unsupervised IDL model has to tell apart real images (A or T) from generated images (B^k), over a large set of objects and colours. For this discriminator an image with different colouring, yet consistent local structure, might look real. Another object can have this colour pattern. This could be partly caused by our optimisation strategy, where for every input and output pair, we sample a *single* intermediate image for IDL. It would be interesting to use all generated images and then enforce a consistency requirement over this sequence.

4.2. Human evaluation

Given that none of the evaluation metrics will be able to properly address image quality, we have performed a (small) user study. Users would see an input image, and the corresponding output images of two models, or the ground-truth. The uses were asked to indicate which output image they found the best of the two considering the input image. We have compared ground-truth, P2P-M30, IG6-M, IG6-MU, and in total 342 image pairs are judged by in total about 10 people.

The results are shown in Table 4, where we sum the scores of the human judges, +1 for the image which is judged as best by the human, and −1 for the other image. When a ground-truth image is shown, it is selected in about 87% of the time (to transform our score average to a selection percentage we use $\frac{1}{2}(1+S)$). Which indicates that in about 13% of the cases the generated image is of such high quality it is selected above the ground-truth image. For the other models, the results are closer together, but indicate that IG6-MU is selected more often than IG6-M, which is selected more often than P2P-M30. This user study confirms our experiments, that the iterative nature of IterGANs helps to generate to more realistic images.

4.3. Unseen objects

In this experiment we use the test set with the *unseen* objects, to compare the generalisation of the different models. The results are shown in Table 3. We observe, as is expected, that the performance on the seen objects is in general better than the unseen objects, and that training for the L_1^M metric does not improve the performance over the unseen objects that much. The models trained with stepwise learning (IGS-M and IGS-MU), however, generalise extremely well to never seen

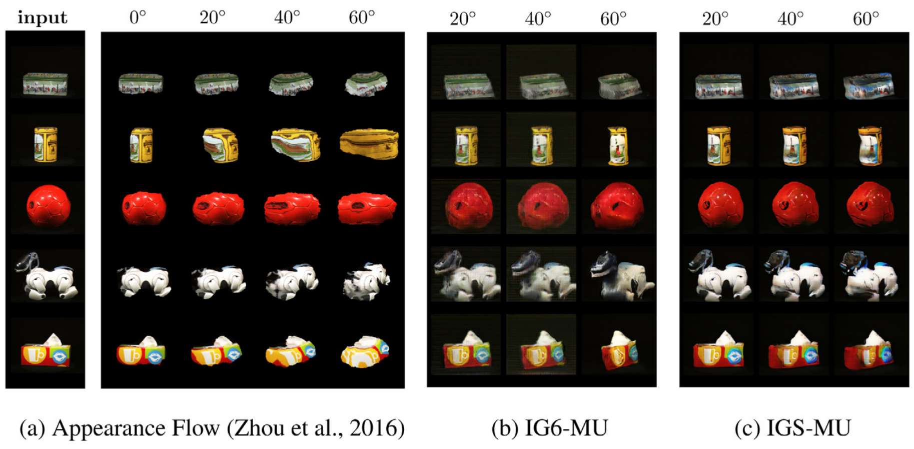

Fig. 9. Illustrative examples of ALOI objects rotated with Appearance Flow (Zhou et al., 2016), compared to our proposed IG6-MU and IGS-MU models. Appearance Flow warps all images into a sedan style shape, already for the 0° rotation, while the colours (not texture) remain realistic. In contrast our models propagate colours less convincingly, yet the original shape is better preserved.

Table 3
Comparison on *seen* and *unseen* test sets. The learning based models lose $0.5-1$ on the L_1^M evaluation metric on the *unseen* data. Surprisingly the stepwise IGS models perform almost equally on the *seen* and *unseen* objects.

	L_1^M seen	L_1^M unseen
Identity	.298 ± .154	.295 ± .139
P2P	.210 ± .092	.256 ± .100
IG6	.200 ± .084	.249 ± .100
IG6-M	.162 ± .060	.252 ± .094
IG6-MU	**.147** ± **.055**	.231 ± .094
IG6-MS	.152 ± .058	.232 ± .092
IGS-M	.155 ± .066	**.167** ± **.073**
IGS-MU	.151 ± .063	**.167** ± **.071**

Table 4
Human evaluation of different models. The human act as judge of a pair of images, and selects one as best. The winning image obtains a +1 score, the other −1. For the average score we summed the scores per model and divided by the number of times the model was shown. The ground-truth image is selected 87% of the time as best, meaning that in the remaining 13% a generated image is deemed more realistic.

	GT	P2P-M30	IG6-M	IG6-MU	*Avg Score*
GT	0	17	18	15	0.74
P2P-M30	−17	0	−7	−3	−0.13
IG6-M	−18	7	0	−14	−0.1
IG6-MU	−15	3	14	0	0.014

Table 5
Comparison of Appearance Flow (Zhou et al., 2016) and the proposed IterGANs on the unseen test set of ALOI. Note that AF is originally trained on a dataset of synthetic cars, while IG6-MU only on rotations of 30°. IterGANs obtain higher performance on any rotation and any evaluation measure.

	L1 (↓)	L1M (↓)	VIFp (↑)	SSIM (↑)
	20° rotation			
AF	.023	.187	.102	.402
IG6-MU	.030	.249	.228	.781
IGS-M	**.012**	**.125**	**.322**	**.834**
	40° rotation			
AF	.030	.218	.202	.392
IG6-MU	.049	.323	**.435**	636
IGS-M	**.015**	**.156**	.302	**.818**
	60° rotation			
AF	.034	.227	.066	.386
IG6-MU	.025	.308	**.390**	**.895**
IGS-M	**.016**	**.179**	.291	.808

objects, performing better than P2P and IG6 on seen data. In Fig. 5 (*bottom three rows*) unseen objects are shown, note how the sheep-shape on the mug is created, while that mug was not seen during training. In conclusion, our models learn depth cues and texture patterns, to rotate even never seen objects.

Comparison to appearance flow. In this section we compare IterGANs to the Appearance Flow (AF) method of Zhou et al. (2016). AF is specifically trained on cars, from CAD renders, disentangling the flow and the appearance of the rotations. We use the pre-trained AF model on our *unseen* test set, see Fig. 9 for some qualitative results and compare to the IGS-M model.

The results are presented in Table 5, where we show the performance of AF and IGS-M for $R = \{20, 40, 60\}$ rotations, following the AF settings. For any of the rotations the IGS model outperforms AF by a large margin, making the explicit control of IterGANs an alternative for the implicit control vector in AF.

In Fig. 10 we show a qualitative comparison to the AF, using only cars from ALOI and their dataset. From the figure we conclude that both methods are specific about the expected transformation/rotation, and have difficulty to generalise well to the other dataset. AF is (logically) more specific about the object, it warps every object into a sedan, but transfer realistic/correct colours. IterGANs, on the other hand, are agnostic towards the kind of object to rotate, introduce some colour changes, and are specific on the turntable setup.

IterGANs for data augmentation. In this section we explore using IterGANs as a means of data augmentation in a classification system. Unfortunately, the ALOI dataset is not suitable for classification, per class it only contains a single object (with photos taken from various viewpoints). We resort to the Office-Home dataset (Venkateswara et al., 2017) with real object centric images, which somewhat resemble the ALOI data used for training IterGANs. We evaluate data augmentation in a domain transfer & classification experiment, where IterGANs are used to generate novel views of images from a novel domain.

The product domain of the dataset contains 4439 images from 65 categories, which we use as follows: for each class, 9 images are used for testing (585 in total), 5 images for validation (325 in total), and the remaining 3529 images for training. The minimum number of example images for a class is 38 (resulting in 24 training images for that class). We have automatically changed the white background for a black background to resemble the ALOI images.

On this dataset we have trained and evaluated the following:

- Train on the original dataset (3529 images); or
- Train on an augmented training set, either *rot3* where IterGANs are used to generate 3 images: 5°, 10°, and 15° rotations, or *rot6*,

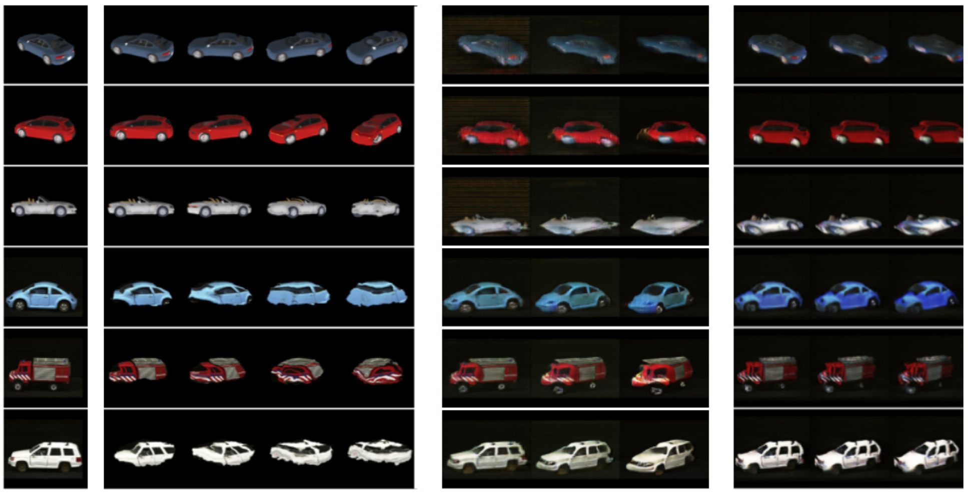

Fig. 10. Illustration of rotated cars from Appearance Flow (Zhou et al., 2016) (*left*) and our proposed IG6-MU (*middle*) and IGS-MU (*right*) models. We show three cars from the AF dataset (*top*) and three from ALOI (*bottom*). Note IG6-MU is only trained on 30° rotations, while the shown rotations are 20/40/60 degree.

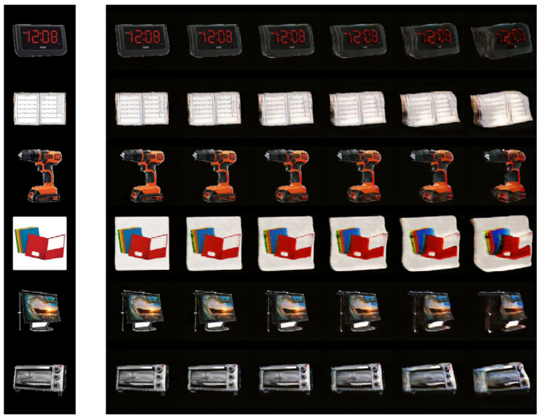

Fig. 11. Examples of product dataset with rotated versions.

Table 6
Classification accuracy for different variants of using IterGANs for data augmentation. Using rotation augmentation is always helpful at test time.

		org	+R5	+R10	+R15	+R20	+R30	+R45
VGG16	org	76.7	***79.0***	77.6	76.9	74.9	73.7	72.7
	rot3	77.4	**78.3**	76.6	75.9	75.4	74.4	73.2
	rot6	**76.2**	**76.2**	**76.2**	76.1	74.5	73.0	73.0
ResNet101	org	70.8	***75.9***	***75.9***	74.5	73.7	71.3	68.6
	rot3	72.5	**74.9**	74.7	74.4	73.2	72.7	71.6
	rot6	66.7	70.9	70.6	71.8	72.3	**72.5**	71.5

where 6 images are generated (also 20°, 30°, and 40°), see Fig. 11 for some illustrative examples.

- Evaluate using the original image only, or an average prediction of the original plus generated rotated versions.

For these experiments we have used fixed image representations (from VGG or ResNet-101) and used the validation set for the learning rate and the number of epochs, using the performance of evaluating the original images only.

The results are shown in Table 6. We observe that using IterGANs for data augmentation can always be beneficial over using only the original data at test and train time. The best performance is obtained when trained on the original data and evaluated using a small rotated version of the object.

Table 7
Comparison of models on VKITTI dataset, using L_1 evaluation metric.

	Seen	Unseen
Pix2pix	0.077 ± 0.017	0.216 ± 0.019
IG6	0.073 ± 0.015	0.209 ± 0.020
IG6-U	0.073 ± 0.015	0.201 ± 0.024

4.4. Camera rotation on VKITTI

In our final experiment, we explore the problem of camera rotation: *generate a scene from a different camera viewpoint*. For this task we use the Virtual-KITTI dataset (Gaidon et al., 2016), and the task is to generate a 30° rotated side-view from the main camera. We train on 4 sequences, using images of 728 × 256 pixels, since the model is fully convolutional, the number of parameters remains the same. This data set is smaller than ALOI, therefore we train for 50 epochs, since the data does not contain object masks, nor intermediate images, we cannot train using the L_1^M objective or the supervised IDL (IG_S) models.

We train using IG6 and IG6-U, and show intermediate generated images in Fig. 12. The IG6 model seem to alternate between adding details and loosing details, it is almost surprising that from this final-last image such a realistic output image is synthesised. The IG6-U model, on the other hand, seems to iteratively add details and rotate the image, yet with a big shift in rotation in the first image and then adding details. The final image has some different colourings than the target image.

In Table 7 we show the quantitative results of three models: P2P, IG6, IG6-U. The performance is rather similar, yet the IG6-U model performs better than the other two models. In Fig. 13 we show qualitative results, of input/output pairs with the generated output images and the closest training example (based on L2 distance). The quality of the generated images on the test parts of the seen sequences are very good, yet on the unseen sequences are not convincing. This might be due to overfitting on the training set, yet the synthesised images are different from the closest train input and looking very realistic.

5. Conclusion & outlook

In this paper we have introduced IterGANs, a GAN model which iteratively transforms an image into a target image whereby the generator has to learn only small transformations. IterGANs are in part inspired on the Shepard and Metzler (1971) mental rotation experiment,[3] which

[3] In Fig. 1 the left objects are not only rotated, but also mirrored.

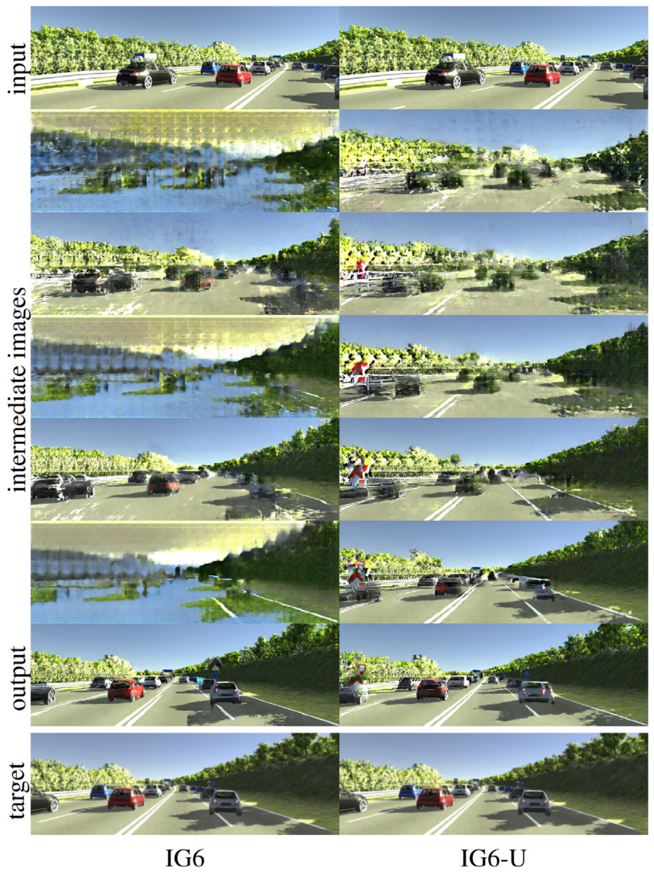

Fig. 12. Illustration of intermediate generated images from the VKITTI dataset. The IG6 model seems to add/remove details, while the IG6-U model iteratively rotates and add details until the final output image.

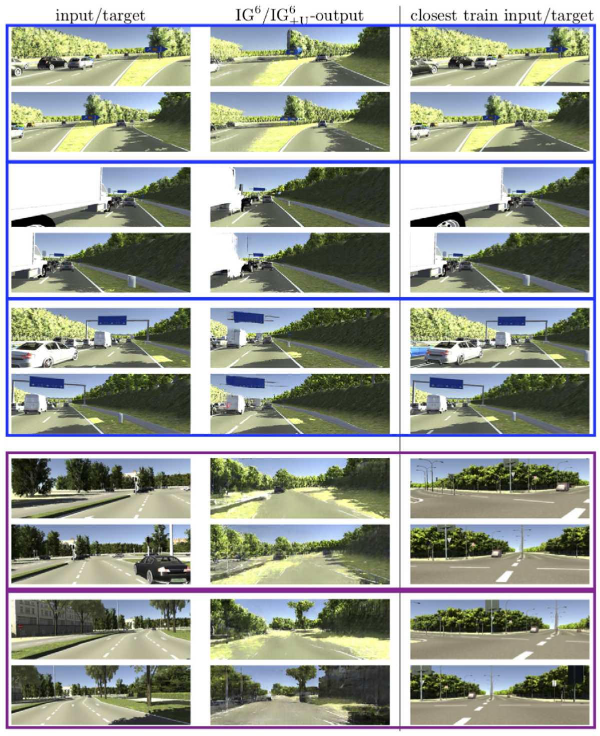

Fig. 13. Qualitative results on VKITTI dataset for test images from seen *(top)* and unseen *(bottom)* sequences. For each showing input/target images, generated results from IG6 and IG6-U, and nearest training images (L_2 distance). The examples from unseen sequences show many artefacts, the difficulty of the problem is highlighted by the dissimilarity with the nearest training images.

indicates that learning small rotations is easier than larger rotations. The iterative nature of IterGANs also allows for additional discriminators in the objective function, either supervised or unsupervised, on the intermediate images. These discriminators help to overcome some artefacts, and lead to better final synthesised images.

Our experiments have shown that IterGANs outperform a direct transformation GAN (P2P model). Surprisingly the unsupervised intermediate discriminator loss is on par with the supervised counterpart, indicating that the additional supervision is not used optimally. A more extensive exploration of possible intermediate loss functions, *e.g*. exploiting the sequence of the generated images rather than sampling a single intermediate image is left for future work.

We have then explored using the additional supervision in a stepwise training approach: in the first few epochs the model learns only small rotations of objects, before learning larger rotations. We have also used the number of iterations in the network as an explicit control signal. This helps to guide the learning process and produce more realistic images, especially for *never seen* objects.

Future research could investigate incorporating the explicit control which is offered by the number of iterations used with IterGANs into an implicit control vector. For example the iterative process could also output a (residual) control vector, and the generated image has the target rotation when the control vector is all zero. While our model only learns right rotations of an object, such a more flexible model could be trained with *cycle awareness* (Zhu et al., 2017b). This exploits the insight that a realistic generated image from an input with a target transformation could also be used as input with the inverse transformation to obtain the input image back. Such an explicit–implicit controlled IterGANs could allow for full 3D transformation of both objects and scenes.

Acknowledgements

We thank all reviewers for the extensive feedback and suggestions for improving our manuscript. This research was supported in part by the NWO VENI What& Where project.

References

Antipov, G., Baccouche, M., Dugelay, J.-L., 2017. Face aging with conditional generative adversarial networks. In: ICIP.
Arjovsky, M., Chintala, S., Bottou, L., 2017. Wasserstein generative adversarial networks. In: ICML.
Bruno, F., Bruno, S., De Sensi, G., Luchi, M.-L., Mancuso, S., Muzzupappa, M., 2010. From 3d reconstruction to virtual reality: A complete methodology for digital archaeological exhibition. J. Cultural Herit. 11 (1), 42–49.
Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: EMNLP.
Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., Choo, J., 2018. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In: CVPR. IEEE.
Denton, E.L., Chintala, S., Fergus, R., et al., 2015. Deep generative image models using a laplacian pyramid of adversarial networks. In: NIPS.
Dosovitskiy, A., Tobias Springenberg, J., Brox, T., 2015. Learning to generate chairs with convolutional neural networks. In: CVPR. IEEE.
Gaidon, A., Wang, Q., Cabon, Y., Vig, E., 2016. Virtual worlds as proxy for multi-object tracking analysis. In: CVPR. IEEE.
Galama, Y., Mensink, T., 2018. Itergans: iterative gans for rotating visual objects. In: ICLR - Workshop.
Geusebroek, J.-M., Burghouts, G.J., Smeulders, A.W., 2005. The amsterdam library of object images. Int. J. Comput. Vis. 61 (1), 103–112.
Gibson, S., Hubbold, R.J., Cook, J., Howard, T.L., 2003. Interactive reconstruction of virtual environments from video sequences. Comput. Graph. 27 (2), 293–301.
Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets. In: NIPS.
Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9, 1735–1780.

Huang, R., Zhang, S., Li, T., He, R., et al., 2017. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In: ICCV.

Isola, P., Zhu, J.-Y., Zhou, T., Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. In: CVPR. IEEE.

Karras, T., Aila, T., Laine, S., Lehtinen, J., 2018. Progressive growing of gans for improved quality, stability, and variation. In: ICLR.

Ko, J., Kim, M., Kim, C., 2007. 2d-to-3d stereoscopic conversion: depth-map estimation in a 2d single-view image. In: Proc. of the SPIE. p. 66962.

Koenderink, J.J., Van Doorn, A.J., 1991. Affine structure from motion. J. Opt. Soc. Amer. A 8 (2), 377–385.

Li, M., Zheng, D., Zhang, R., Yin, J., Tian, X., 2015. Overview of 3d reconstruction methods based on multi-view. In: IHMSC. IEEE.

Liang, X., Zhang, H., Lin, L., Xing, E., 2018. Generative semantic manipulation with mask-contrasting gan. In: ECCV.

Mirza, M., Osindero, S., 2014. Conditional generative adversarial nets, arXiv preprint arXiv:1411.1784.

Ng, J.Y.-H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G., 2015. Beyond short snippets: Deep networks for video classification. In: CVPR. IEEE.

Nistér, D., 2005. Preemptive ransac for live structure and motion estimation. Mach. Vis. Appl. 16 (5), 321–329.

Park, E., Yang, J., Yumer, E., Ceylan, D., Berg, A.C., 2017. Transformation-grounded image generation network for novel 3d view synthesis. In: CVPR. IEEE.

Pilzer, A., Xu, D., Puscas, M.M., Ricci, E., Sebe, N., 2018. Unsupervised adversarial depth estimation using cycled generative networks. In: International Conference on 3D Vision.

Pollefeys, M., Nistér, D., Frahm, J.-M., Akbarzadeh, A., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S.-J., Merrell, P., et al., 2008. Detailed real-time urban 3d reconstruction from video. Int. J. Comput. Vis. 78 (2–3), 143–167.

Radford, A., Metz, L., Chintala, S., 2015. Unsupervised representation learning with deep convolutional generative adversarial networks, arXiv preprint arXiv:1511.06434.

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H., 2016. Generative adversarial text to image synthesis. In: ICML.

Rematas, K., Nguyen, C.H., Ritschel, T., Fritz, M., Tuytelaars, T., 2017. Novel views of objects from a single image. IEEE Trans. Pattern Anal. Mach. Intell. 39 (8), 1576–1590.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X., Chen, X., 2016. Improved techniques for training gans. In: NIPS.

Saxena, A., Sun, M., Ng, A.Y., 2009. Make3d: Learning 3d scene structure from a single still image. IEEE Trans. Pattern Anal. Mach. Intell. 31 (5), 824–840.

Sheikh, H.R., Bovik, A.C., 2006. Image information and visual quality. IEEE Trans. Image Process. 15 (2), 430–444.

Shen, W., Liu, R., 2017. Learning residual images for face attribute manipulation. In: CVPR. IEEE.

Shepard, R., Metzler, J., 1971. Mental rotation of three-dimensional objects. Science 171 (3972), 701–703.

Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E., 2015. Multi-view convolutional neural networks for 3d shape recognition. In: ICCV. IEEE.

Suveg, I., Vosselman, G., 2004. Reconstruction of 3D building models from aerial images. J. Photogramm. Remote Sens. 58 (3), 202–224.

Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S., 2017. Deep hashing network for unsupervised domain adaptation. In: CVPR. IEEE.

Vicente, S., Agapito, L., 2013. Balloon shapes: Reconstructing and deforming objects with volume from images. In: 3DV. IEEE.

Vinyals, O., Toshev, A., Bengio, S., Erhan, D., 2015. Show and tell: A neural image caption generator. In: CVPR. IEEE.

Vishwanath, D., Hibbard, P.B., 2013. Seeing in 3-d with just one eye: Stereopsis without binocular vision. Psychol. Sci. 24 (9), 1673–1685.

Wang, Z., Bovik, A.C., Lu, L., 2002. Why is image quality assessment so difficult?. In: ICASSP. IEEE.

Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., 2004. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. 13 (4), 600–612.

Wang, T., Liu, M., Zhu, J., Tao, A., Kautz, J., Catanzaro, B., 2018. High-resolution image synthesis and semantic manipulation with conditional gans. In: CVPR. IEEE.

Xie, J., Girshick, R., Farhadi, A., 2016. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In: ECCV.

Yan, X., Yang, J., Yumer, E., Guo, Y., Lee, H., 2016. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In: NIPS.

Yi, Z., Zhang, H., Tan, P., Gong, M., 2017. Dualgan: Unsupervised dual learning for image-to-image translation. In: ICCV.

Zhou, T., Tulsiani, S., Sun, W., Malik, J., Efros, A.A., 2016. View synthesis by appearance flow. In: ECCV.

Zhu, J.-Y., Park, T., Isola, P., Efros, A., 2017a. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV.

Zhu, J.-Y., Park, T., Isola, P., Efros, A.A., 2017b. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV. IEEE.