A Survey of Image Synthesis Methods for Visual Machine Learning

A. Tsirikoglou, G. Eilertsen and J. Unger

Department of Science and Technology, Linköping University, Norrköping, Sweden
{apostolia.tsirikoglou, gabriel.eilertsen, jonas.unger}@liu.se

Abstract

Image synthesis designed for machine learning applications provides the means to efficiently generate large quantities of training data while controlling the generation process to provide the best distribution and content variety. With the demands of deep learning applications, synthetic data have the potential of becoming a vital component in the training pipeline. Over the last decade, a wide variety of training data generation methods has been demonstrated. The potential of future development calls to bring these together for comparison and categorization. This survey provides a comprehensive list of the existing image synthesis methods for visual machine learning. These are categorized in the context of image generation, using a taxonomy based on modelling and rendering, while a classification is also made concerning the computer vision applications they are used. We focus on the computer graphics aspects of the methods, to promote future image generation for machine learning. Finally, each method is assessed in terms of quality and reported performance, providing a hint on its expected learning potential. The report serves as a comprehensive reference, targeting both groups of the applications and data development sides. A list of all methods and papers reviewed herein can be found at https://computergraphics.on.liu.se/image_synthesis_methods_for_visual_machine_learning/.

Keywords: methods and applications

ACM CCS: [Computing Methodologies]: Computer graphics; Machine learning; Rendering

1. Introduction

We are currently witnessing a strong trend in the use of machine learning (ML), particularly through deep learning (DL) [LBH15, GBC16]. Many areas of computer science are now considering DL as an integral part for advancing the state of the art, from recommender systems [ZYST19] and medical diagnosis [LKB*17], to natural language processing [YHPC18] and computer vision [VDDP18, SSM*16]. The techniques used in DL, and the overall computational resources are fastly evolving. However, today the bottleneck is often caused by the limited availability and quality of training data [RHW18]. No matter the potential of a particular model and the computational resources available for training it, the end performance will suffer if the training data cannot properly represent the distribution of data that is supposed to be covered by the model.

Data acquisition is a limiting factor, not only due to the actual capturing process, but most often because annotations for supervised learning can be expensive and prohibitively time-consuming to generate. Moreover, it is difficult to cover all possible situations that are relevant. These problems have made it crucial to make the most of the available training data, and augmentation techniques for various purposes (generalization, domain adaptation, adversarial robustness, regularization, etc.) is today an essential step in the DL pipeline [SK19, PW17]. While data augmentation can be thought of as a synthetic data generation process, the synthesized samples are bound by the data at hand. Therefore, it is becoming increasingly popular to generate data in a purely synthetic fashion.

The demands for large quantities of data are especially important in DL as compared to classical ML, meaning that data generation techniques tailored for this purpose have mainly appeared within the last decade, as illustrated in Figures 1 and 6. Synthetic data for DL also open up for interesting improvements on how training data should be formulated. For example, it could be advantageous to oversample difficult examples, instead of reflecting the exact intended real distribution. Also, the data samples themselves could be shaped in an unrealistic fashion in order to promote efficient optimization, e.g. as in techniques for domain randomization [TFR*17, TPA*18].

With the rapid progress both in DL, and in methods that leverage the synthetically generated data, we recognize the need for a comprehensive overview of the existing methods of synthesizing training data. In this paper, we present a survey of image synthesis
techniques for visual ML. Visual ML concerns tasks connected to visual perception, and includes algorithms that utilize visual training data. To a great extent, in practice, visual ML attempts to solve computer vision tasks, from low-level ones like visual odometry and optical flow to higher-level tasks like 3D scene layout, semantic understanding and object detection and tracking.

Our focus is on image data, and the computer graphics aspects of the image generation process. In computer graphics, image synthesis, or rendering, is used to transform a geometric scene description into an image by simulating or approximating the lighting interactions in the scene and onto the sensor of a virtual camera device [Kaj86, PJH16]. We consider image synthesis methods that can be used to create a stand-alone training data set for ML optimization. In these, we also include hybrid techniques that are not based on physically based rendering. Importantly, this formulation is restricted to techniques that intend to create an entire training set and therefore does not include methods specific to, e.g. data augmentation or synthetic image generation only for testing purposes.

To distinguish between the image synthesis methods used in ML, we provide a taxonomy that considers the image generation perspective. This focuses on the employed computer graphics techniques, where we consider both differences in scene modelling as well as rendering. Moreover, we also discuss around the different methods in relation to the specific computer vision problem that the generated synthetic data were formulated to solve. This means that each method we discuss is categorized in the provided image generation taxonomy, and listed according to its intended application within computer vision.

Providing a structure and categorization of what has been done so far can promote future work on synthetic training data generation. In the future, synthetic data will become increasingly more important. Some of the most challenging remaining problems in DL can be tied to data, such as adversarial examples, data set bias and domain adaptation/generalization. Synthetic data have the potential to be a central component in solving such problems. In addition, there are many issues left to solve in the training data synthesis pipeline itself. For example, bridging the domain gap between generated images and the reality, where the current synthetic data sets often are used in combination with real data to achieve the best performance. Also, it is still an open question how to generate scene descriptions in an optimal sense for solving the specific task at hand.

In summary, this report provides the following contributions:

- A background on image formation and rendering, including both classical and recent learning-based image synthesis techniques (Section 2).
- A taxonomy of image synthesis methods for visual ML, from the image generation perspective, as well as a brief analysis of the active computer vision applications that benefit from synthetic training data (Sections 3 and 4).
- A survey and overview of the existing methods for synthetic image generation for visual ML, focusing on the image generation aspects (Section 5).
- A brief qualitative evaluation of the methods, focusing on the data complexity and performance aspects when using the synthesized data for visual ML (Section 6).
- A discussion around the current situation in ML employing synthesized training data, and the main challenges and opportunities for future work (Section 7).

2. Background

This section provides a background on image formation and synthetic image generation for ML. We start from a short historical overview of image synthesis and its introduction in the ML community, followed by a description of the techniques involved in scene modelling and image synthesis. Finally, the recent developments in learning-based generative modelling, and how image synthesis relates to methods for data augmentation, are also described.

2.1. Historical overview

Although computer-generated image content has been a concept since the mid 20th century, it was in the mid 1970s that research on this topic gained momentum, e.g. with the work on fundamental concepts such as shading [Gou71, Bli77], bump mapping [Bli78] and ray tracing 1979. In the 1980s, computer graphics research was spurred by the interest in computer games, and then from the movie industry in the 1990s. The interest has since expanded to a diverse set of applications, including advertisements, medicine, virtual reality, science and engineering.

Pre-requisites for ML, on the other hand, can be dated back to the 18-19th centuries when Bayes’ Theorem [Bay63] and least squares [Leg05] were introduced. Other important early concepts were the Markov chains by Andrey Markov in the early 20th century and thinking machines by Turing [Whi79]. The first steps towards DL and neural networks were conceptualized already in the 1940s, and the Perceptron was introduced in the 1950s by Rosenblatt [Ros58]. This can be thought of as the first wave in DL [GBC16]. In the 1980s, a second wave introduced many concepts fundamental to DL, such as Recurrent Neural Networks [Hop82], back-propagation [RHW*88], Reinforcement Learning [Wat89] and Convolutional Neural Networks [Fuk80, LBD*89]. The third and current wave in DL started in the early 2010s, where the techniques and results presented by Krizhevsky
et al. [KSH12] to many mark the start of the deep learning revolution. Since then, there has been a rapid development in important techniques for training increasingly deeper networks, such as dropout [SHK*14], batch normalization [IS15] and residual connections [HZRS16], on increasingly more difficult problems.

The intersection of computer graphics generated images and computer vision can be traced back to the 1980s, when algorithms for optical flow demanded ground truth annotations for evaluation [Hee87, BFB94, MNCG01]. Since the flow vectors are close to impossible to annotate by hand (a 2D vector is needed for each pixel), and custom scene setups are required to provide the flow vectors in real data [BSL*11], optical flow has been one of the areas within computer vision that has been most dependent on synthetic data. However, it was not until Baker et al. introduced the Middlebury dataset [BSL*11] in 2011, that a separate training set for learning optical flow was made available. Since then, with the recent development in DL, a significant increase in the number of training images was required, and creative solutions have been applied for image generation in large quantities, such as pasting simple 3D objects on background images [DFI*15, MIH*16].

During the last 5 years, semantic segmentation has been receiving a great deal of attention in DL for computer vision [LSD15, CPK*17, CZP*18]. This stems from DL’s efficiency to solve this task in complex scenes, as well as from the fact that semantic segmentation is one of the central computer vision problems within autonomous driving. Although semantic segmentation annotations are possible to create manually, they are time-consuming (e.g. more than 1.5 h per image for the Cityscapes data set [COR*16]). Together, this makes semantic segmentation one of the most popular application areas for learning from synthetic images, and many methods have been proposed (see Table 1 and Section 5.5).

Even though semantic segmentation and optical flow are popular tasks for synthesized image content, there are also many examples of other applications. For instance, one of the first cases of training on synthetic data and testing on real data was demonstrated for pedestrian detection [MVGL10]. Other early examples include pose estimation and object detection [PJA*12]. Figure 6 demonstrates how synthetic data have been introduced in different areas of computer vision, showing the number of methods and data sets presented each year.

Finally, Figure 1 illustrates the recent trend in using synthetic data within ML. Although ML has seen a close to exponential increase in the number of publications over the last decade, there is an ongoing trend on synthetic data within ML. This is reflected in Figure 1 by an increasing fraction of publications connected to synthetic data, i.e. when the number of publications is normalized by the total number of ML papers. However, it should be noted that this only shows the general trend; there could be papers that fit the search criteria, but they do not provide synthetic training data for ML, and there could be papers that treat synthetic training data but do not use the term synthetic data in the paper title or abstract.

2.2. Visual data generation

The pipeline for visual data generation can be divided into two main parts; content/scene generation and rendering. By content generation, we mean the process of generating the features that build up the virtual environment in which the sensors are simulated. By rendering, we mean the process of simulating the light transport in the environment and how cameras and other sensors, e.g. LIDAR or radar, captures/measure the virtual world. Data for training or testing of ML algorithms should meet the following requirements:

- Feature variation and coverage—The features included in the generated data should be diverse while covering the domain of possible features densely enough to be representative for the application domain and the feature distributions in real data.
- Domain realism—The simulated sensor data, e.g. images, should be generated in a way such that the domain shift to the real counterpart is minimized either directly in the synthesis or by applying some domain transfer model.
- Annotation and meta-data—One of the key components of synthetic data is that annotations and meta-data can be generated automatically and with high quality.
- Scalable data generation—In most, if not all, applications large amounts of data points with annotations are required. It is, therefore, necessary that the data generation process scales easily in both content generation as well as in sensor simulation.

Below we give an overview of the principles behind content creation and image synthesis in the context of visual data for ML. It should be noted that very similar principles can be generalized and extended to cover also other types of sensors.

Content generation is the process of generating the virtual world, the objects and the environments in which the sensors are simulated. Depending on the application and approach, the content generation may range from simplistic objects to fully featured photorealistic virtual worlds and include aspects such as geometries, materials, light sources, optics, sensors and other features that build up the world.

Many data generation methods rely on building entire virtual worlds where the simulated sensors move around to capture a variety of images, videos or other measurements. This approach is typically employed when 3D development platforms, such as the Unreal Engine [UE4] from Epic Games or Unity [UNT], are used for data generation [VRK16, RHK17]. The virtual world can be configured using a wealth of tools ranging from geometry and material modelling software packages to animation tools and scripting frameworks. However, most frameworks for modelling and scripting rely on significant manual efforts and artistic work. Although a fixed virtual world may be very large and include dynamic animated objects, the variation or diversity, in the resulting data is limited since the number of possible images that can be generated is built upon a finite set of features.

An alternative method to a single virtual world where a virtual camera moves around is to generate the content on-demand and generate only what is currently in the view of the virtual sensors. Such on-demand content generation can be achieved using procedural methods [TKUW17, WU18] (Figure 2). This approach is more time-consuming per frame than reusing the same geometric structure for multiple images/measurements. However, it is enabled in practice by generating only the set of geometry, materials and lighting environments that are visible to the camera either directly,
or through reflections and shadows cast. In procedural content generation, the construction of the virtual world is defined by a set of parameters and a rule set that translates the parameter values into the concrete scene definition. This means that each scene configuration can be thought of as a sampling of the generating parameter space and that it is possible to shape the virtual world(s) by associating each parameter to a statistical distribution. The scene construction is often a mixture of fully procedural objects and objects from model libraries. For example, in a synthetic data set designed for street scene parsing [WU18] the buildings, road surface, sidewalks, traffic lights and poles could be procedurally generated and individually unique, while pedestrians, bicyclists, cars and traffic signs, could be fetched from model libraries. Even though the geometry is shared between all model library instances, large variability can be achieved by varying properties such as placement, orientation and certain texture and material aspects between instances.

Image synthesis, or rendering, can be performed in many different ways with different approximations affecting the trade-off between computational complexity and accuracy. The transfer of light from light sources to the camera, via surfaces and participating media in the scene, see Figure 3, is described by light transport theory. For surfaces, the light transport is often described using the geometric-optics model defined by the rendering equation [Kaj86], expressing the outgoing radiance \( L(\vec{x} \rightarrow \omega_o) \) from a point \( \vec{x} \) in direction \( \omega_o \) as

\[
L(\vec{x} \rightarrow \omega_o) = L_o(\vec{x} \rightarrow \omega_o) + \int_{\Omega} L(\vec{x} \leftarrow \omega_i) p(\vec{x}, \omega_i, \omega_o) (\vec{n} \cdot \omega_o) d\omega_i,
\]

where \( L(\vec{x} \leftarrow \omega_i) \) is the incident radiance arriving at the point \( \vec{x} \) from direction \( \omega_i \), \( L_o(\vec{x} \rightarrow \omega_o) \) is the radiance emitted from the surface, \( L(\vec{x} \rightarrow \omega_o) \) is the reflected radiance, \( p(\vec{x}, \omega_i, \omega_o) \) is the bidirectional reflectance distribution function (BRDF) describing the reflectance between incident and outgoing directions [Nic65], \( \Omega \) is the visible hemisphere and \( \vec{n} \) is the surface normal at point \( \vec{x} \).

Image synthesis is carried out by estimating the steady-state equilibrium of Equation (1), which represents how the radiance emanating from the light sources scatters at surfaces and participating media in the scene, and finally reaches the camera. This requires solving the rendering equation for a large number of sample points in the image plane, i.e. a set of potentially millions of interdependent, high-dimensional analytically intractable integral equations. Solving the rendering equation is challenging since the radiance \( L \), which we are solving for, also appears inside the integral expression. The reason is that the outgoing radiance from any point affects the incident radiance at every other point in the scene. As a result, the rendering equation becomes a very large system of nested integrals. In practice, the rendering problem can be solved using numerical integration. This can be carried out in several ways with different approximations and light transport modelling techniques.

Rendering and sensor simulation can broadly be divided into two main classes of techniques, rasterization and ray/path tracing. Both rasterization, with further pixel processing, which is the method generally used in graphics processing unit (GPU) rendering, and path tracing, in which Equation (1) is commonly solved using Monte Carlo integration techniques, aim to solve the rendering problem using different approximations. Following the literature, this categorization can also be expressed as a separation between approaches using computer games engines (rasterization) and approaches using offline rendering (ray/path tracing) primarily developed for photorealistic image synthesis and applications in VFX and film production. Rasterization techniques are generally significantly faster as they allow for GPU rendering and fast generation of time-sequential images in the same environment. They are however limited, as significant pre-computations are usually necessary to achieve realistic scene representations and accurate sensor simulation. Path-tracing techniques are extremely general and can accurately simulate any type of light transport and a wide range of sensors without pre-computations. This generality is in many data generation systems a very important aspect, due to the fact that it is often necessary to generate large numbers, sometimes up to hundreds of thousands or more, of diverse images/samples. In many cases the need for diversity, i.e. variation in the content, makes pre-computation unfeasible as it does not scale efficiently.

Rasterization is the technique used by most computer game engines to display 3D objects and scenes on a 2D screen. The 3D objects in the scene are represented using a mesh of polygons, e.g. triangles or quadrilaterals. The polygons describing the 3D models are then traversed on the screen and converted into pixels that can be assigned a colour. By associating normals, textures and colours to each of the polygons, complex materials and lighting effects can be simulated. Using pre-computations to compute complex scattering effects or global illumination (light rays have bounced two or more times at surfaces in the scene) realistic rendering results can be achieved. Although each image can be rendered at high frame rates,
the drawback of rasterization techniques is that the pre-computation required to achieve realistic approximations of Equation (1) is computationally complex and leads to problems in scalability.

Path tracing can simulate any type of lighting effects including multiple light bounces and combinations between complex geometries and material scattering behaviours. Another benefit is that it is possible to sample the scene being rendered over both the spatial and temporal dimensions. For example, by generating several path samples per pixel in the virtual film plane it is possible to simulate the area sampling over the extent of each pixel on a real camera sensor, which in practice leads to efficient anti-aliasing in the image and enables simulation of the point spread function (PSF) introduced by the optical system. By distributing the path samples over time by transforming (e.g. animating) the virtual camera and/or objects in the scene, it is straightforward to accurately simulate motion blur, which in many cases is a highly important feature of the generated data. Path tracing is a standard tool in film production and is implemented in many rendering engines. The drawback of path-tracing techniques is that the computational complexity inherent to solving the light transport in a scene is very high. However, path-tracing algorithms parallelize extremely well even over multiple computers. For large-scale image production, it is common practice to employ high-performance compute (HPC) data centres and cloud services, [WU18]. For an in-depth introduction to path tracing and the plethora of techniques for reducing the computational complexity such as Monte Carlo importance sampling and efficient data structures for accelerating the geometric computations, we refer the reader to the textbook by Pharr et al. [PJH16].

2.3. Learning-based image synthesis

With the introduction of DL for generative modelling, today there is also a category of image generation methods that cannot be placed in the classical image synthesis pipeline, including Variational Autoencoders (VAEs) [KW13] and Generative Adversarial Networks (GANs) [GPAM*14]. Rather, these operate directly in pixel space, producing a complete image as the output of a neural network. Unsupervised learning with GANs uses the concept of a discriminator \( D(x) \), which is trained to determine if the sample \( x \) is from the true distribution \( p_{data}(x) \). The data-generating model \( G(z) \) takes values from a latent distribution \( p_{z}(z) \) (usually a uniform or normal distribution) and is trained to fool the discriminator \( D \), i.e. to maximize the output of \( D(G(z)) \). The loss of \( D \) can be formulated as

\[
\mathcal{L}_D = - \log(D(x)) - \log(1 - D(G(z))),
\]

with the objective to separate between real and synthetic data samples. The generator is trained with the opposite objective,

\[
\mathcal{L}_G = - \log(D(G(z))).
\]

During the optimization process, the two loss functions are iterated, so that the optimization is formulated to play the adversarial min-max game,

\[
\min_{G} \max_{D} \mathbb{E}_{x \sim p_{data}(x)} \left[ \log D(x) \right] + \mathbb{E}_{z \sim p_{z}(z)} \left[ \log(1 - D(G(z))) \right].
\]

with value function \( V(D, G) \). This means that when the generator \( G \) improves the generation of samples, the discriminator \( D \) is also improved to better use features that can be recognized as fake. The generator has to learn more authentic features, and this process is repeated until an equilibrium is reached. The training forces the distribution \( p_{G(z)} \) to be similar to \( p_{data} \). However, the minimax game is also sensitive and prone to fail in the original GAN formulation. For example, if one model takes over, the gradients for optimization can vanish, leaving the stochastic gradient descent (SGD) ineffective. Another problem is mode collapse, where the generator focuses on a few modes and cannot represent the diversity contained in \( p_{data} \). For these reasons, there has been a large body of work devoted to improving the quality and stability of GANs, including the DC-GAN formulation [RMC15] for CNN generator and discriminator, and Wasserstein GANs for more robust optimization [ACB17]. Today, the state-of-the-art GANs usually combine different concepts to achieve stable training on high-resolution data and can produce very convincing synthetic images [KALL18, BDS19, KLA19].

While GANs are very effective, automatic generative models, the original setting offers little control over the generated content. There have been attempts for increasing control by, e.g. disentangling the dimensions of the latent space [CDH*16], controlling individual neurons of the generator which correspond to interpretable
concepts [BZS*19] or disentangling generation at different levels of the generated attributes [KLA19].

For increasing control in image generation, it is also possible to use hybrid techniques that mix GANs and classical methods for scene generation. For example, a scene model can be created and represented without materials, textures and lighting information. In this case, the GAN takes the role of the renderer and transforms the representation to its final form. For this purpose, the GAN should transform an image from one domain to another, a task that can be performed with image-to-image mapping GANs, either in a supervised [IZZE17, WLZ*18] or an unsupervised fashion [ZPIE17].

### 2.4. Data augmentation

As stated earlier, this survey focuses on presenting image synthesis methods that generate entire training data sets for ML applications. Even though out of its scope, it could not completely skip mentioning methods for data augmentation. These are used to augment the training set by transforming the samples according to some predefined or learned rules. Augmentation is one of the most well-used strategies in DL for alleviating overfitting, especially on small data sets. Overfitting occurs when the model is complex enough to memorize the individual samples of the training data, which is often the case with deep neural networks. Increasing the diversity of training images can effectively alleviate this problem. However, augmentation can also have other purposes, such as improved domain adaptation capabilities, or increased robustness to adversarial examples.

Although classical augmentation techniques perhaps do not qualify as image synthesis, there is a whole spectrum of methods with different complexities. Thus, augmentation is closely connected to image synthesis for ML and deserves some explanation. To provide a very brief summary of augmentation techniques, the following paragraphs attempt to cover the majority of the most common, and some of the more unconventional, strategies for augmentation of image data:

- **Simple transformations** cover the great majority of operations that are currently used in practice in a typical DL pipeline. These include geometric transformations, such as rotation, translation, shearing, flipping and cropping, as well as colour and intensity transformations such as changes in contrast, intensity, colour saturation and hue. Simple operations of more local nature could also be included, e.g. blurring and adding noise.

- **Complex transformations** include more sophisticated algorithms for altering an image. One strategy is, for example, to use neural style transfer [GEB16] for increasing the diversity of images.

- **Learning-based methods** are designed to attempt deriving an optimal augmentation policy given a particular model and task [CZM*18, ZCG*19]. For example, AutoAugment uses reinforcement learning to find the best augmentation given a search space of different transformations [CZM*18]. Another type of learning-based image generation strategy is to make use of GANs. However, these are learned to approximate the training data distribution rather than optimizing the end performance of a particular model. For augmentation, GANs can be effective as they can produce faithful images with different scene composition as compared to the original training images [ASE17, BCG*18]. One area where GAN augmentation is becoming popular is within medical imaging [FADK*18, STR*18], where large quantities of training data can be difficult to capture.

- **External resources** can also be an option for augmentation. Although the great majority of augmentation techniques use the training data to formulate additional training samples, it is also possible to use other resources. For example, Peng et al. [PSAS15] make use of a database of CAD models for generating synthetic images for augmentation, and particularly aimed at few-shot learning.

- **Mixup** is more of a training strategy for classification problems, where interpolation between randomly selected samples and their corresponding labels improves stability [ZCDLP17]. Interpolation between different classes makes the behaviour of
a neural network closer to linear in-between the classes, and thus less sensitive to differences in input. However, mixup is also closely connected to augmentation, and the technique has been explored in a variety of settings [SD19, Ino18].

For a thorough overview of data augmentation techniques, we refer to the recent survey of Shorten and Khoshgoftaar [SK19]. In our survey, we do not include synthesis methods for augmentation but focus only on the methods that have been used to provide self-contained training data.

3. Taxonomy of Training Data Generation Methods Based on Image Synthesis Pipeline

We view synthetic data sets and data generation methods from a computer graphics perspective, and make use of the principles from the image synthesis pipeline in Section 2.2. Figure 4 provides a categorization that projects the underlying synthesis technique used by each of the included methods. For the sake of simplicity, we divide the image synthesis pipeline into two major consecutive steps:

1. **Modelling** is the process of developing a representation of all aspects related to the scene content, ranging from the configuration of 3D object models to surface textures and light sources.

2. **Rendering** is ‘the process of producing an image from the description of a 3D scene’ [PJH16]. As described in Section 2.2 rendering is a computational technique that attempts to simulate, in various levels of accuracy, the principles of physics describing the interaction of light and matter from a defined viewpoint.

3.1. Modelling

When it comes to modelling, all the constituting elements of a scene can be either procedurally or non-procedurally generated.

- **Procedural modelling** is the use of algorithms and mathematical functions to determine, for example, the layout of a scene, the shape of an object or the colour and pattern of textures [EMP*02, FSL*15]. In essence, procedural modelling is about parametrizing the generation process and can be applied to any factor of scene content specification, providing a high level of control, flexibility and variability.

- Besides being procedural or not, the scene specification generation can be further categorized according to how the attributes of the scene are modelled:

- **Data-driven modelling** focuses on developing statistical models based on high-quality sensor data and measurement procedures. The acquired data can be, for example, fitted to parametric models or fed to learning-based non-parametric methods. In this way, reliable approximations of real-world elements can be generated; elements for which there are no well-defined mathematical representations, or where a physically based model cannot be used due to limitations of the computer graphics generation system. For example, a data-driven learning-based approach to learn a statistical model for the shape of an elephant body, could use and train on 3D laser scans taken from several subjects (Figure 5, left). In the context of image synthesis, data-driven modelling has been widely used for human face and body shape modelling and material representations.

- **Physically based modelling** is generally defined by certain and well-established processes, usually described mathematically from laws of physics. In our context, we expand the term also to include hand-crafted models that only visually follow the laws that govern our physical world, i.e. which do not make us of underlying scientific formulations and rely on perceptual similarity. To make the point clear, consider the following example: a human-made 3D scene of an elephant next to a cat that is ~10 times shorter could be classified as physically based modelling (Figure 5, middle), while the opposite could not; from visual inspection this violates the rules of proportion.

- **Non-physically based modelling** includes everything that cannot lie in physically based modelling, meaning scene content that does not follow any physical rule accurately or approximately. The models are usually developed either according to some random scheme, or following an abstract or fictional concept (Figure 5, right).

3.2. Rendering

Moving to the rendering step in the image generation pipeline, we categorize according to three main directions for image synthesis:

- **Real-time rendering** based approaches either acquire existing data from real-time game environments or directly generate the images employing real-time visual simulators. A game engine is the software package that incorporates all the necessary elements to build a computer game, i.e. support for importing 2D and 3D assets, a physics engine to define the behaviour and relations between them, special effects and animation support and a rendering engine to create the final visual result. In addition, modern game engines include sound, and an extended set of features for games solution development, like multi-threading and networking. Over the past few years, game engines are more accessible, and able to render...
Table 1: Computer vision applications that benefit from synthetic training data, along with associated image synthesis approaches (superscripts denote cross-application approach).

| Feature-based alignment | Pose estimation | [SQLG15], [TFR*17], [TTB18]^{18} |
|-------------------------|-----------------|----------------------------------|
| Human body              | [SFC*11], [PJA*12]^{11}, [PR15], [CWL*16], [VRM*17]^{18} | [ZB17], [FLC*18]^{16}, [TPAB19] |
| Motion estimation       | Optical flow    | [BSL*11], [BWSB12]^{14}, [HU13]^{22}, [DFI*15], [MIH*16]^{5} |
|                         |                 | [GWCV16]^{6}, [IMS*17], [RHK17]^{7}, [VRM*17]^{8} |
| Stereo correspondence   | Disparity estimation | [BWSB12]^{14}, [MIH*16]^{5}, [ZQC*18] |
|                         | Depth estimation | [BWSB12]^{14}, [HU13]^{22}, [ASS16]^{3}, [GWCV16]^{4}, [VRM*17]^{8} |
|                         | Scene flow estimation | [MG15], [MIH*16]^{5} |
| Recognition             | Semantic segmentation | [HU13]^{22}, [ASS16]^{3}, [RSM*16], [GWCV16]^{4}, [RVRK16], [HPSC16], [HPB*16] |
|                         | Instance segmentation | [GWCV16]^{6}, [MLD17]^{15}, [HJSE*17]^{61}, [RHK17]^{7}, [VRM*17]^{8}, [CGM*17], [STR*18] |
|                         | Object detection | [MVGL10], [PSA*12]^{11}, [GWCV16]^{4}, [JRB*16], [MLD17]^{15}, [QZZ*17], [TTB18]^{18} |
|                         | Classification | [TF18], [KPL*19]^{10} |
|                         | Point-cloud segmentation | [YWS*18] |
|                         | Face recognition | [WHHBO4], [ASN*17], [KSG*18], [TMH18], [GBKK18] |
| Structure from motion   | Visual odometry | [RHK17]^{7} |
| Computational           | Camera design   | [BFL*18], [LSZ*19], [LLFW19] |
| photography &           | Noise modelling | [LMH*18], [ABB19], [BHLH19] |
| Image formation         | Intrinsic decomposition | [BSvdW*13], [RRF*16], [LS18], [BGD*18]^{17}, [BLG18], [WL19], [BSP*19], [SBV20] |

realistic images, making them a capable source of synthetic data for several vision applications. Synthetic data have been extracted from existing commercial games—utilizing open source modifications (in short mods) that can change or tweak in small or large extent one or more aspects of the game, or their generation pipeline has been developed directly on a game development platform.

Extracting data from commercial video games without accessing the game’s source code or content heavily relies on understanding how the rendering pipeline is integrated into the GPU and building dedicated software solutions for their communication. Another prominent method for synthetic data production builds upon virtual world generators created with 3D development platforms such as Unity [UNT] or Unreal Engine 4 (UE4) [UE4]. Initially, these platforms were mainly aimed for game development, but nowadays they offer tools for cinematics, automotive and engineering applications.

Both computer games and 3D development platforms are based on rasterization rendering in conjunction with precomputed lighting to achieve high frame rate. Modern platforms have started to provide real-time ray-tracing effects and capabilities. These implement hybrid techniques that combine simple sampling schemes with denoising algorithms and require specialized graphics hardware to run.

Lastly, other kinds of simulators, such as driving simulators and physics engines [TET12, VDF, DRC*17, MCL*18, SDLK18, MAO*19], have been another source for collecting synthetic data for visual ML. Some of these simulators may have been implemented primarily to serve other research purposes, but by providing visual representations they are suitable for synthetic visual training data collection as well.

**Offline rendering** refers to techniques where rendering speed is not of crucial importance, and both the central processing unit (CPU) and GPU can be used. Offline rendering enables, apart from simple methods like rasterization and ray casting, physically based ray and path tracing. To date, offline physically based rendering is most often the only way to achieve photorealism.

There are several offline renderers used for synthetic data generation, either stand-alone or integrated to open-source, e.g. Blender [BLD], or commercial 3D software suites.

**Object infusion** refers to techniques that render single or multiple objects offline, and infuse these on a background image to compose the final result. Moreover, we include in this category frameworks that employ a cut-and-paste style...
approach to image synthesis. This means that one or more objects are removed from an image, possibly modified and finally inserted to the same or a new background.

4. Training Data Generation Methods in Computer Vision

In this section, we provide an overview of the active areas within computer vision that make use of and benefit from image synthesis methods as a source of generating training data. The considered computer vision areas, shown in Table 1, are connected with but not limited to, the data generation frameworks presented in the image synthesis taxonomy in Figure 4. Table 1 demonstrates tasks where synthetic training data have been significantly used over the last decade. However, the list is extended to also include methods that cannot lie in the image synthesis taxonomy, such as algorithms that modify captured or synthesized images to create a new data set for a specific purpose.

Together with Figure 6, Table 1 can give insights in the development of image synthesis frameworks for computer vision over time, while Figure 7 draws connections to the computer graphics taxonomy in Figure 4. Table 1 should not be seen as static portrayal, but rather as a dynamic list where the balance of applications is subject to change as new fields and example methods are added. In addition, the presented methods are listed under the tasks that are mentioned in the original papers. However, we need to emphasize that it is possible that several of these techniques, and the synthetic data sets they produce, can be applied to solve other tasks too. That is, one type of ground truth for supervised learning, suitable for a certain problem, can often be used to produce data for other applications. For example, semantic segmentation labels can be used to provide object detection bounding boxes. In the accompanying analysis (Section 5), we will highlight the common trends as well as the main differences in the selected approaches.

Following the categorization of Szeliski [Sze10], we have found that image synthesis for use in visual ML applications has been primarily utilized in the following computer vision categories:

- **Feature-based alignment** relies on 2D or 3D feature correspondences between images and estimates of their motion. Synthetic data generation has been mostly used for human body or object pose estimation and single or multi-object tracking tasks.

- **Dense motion estimation** tries to determine the motion between two or more subsequent video frames. Optical flow is among the most extensively researched computer vision tasks where synthetic data frameworks have been widely developed.

- **Stereo correspondence** aims to estimate the 3D model of a scene from two or more 2D images. Depth, disparity and scene flow estimation are representative tasks where synthetic data methods have been used.

- **Recognition** focuses on scene understanding, either identifying the contextual components of the scene, or determining if specific objects and features are present. The recent research focus on neural networks for robotics and autonomous driving has led to advances in semantic, instance and point-cloud segmentation, object detection and class and face recognition. For this reason, the number of image synthesis methods for these tasks is ranked in the first place in the taxonomy tree.

- **Structure from motion** estimates 3D point locations, structures and egomotion from image sequences with sparse matching features. In this category, visual odometry is a basic and low-level task where synthetic data have been explored.

- **Computational photography and Image formation** apply image analysis and processing algorithms to captured sensor data, to create images that go beyond the capabilities of traditional imaging systems. In this framework, the context is extended to also include the cases where parts of the camera imaging pipeline are applied onto synthetically generated images. Camera design for a particular computer vision task, and noise modelling has benefited from the use of synthetic data and image synthesis approaches over the past years. Image formation studies the constituting image elements: lighting conditions, scene geometry, surface properties and camera optics. Intrinsic image decomposition has commonly been utilizing synthetically generated data.

5. Image Synthesis Methods Overview

This section provides an in-depth exploration of the similarities and differences of the various synthetic training data generation methods. To this end, we provide an overview of the methods grouped according to the computer vision application areas and tasks in Table 1, to support visual ML applications research and development. Moreover, the methods are explained in the context of the image generation taxonomy from Section 3, to provide a clear picture of which techniques have been used for image generation.

5.1. Basic concepts

There are two basic concepts useful to better understand and reason around the design choices of existing data synthesis methods. These deal with how to model and render scenes in order to bridge the gap between the synthetic data and the real world.

- **Domain randomization** [TFR*17] is a simple technique, applicable in principle to any generation method that builds data from square one for any task. It tries to fill in the reality gap between training and testing sets by randomizing the content in the simulated environments that produce training data. By providing enough
variability at training time, domain randomization aims to make the real world appear to the model as yet another variation.

A similar concept is what we in the rest of the paper refer to as rendering randomization, which randomizes the lighting conditions and camera configurations for image rendering. Lighting conditions incorporates the number, type, position and intensities of the light sources in the scene, while camera configurations involve variations in the camera extrinsic parameters and possibly trajectories.

5.2. Feature-based alignment

Pose estimation is the problem of determining the position and orientation of the camera relative to the object, or vice versa, in a scene. To solve this problem, the correspondences between 2D image pixels and 3D object points need to be estimated. Typical approaches of synthetic data generation pipelines designed for this problem have predominately focused on image diversity instead of realism, to prevent problems with overfitting. Such approaches are based on domain and rendering randomization techniques [TFR*17, SQLG15]. Su et al. [SQLG15] presented an object infusion data synthesis pipeline for creating millions of low-fidelity images, in terms of modelling and rendering complexity, with accurate viewpoint labels. The 3D models from ShapeNet [CFG*15] were used to produce new models by transforming these through symmetry preserving free-form deformations. The models were consequently rendered with lighting and camera settings randomly sampled from the distributions of a real data set and then blended with SUN397 [XHE*10] background images. Finally, the images were cropped by a perturbed object bounding box, to further increase the diversity. The pipeline and example images are shown in Figure 8(a).

Human body, or articulated pose estimation, is an important problem within general pose estimation where the configuration of a human body is estimated from a single, generally monocular, image. For this purpose, synthetic data generation frameworks are typically based on non-procedural and data-driven modeling methods, such as motion capture (mocap) data and statistical models of the shape and pose of the human body [ASK*05, HSS*09, LMR*15], and employing object infusion rendering techniques [PJA*12]. The same principles have also been used to generate data to support 3D human pose estimation [CWL*16, VRM*17] (Figure 8b). These methods are real-data oriented and utilize rendering randomization along with randomly sampled texture maps and background pictures from real images. One of the earliest approaches used synthetic depth images of humans, generated from mocap data along with rendering randomization, to solve the articulated body tracking problem [SFC*11]. Choosing a different modelling direction, Park et al. [PR15] extracts body parts from the first frame of a sequence and modifies these according to a pre-defined pose library. Fabbrini et al. [FLC*18] tackle the problem of multi-person pose estimation and tracking by providing ~10 million body poses collected from the Grand Theft Auto V (GTA-V) video game [GTA]. They developed a game mod and created virtual scenes of crowds and pedestrian flow along with behaviour alterations (such as sitting and running). The scenes were directed from real-world scenarios, i.e. recreated in the virtual world from existing references of pedestrians. On the offline rendering side, recent methods generate training data for human pose estimation utilizing rendering randomization with mocap data from a head mounted display view, and physically based rendering [TPAB19], as well as 3D models and corresponding animations from web-based 3D character services rendered with object infusion for hand pose estimation [ZB17, MXM].

5.3. Dense motion estimation

Optical flow estimation is one of the most challenging and widely used tasks in computer vision. In general, optical flow describes a sparse or dense vector field, where a displacement vector is assigned to a specific pixel position that points to where the pixel can be found in a subsequent image. Video sequences, or any other ordered set of image pairs, are used to estimate the motion as either instantaneous image velocities or discrete image displacements. Baker et al. [BSL*11] were one of the first ones to introduce synthetic training data for optical flow. In their work, they used
Figure 8: Image synthesis pipelines and example data for training feature-based alignment tasks. Domain and rendering randomization with object infusion have been the standard techniques for data synthesis in this application.

procedural modelling to create a set of eight annotated frames from one natural and one urban scene. The scenes were composed with a random selection of textures and surface displacements and rendered offline using the ambient occlusion approximation to global illumination [Lan02] (Figure 9a). Since then, several offline and real-time rendering data synthesis approaches have been developed, including the use of short animated films [BWSB12, MIH*16] and game/simulator engines [HUI13, GWCV16, RHK17]. These methods provide longer image sequences, and thus bigger training sets, as well as ground truth flow for all frames, large non-rigid motions and more complexity (like blur, atmospheric effects and specular surfaces). Moreover, by employing domain randomization, non-procedural physically based modelling and object infusion, Dosovitskiy et al. [DFI*15] and Mayer et al. [MIH*16] created the

(c) FlyingThings3D employs domain randomization and physically-based modeling in an offline rendering framework. © 2016 IEEE. Reprinted, with permission, from [MIH*16].

Figure 9: Synthetic data image pairs ((a) and (b)) and samples (c) along with optical flow ground truth images for training optical flow estimation.

Flying Chairs and FlyingThings3D, respectively. These are two of the most commonly used large-scale training data sets for optical flow estimation (Figure 9b–c). Flying Chairs use 3D CAD chair models [AME*14] infused on random background pictures. Its successor, ChairsSDHom [IMS*17] also incorporates tiny displacements, to improve small motions estimation, and a displacement...
5.4. Stereo correspondence

Given a rectified image pair, disparity is the relative difference in the positions of objects in the two images, while depth refers to the subjective distance to the objects as perceived by the viewer. Disparity and depth estimation are closely connected tasks in stereo matching, where the goal is to produce a uni-valued function in disparity space that best describes the geometry of the objects in the scene. The aforementioned data sets FlyingThings3D, Monkaa and Driving [MIH*16] provide, apart from optical flow, disparity ground truth maps (Figure 10a). The UnrealStereo data set [ZQC*18], on the other hand, is designed for disparity estimation using non-procedural and physically based modelled game scenes implemented and rendered in Unreal Engine 4 [UE4]. The majority of synthetic data generation frameworks for depth estimation rely on game/simulator engines for urban and traffic scenes [HUI13, ASS16, GWCV16] (Figure 10b), while Varol et al. [VRM*17] provide depth maps for human parts depth estimation.

While most methods focus on the disparity and depth estimation tasks, only a few provide ground truth data for scene flow estimation [MIH*16]. Menze et al. [MG15] created a subset of KITTI [GLU12] with scene flow ground truth, by annotating 400 dynamic scenes using 3D CAD models for all vehicles in motion, used for quantitative scene flow evaluation.

5.5. Recognition

Semantic segmentation is a key challenge for visual scene understanding with a broad range of applications. Image synthesis for this task has been one of the most active research areas over the past decade. Driving simulators and computer games with urban and traffic scenes revolutionized the way training data were generated by collecting images from already existing virtual worlds [HUI13, ASS16]. Numerous data generation approaches build upon extracting images and video sequences from the GTA-V [GTA] commercial computer game, utilizing dedicated middleware game mods, with the main issue to be the ground truth annotation process. Richter et al. [RVRK16, RHK17] (Figure 11a) presented a semi-automatic approach for pixel-level semantic annotation maps by re-constructing associations between parts of the image and label them to a semantic class through either rule mining [AS94] or a user interface annotation tool. Angus et al. [AEK*18] approach the problem from a different perspective by labelling the GTA-V game world at a constant human annotation time, independently of the extracted data set size. At the same time, real-time 3D development platforms, that enable automatically generated pixel-perfect ground truth annotations, were also used to built data generation frameworks employing hand-modelled virtual cities with different seasons and illumination modes [RSM*16, HJSE*17], semi-automatic real-to-virtual cloning methods [GWCV16] and procedural, physically based modelling [KPS19]. Wrenninge et al. [WU18, TKWU17] introduced the only to date photorealistic data set for urban scene parsing using procedural modelling, to create unique virtual worlds for each image and offline unbiased path-tracing rendering (Figure 11b).

SceneNet [HPSC16, HPB*16] has paved the way for the development of synthetic data sets for semantic segmentation for indoor environments by building an open-source repository of manually annotated synthetic indoor scenes capable of producing training sets with several rendering setting variations. Later works utilized Metropolis Light Transport rendering [VG97] to create large-scale data sets [SYZ*17, ZSY*17], and photon mapping to approximate the rendering equation in dynamically simulated 3D scenes based on real-data distributions of object categories [MHLD17] (Figure 11c).
The goal of object detection is to detect all instances of objects from a known class, such as people, cars or faces in an image. Object detection algorithms typically leverage ML or DL to produce meaningful results. Marin et al. [MVGL10, VLM*13] was probably the first to explore synthetic images generation, using a computer game (Half-Life 2 [HL2]) that depicts urban environments, by developing appropriate game mods. In such manner, several data generation methods followed based on capturing video sequences from the GTA-V game and provided large-scale synthetic training sets with vehicles, pedestrians and various objects detection annotations [JRBM*16, RHK17, HCW19, ACF*19]. More real-time rendering generation pipelines have been developed in 3D development platforms utilizing non-procedural physically based modelling [GWCV16, QY16, QZZ*17], non-procedural non-physically based modelling with domain and rendering randomization and object infusion [TPA*18] (Figure 12a) and procedural, physically based modelling in a structured domain and rendering randomization manner [PBB*18] (Figure 12b). Recently, offline rendering methods that employ both procedural and non-procedural physically based modelling have been also introduced [WU18, TTB18, HVG*19].

A closely related problem is object tracking that aims to detect and follow one or several moving objects in a video sequence. Multi-object tracking is nowadays increasingly associated with object detection, where a detector defines candidate objects, from monocular video streams, and a subsequent mechanism arranges them in a temporally consistent trajectory (tracking-by-detection). Real-time and non-procedural physically based modelling based data generation approaches have been mainly the source of annotated data for this task. Methods involving GTA-V extracted video sequences and Unity developed virtual worlds provide training data for multi-object and multi-person tracking [GWCV16, RHK17, FLC*18].

Among the various recognition tasks, face recognition is one of the most popular. Recognizing people, or face analysis in general, has been in the centre of research attention for many years as it can provide an automatic tool to interpret humans, their interactions and expressions [MWHN]. It is a mature discipline within the computer vision recognition area and quite a few synthetic data generation methods have assisted to provide solutions. Most of these methods involve fitting a statistical 3D model to captured data [BV99, BV03, RV03] and sampling from it in order to vary the facial properties and expression parameters, thus creating a diverse synthetic training data set. In addition, rendering randomization is commonly used, along with object infusion rendering [WHHB04, KSG*18] (Figure 13a). In a similar approach, Abbasnejad et al. [ASN*17] propose a data generation method for facial expression analysis where a 3D face template model, consisting of a shape and a texture component, is fitted to face scans from real data sets (Figure 13b).

In the area of recognition we also find the few, as of now existing, methods that take a learning-based approach to image
Figure 13: Synthetic training examples generated for face recognition.

Figure 14: Image synthesis for camera design. Photorealistic training data (A), along with pixel-level depth annotations (B), and sensor responses for different color filter arrays (C-D). Reprinted with permission from [AS*17].

5.6. Computational photography and Image formation

Building different cameras to capture the variations caused by different camera specifications, and subsequently annotating the necessary data for a specific application, is not a realistic scenario. The apparent necessity of developing software simulations of the camera sensor has made camera design an active computer vision research area that utilizes data generation methods [BJS17]. This task has been lately popular within the autonomous driving community, where the image synthesis methods are well established, and recent studies show the impact of camera effects in the learning pipeline [CSVJR18, LLFW20]. The introduced data generation techniques rely both on non-procedural and procedural physically based modelling and employ offline physically based rendering, which leverages the modern cloud-scale job scheduling possibilities to improve the rendering times [BFL*18, LSZ*19, LLFW19]. In this set-up, different types of sensors can be simulated, and attributes such as the colour filter array (CFA) and the camera pixel size can be sampled from a distribution of values (Figure 14).

Image processing can be the first stage in many computer vision applications, to convert the images into suitable forms for further analysis. Noise modelling is an important part of the imaging synthesis for constructing training data. Some are found in medical imaging recognition problems, where GANs have gained much attention during the last few years [YWB19]. There are several reasons why generative image synthesis is interesting in medical imaging. First of all, it is difficult to collect large amounts of data, both due to restrictions in capturing, and since annotations are time-consuming and require experts. Moreover, medical images are usually represented by different modalities than natural images, such as computed tomography (CT) scans, magnetic resonance imaging (MRI), ultrasound, or digital pathology slides. Thus, it is not possible to use classical image generation methods for image synthesis. One popular application of GANs is for data augmentation of medical images [FADK*18, STR*18]. There is also a number of examples for demonstrating how GANs can learn to generate high-quality medical images, but which have not yet been tested in ML applications [GVL17, CBAR17, SST*18]. Shin et al. [STR*18] used a conditional GAN [IZZE17] to create synthetic MRI images of abnormal brains from segmentation masks, for the purpose of tumor segmentation. Costa et al. [CGM*17] considered vessel segmentation of retinal images, but instead of providing existing segmentation masks, these were synthesized from an adversarial auto-encoder. The synthesized segmentations were subsequently transformed to full retinal colour images by means of a conditional GAN. Although these segmentation tasks are focused on segmentation of one type of feature, we categorize them as semantic segmentation tasks that use binary pixel labels.

GANs have also been suggested for the purpose of anonymization of medical data [GVL17, STR*18], where privacy concerns are common. For the general purpose of anonymization, Triastcyn and Faltings [TF18] focused on the problem of providing a privacy guarantee by means of the differential privacy definition (DP) [DMNS06]. To increase privacy preservation in the generated images, the last layer of the discriminator was modified by clipping its input and adding noise. The GAN was tested for generating training data for classification on the MNIST and SVHN data sets.

Apart from medical imaging applications, GANs have been also recently used to generate photorealistic images to enhance the training data sets for face recognition applications [TMH18, GBKK18]. Finally, Kar et al. [KPL*19] presented a generative model to produce training data matching real-world distributions for any recognition task. They employed procedural modelling to generate scene graphs that are later used to parametrize a neural network aiming to minimize the distribution gap between simulated and real data.
pipeline which usually aims to solve denoising and help in synthetic-to-real adaptation problems. For these applications, several approaches have been proposed, from the most commonly used Poissonian–Gaussian noise model to noise models based on conditional normalizing flow architectures. These are used for modifying existing data, either captured or synthetic, into training sets suited for the problem at hand [BHLH19, ABB19]. For the same purposes, Lehtinen et al. [LMH*18] used low-samples-per-pixel path-tracing renderings, where the Monte Carlo image synthesis noise is analytically intractable, to train a denoising network without clean training data or image priors.

Intrinsic image decomposition is a long-standing computer vision problem, focusing on the decomposition of an image into reflectance and illumination layers [BKPB17]. Early approaches used simple hand-modelled scenes, populated with either single or few objects, with accurate light transport simulation enabled by photon mapping [BSvdW*13]. Later methods built their scene content using 3D models and scene databases, along with rendering randomization, measured materials and environment maps for global illumination, while employing various flavours of path-tracing and tone mapping algorithms [RRF*16, LS18, BLG18, BSP*19] (Figure 15a). In the same spirit, Baslamisli et al. [BGD*18] present a data generation method for natural images of flora, utilizing procedural physically based modelling, suitable to learn both intrinsic image decomposition and semantic segmentation (Figure 15b). Finally, Sial et al. [SBV20] used multi-sided rooms with highly variable reflectances on the walls instead of environmental maps to illuminate non-procedural and non-physically based modelled 3D scenes, rendered in an object infusion, offline fashion (Figure 15c). They claim that the cast shadows and the physical consistency that some point light sources in a synthetic 3D textured room generate, can benefit the image decomposition task.

6. Qualitative Comparisons

Attempting to provide insights into the generation and performance of the presented data generation methods, along with potential usability guidelines, we define qualitative criteria in order to rank methods within the different computer vision application areas. It has to be clear that these relative quality indices are derived by linear combinations of the originally reported performances of the methods and a qualitative ranking that we define regarding the data complexity, and not on the perceived quality of the generated images. For this reason, we provide this comparative quality metric only for the computer vision areas and tasks where we can derive meaningful results (Tables 2–5).

---

**Table 2: Feature-based alignment data generation frameworks.** The top part of the table refers to object pose estimation, while the bottom part refers to human body pose estimation. The quality score refers to a relative index between the methods of the specific computer vision area, based on generation complexity and reported performances criteria. Quantity and resolution are not incorporated in the quality index.

| Method   | Year | Sequence | Quantity | Resolution | Relative quality |
|----------|------|----------|----------|------------|------------------|
| [SQLG15] | 2015 | –        | –        | –          | **☆☆☆☆☆**       |
| [TFR*17] | 2017 | –        | –        | –          | **☆☆☆☆☆**       |
| [PIA*12] | 2012 | ✓        | –        | –          | **☆☆☆☆☆☆**      |
| [PR15]   | 2015 | ✓        | 400      | –          | **☆☆☆☆☆**       |
| [CWL*16] | 2016 | –        | 5 099 405| –          | **☆☆☆☆☆☆☆**     |
| [VRM*17] | 2017 | ✓        | 6 536 752| 320 × 240  | **☆☆☆☆☆☆**      |
| [FLC*18] | 2018 | ✓        | 460 800  | 1920 × 1080| **☆☆☆☆☆☆**      |
| [TPAB19] | 2019 | –        | 383 000  | 1024 × 1024| **☆☆☆☆☆**       |

**Table 3: Dense motion estimation data generation frameworks.** All of the methods relate to optical flow. [BSL*11] – G/U and [MIH*16] – FT refer to Grove, Urban and Flying Things 3D data sets, respectively. The quality score refers to a relative index between the methods of the specific computer vision area, based on generation complexity and reported performances criteria.

| Method   | Year | Sequence | Quantity | Resolution | Relative quality |
|----------|------|----------|----------|------------|------------------|
| [BSL*11] – G | 2011 | ✓        | 8        | 640 × 480  | **☆☆☆☆☆**       |
| [BSL*11] – U | 2011 | ✓        | 8        | 640 × 480  | **☆☆☆☆☆**       |
| [MIH*16] – FT | 2016 | ✓        | 25 066   | 960 × 540  | **☆☆☆☆☆**       |
| [RHK17]   | 2017 | ✓        | 254 064  | 1920 × 1080| **☆☆☆☆☆**       |
Table 4: Stereo correspondence synthetic data sets and data generation frameworks. [MIH*16] – FT/M/D refer to Flying Things 3D, Monkaa and Driving data sets, respectively. The top part of the table refers to scene flow estimation, the middle one to disparity estimation and the bottom part to depth estimation. The quality score refers to a relative index between the methods of the specific computer vision area, based on generation complexity and reported performances criteria. Quantity and resolution are not incorporated in the quality index.

| Method | Year | Sequence | Quantity | Resolution | Relative quality |
|--------|------|----------|----------|------------|-----------------|
| [MG15] | 2015 | ✓        | 400      | –          | ✔              |
| [MIH*16] – FT | 2016 | ✓        | 26 066   | –          | ✔              |
| [MIH*16] – M | 2016 | ✓        | 8591     | 960 × 540  | ✔              |
| [MIH*16] – D | 2016 | ✓        | 4392     | –          | ✔              |
| [VRM*17] | 2017 | ✓        | 6 536 752 | 320 × 240  | ✔              |

Data complexity is threefold and refers to computational efficiency, visual complexity and data production competence. Computational efficiency is related to the rendering speed, while visual complexity aims to reflect the depth of computer graphics techniques integrated to the proposed image synthesis framework, e.g. use of procedural algorithms or physically based light transport simulation. Data production competence concerns the production pipeline efficacy in terms of automatic or manual procedures, including estimating human-hours scripting, hand-modelling or annotating. Therefore, a training data generation method is ranked with the highest data complexity score when utilizes a low human effort, automatic generation process while employing photo-realistic, real-time rendering. In addition, we use the originally reported performance for each method and evaluate it according to the percentage of improvement that it achieves compared with the reported baselines, on real or other synthetic data, whenever they are provided. The linear combination of the originally reported performance, along with the data complexity measure defines the final relative quality index. The quantity and resolution of the evaluated data set are not incorporated in the criteria, as it is often the case that a method is capable of producing variations of these factors, and the denoted values are for demonstration purposes. It is then left to the reader to consider these factors for the final assessment.

In Table 2, we provide an example set of synthetic data generation methods for pose estimation. For the problem of object pose estimation, the best performing method, according to our subjective rating system, is presented by Su et al. [SQLG15] that manages to achieve high performances with a simple and fast domain and rendering randomization image synthesis pipeline developed in an object infusion fashion. It seems that for the task of viewpoint estimation, a large-scale data set of low fidelity but highly varied images is sufficient for accurate results. In the area of human body pose estimation, the selected methods seem to perform roughly the same, with the top one to employ data-driven statistical models for the human body modelling and object infusion rendering. However, this subjective quality metric indicates that more room for research and improvement is still available.

Table 5: Recognition data generation frameworks. [RSM*16] – R refers to SYNTHIA–Rand data set. From top to bottom, the table partitions refer to semantic segmentation, object detection, object tracking, classification, point cloud segmentation and face recognition, respectively. The quality score refers to a relative index between the methods of the specific computer vision area, based on generation complexity and reported performances criteria. Quantity and resolution are not incorporated in the quality index.

| Method | Year | Sequence | Quantity | Resolution | Relative quality |
|--------|------|----------|----------|------------|-----------------|
| [RSM*16] – R | 2016 | ✓        | 13 400   | 960 × 720  | ✔              |
| [ASS16] | 2016 | ✓        | 60 000   | 1024 × 768 | ✔              |
| [VRK16] | 2016 | ✓        | 24 966   | 1914 × 1052 | ✔             |
| [HPK16] | 2016 | ✓        | –        | –          | ✔              |
| [VRM*17] | 2017 | ✓        | 6 536 752 | 320 × 240  | ✔              |
| [RHK17] | 2017 | ✓        | 254 064  | 1920 × 1080 | ✔             |
| [MHLD17] | 2017 | ✓        | 5 000 000 | 320 × 240  | ✔              |
| [ZYH*17] | 2017 | –        | 500 000  | –          | ✔              |
| [CMM*17] | 2017 | –        | 614      | 256 × 256  | ✔              |
| [STR*18] | 2018 | –        | 211      | 128 × 128  | ✔              |
| [AEK*18] | 2018 | –        | 1 355 568 | 1920 × 1080 | ✔             |
| [WU18] | 2018 | –        | 25 000   | 1440 × 720 | ✔              |
| [KPL*19] | 2019 | –        | –        | –          | ✔              |
| [TPA*18] | 2018 | –        | 460 800  | 1920 × 1080 | ✔             |
| [GWCV16] | 2016 | ✓        | 21 260   | 1242 × 375 | ✔              |
| [ACF*19] | 2019 | ✓        | 500 000  | 1920 × 1080 | ✔             |
| [KPL*19] | 2019 | –        | –        | –          | ✔              |
| [TFI18] | 2018 | –        | 32 × 32  | –          | ✔              |
| [YWS*18] | 2018 | –        | 10 848   | –          | ✔              |
| [ASN*17] | 2017 | –        | 648 × 490 | –          | ✔              |
| [KSG*18] | 2018 | –        | –        | –          | ✔              |
rendering data for scene flow estimation.

Coming to the most extensively developed area of recognition applications, Table 5 lists a range of synthetic data sets mostly for the tasks of semantic segmentation and object detection. Scene understanding is highly related to the real world as we humans perceive it, and therefore there have been conducted substantial efforts to generate realistic data. Although the method by Wrenninge and Unger [WU18] is computational costly, it outperforms the rest in terms of performance, visual complexity and production competence. It demonstrates that tuning factors of realism are of crucial importance for efficient recognition applications. For example, this includes aspects such as procedural modelling, for context-rich variations and production efficiency, as well as accurate approximation of light transport for promoting photorealism.

To summarize, given the defined qualitative criteria, it is evident that the perfect synthetic data set or generation method does not exist, yet. The ideal synthetic data set, designed for any area and task, would be generated in real time through automatic procedures that integrate procedural modelling and physically based rendering, with possibly some level of domain and rendering randomization, in order to be able to outperform current benchmarks significantly. Although remarkable efforts have been carried towards this direction, we highlight that there are still many possibilities for improvement.

7. Discussion

From the categorization in Figure 4 and Table 1, it is clear how a wide variety of applications and image synthesis methods have been covered. Despite this, we believe that synthetic training data for visual ML are still in its infancy, and there are many challenges and opportunities that could be investigated in the future. Synthetic data have the potential of becoming one of the central components in the ML pipeline. In this section, we provide a discussion focusing on some of the topics we believe to be important for future work.

Optimal image generation. The existing ML methods that make use of synthetically generated images have predominantly done so in order to reduce the time and cost required for capturing and annotating images for supervised learning. This is why the computer vision tasks with most existing work on synthetic training data are where annotation is a main limiting factor, such as optical flow estimation and semantic segmentation. There are few examples of synthetic data sets, e.g. image-level classification, where instead augmentation of the data at hand is a more common strategy. However, with the rapid increase in quality of image synthesis methods for ML, this will likely be different in the future.

Methods for synthetic image generation not only make it possible to create large numbers of training samples, but they also allow for detailed control of image content. This opens up for future work on optimal sampling of the data distribution, and for synthesis of images that have the optimal distribution within the image. For example, learning could benefit from having certain objects overrepresented, or constructed unnaturally complicated.

Another problem of real data is bias; even if we do not want the model to be biased, it will learn the bias contained in the training data, in terms of parameters such as gender, age and ethnicity. Using synthetically generated content makes it possible to control the distribution of training data in such way not to allow for biases. In order for this to work, however, the scene modelling needs to be formulated to not learn from data that already exist (data-driven modelling in Figure 4).

Benchmarking. The current approaches to learning from synthesized data are focused on training using the generated data, but the model is then to be evaluated on real-world data. This makes sense, as the end goal most often is to utilize a trained model in a real-world scenario. However, given that synthetic images can be created with high quality, it would also be of interest to evaluate a model on synthetically generated images. As synthetic image generation allows for fine control over the image content, and can produce a large variety of meta-data, this opens up for possibilities in detailed statistical testing, across widely different parameters. For example, it could be possible to benchmark performance under different demographic circumstances, or for individual types of objects and scene content. It is also possible to test the model in situations that are difficult to capture with real-world sensors, e.g. traffic accidents for autonomous driving, or environments that are outside the training data.

Detailed control over the data generation and the feature distribution statistics in the data enables new ways of analysing the data and model performance during testing. Meta-information describing features such as depth, camera parameters, occlusion data...
and other information and statistics describing the simulated scene can be generated in addition to the simulated sensor measurements. Such meta-information can be generated in very flexible ways, e.g. at a per pixel, per image, per class or per instance level, and give valuable information for data and algorithm analysis. An example of this is shown in Figure 16 where Wrenninge and Unger [WU18] use meta-information to analyse how a model trained for object detection performs (detection of pedestrians and cars) as a function of depth (top row), as well as depth and occlusion (bottom row). In the top row, the analysis is also comparing how the orientation (forward, backward, left and right facing) affects the performance. It is very difficult to accurately perform this type of detailed analysis without carefully generated synthetic data and meta-information. It is, however, important that the synthetic data are representative given both the task being performed/evaluated and the real data being simulated.

**Domain gap.** One problem of using synthetically generated images in ML is that the domain gap between synthetic and real images is still substantial. As of now, no methods have been shown to work better when tested on real data and trained on purely synthetic data, given that the problem is complex enough. The differences between the synthetic and real domains are often both due to rendering performance as well as scene modeling.

For future development, it will be of importance to make the domain gap smaller. Since photorealistic rendering is fully capable of generating very realistic images, the problem lies mainly in computational resources, and in simulating all the details (objects, materials, light sources, camera parameters, imperfections, etc.) in a realistic way. However, if real data are also available, the diversity induced by combining synthetic and real data are also an important factor. Also, there could be cases where generalizing from synthetic to real data does not present a larger domain shift as compared to generalizing between different real data sets [MAKS16].

**Learning-based image generation.** With the tremendous progress in generative modelling over the last couple of years, mainly using GANs, we are today approaching a position where GANs can be used for image synthesis of training data.

Although it still is difficult to allow for complete control of the image generation process, GANs are very promising for the future. It could be possible to infuse more and more information into the image generation process, such as physical constraints, demographic information or detailed scene information.

One of the fundamental problems with a learning-based image generation approach, however, is that the generative model cannot go outside the distribution of the data used to train the model. Thus, allowing increased control of image generation and enabling generation of images far from the training data distribution, will require to formulate the image generation differently. For example, one option could be to combine classical methods for scene modeling with a GAN for rendering [IZZE17, WLZ*18], or the modeling could be separately controlled by another GAN [KPL*19]. A GAN could also be used explicitly for bridging the gap between synthetic images and real-world images [SPT*17], e.g. by means of techniques for unsupervised domain adaptation such as the CycleGAN [ZPIE17].

**Privacy and ethics.** Attempting to touch on synthetic data from another perspective and as part of the big data era, we foresee their potential to become a useful tool, under well-defined frameworks and purposes, to address a set of privacy issues that arise along with the continually increasing need for training sets that in some cases contain sensitive information. Especially when personal information is involved, either concerning unique physical characteristics or medical records, synthetic data schemes can be used to ensure both data diversity and individual privacy protection. As elaborated earlier, currently, anonymization is one of the most interesting applications for GAN generated training data for medical applications [GVL17, STR*18], while efforts towards the anonymity of subjects have also been pursued in methods that utilize data-driven modeling of humans [VRM*17]. We need to emphasize that the more we support the development of data generation methods and the use of synthetic visual training data, the more we recognize the need for minimizing the risks of deanonymization.

In the general context of ethics, we believe that the use of synthetic data, in a legal and human-rights preserving manner, can significantly contribute to the democratization of high-level computer vision applications. As it was mentioned earlier, the data set bias is one of the main training data-related problems. So far, many vision applications have been developed, employing data sets coming from specific target domains. While domain adaptation [CCC*17, HTP*18] tries to reduce the gap between source and target distributions as a way towards domain generalization, synthetic data sets offer the possibility to integrate the world’s variations directly in the generation pipeline. However, without a doubt, data set bias is not a feature of captured data only, but the synthesis pipeline can introduce it, too. A content generation that depicts reality and provides at the same time enough variations of both the majority and minority scenarios, coupled with photorealistic rendering, seem to be currently the tools to create visual data for scene understanding and detection applications.

8. Conclusion

The recent advances in ML, and particularly in DL, have made it clear that the development of efficient algorithms to a large extent relies on the available training data fed to the learning algorithm. Image synthesis has proven to be a catalyzing factor in the creation of training data, allowing for flexibility both in the quality and quantity of the generated data. Image synthesis adapts to the application needs, allows for high levels of control in the production pipeline and has the potential to automate the data synthesis procedure under well-defined ethical frameworks.

This survey presented an in-depth overview of image synthesis methods for visual ML. The focus of the overview was on computer vision applications, and on computer graphics techniques for image synthesis. In order to describe and categorize approaches for synthetic data set generation, we introduced a taxonomy of the modeling and rendering steps in the image synthesis pipeline. These can be broadly divided into procedural or non-procedural scene content generation methods and real-time or offline rendering, respectively. Moreover, the descriptions of the image synthesis methods
were also made in the context of the application areas within computer vision that the generated data were aimed for. The accompanying qualitative comparisons described the different methods and data sets in terms of data complexity, visual complexity, the efforts required to generate data, as well as the originally reported performances obtained when training and testing ML models. We have also emphasized the importance of synthetic data for overcoming some of the inherent problems with real-world data, such as capturing and annotation cost, bias, privacy issues and limited control over the generated material. For future development, we foresee an increase in the research around image synthesis for optimizing the training data distribution, e.g. where real data capture the main mode of the distribution and synthetic data can be used to model the rare (and even unrealistic) samples.

Acknowledgment

This work was partially supported by the strategic research environment ELLIIT and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

[ABB19] Abdelhamed A., Brubaker M. A., Brown M. S.: Noise flow: Noise modeling with conditional normalizing flows. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Seoul, South Korea, 2019).

[ACB17] Ariovsky M., Chintala S., Bottou L.: Wasserstein GAN. arXiv preprint arXiv:1701.07875 (2017).

[ACF*19] Amato G., Ciampi L., Falchi F., Gennaro C., Messina N.: Learning pedestrian detection from virtual worlds. In Image Analysis and Processing – ICIAP 2019 (Cham, 2019), E. Ricci, S. Rota Bulò, C. Snoek, O. Lanz, S. Messelodi and N. Sebe (Eds.), Springer International Publishing, pp. 302–312.

[AEK*18] Angus M., ElBalkini M., Khan S., Harakeh A., Andrienko O., Reading C., Waslander S., Czarnecki K.: Unlimited road-scene synthetic annotation (URSA) dataset. In Proceedings of 21st International Conference on Intelligent Transportation Systems (ITSC) (Maui, Hawaii, USA, 2018), IEEE, pp. 985–992.

[AME*14] Aubry M., Maturana D., Efros A., Russell B., Stivic J.: Seeing 3D chairs: Exemplar part-based 2D-3D alignment using a large dataset of CAD models. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Columbus, OH, USA, 2014).

[AS94] Agrawal R., Srikant R.: Fast algorithms for mining association rules. In Proceedings of 20th International Conference on Very Large Data Bases (VLDB) (Santiago de Chile, Chile, 1994), vol. 12/3, pp. 487–499.

[ASE17] Antoniou A., Storkey A., Edwards H.: Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340 (2017).

[ASK*05] Anguelov D., Srivivasan P., Koller D., Rodgers J., Davis J.: SCAPE: Shape completion and animation of people. ACM Transactions on Graphics 24, 3 (2005), 408–416.

[ASN*17] Abbasnejad I., Sridharan S., Nguyen D., Denman S., Fookes C., Lucey S.: Using synthetic data to improve facial expression analysis with 3D convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops (Venice, Italy, 2017).

[ASS16] Alizea Shafaei J. L., Schmidt M.: Play and learn: Using video games to train computer vision models. In Proceedings of the British Machine Vision Conference (BMVC) (York, UK, September 2016), R. C. Wilson, E. R. Hancock and W. A. P. Smith (Eds.), BMVA Press, pp. 26.1–26.13.

[Bay63] Bayes T.: LII. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, F.R.S. communicated by Mr. Price, in a letter to John Canton, A. M. F. R. S. Philosophical Transactions of the Royal Society of London, 53 (1763), 370–418.

[BCG*18] Bowles C., Chen L., Guerrero R., Bentley P., Gunn R., Hammers A., Dickie D. A., Hernández M. V., Wardlaw J., Rueckert D.: GAN augmentation: Augmenting training data using generative adversarial networks. arXiv preprint arXiv:1810.10863 (2018).

[BDS19] Brock A., Donahue J., Simonyan K.: Large scale GAN training for high fidelity natural image synthesis. In Proceedings of International Conference on Learning Representations (New Orleans, LA, USA, 2019).

[BFB94] Barron J. L., Fleet D. J., Beauchemin S. S.: Performance of optical flow techniques. International Journal of Computer Vision 12, 1 (1994), 43–77.

[BFL*18] Blasinski H., Farrell J., Lian T., Liu Z., Wandell B.: Optimizing image acquisition systems for autonomous driving. Electronic Imaging 2018, 5 (2018), 161:1–161:7.

[BGD*18] Baslamisli A. S., Groenestege T. T., Das P., Le H.-A., Karaoglou S., Gevers T.: Joint learning of intrinsic images and semantic segmentation. In Proceedings of the European Conference on Computer Vision (ECCV) (Munich, Germany, 2018).

[BHLLH19] Bieleva O., Hansch R., Ley A., Hellwich O.: A digital image processing pipeline for modelling of realistic noise in synthetic images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (Long Beach, CA, USA, 2019).

[BJS17] Buckler M., Jayasuriya S., Sampson A.: Reconfiguring the Imaging pipeline for Computer Vision. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Venice, Italy, 2017).

[BKPB17] Bonneel N., Kovacs B., Paris S., Bala K.: Intrinsic decompositions for image editing. Computer Graphics Forum 36, 2 (2017), 593–609.
[BLD] Blender Foundation. Retrieved August 4, 2020, https://www.blender.org.

[BLG18] BaslamiSl A. S., Le H.-A., Gevers T.: CNN based learning using reflection and retinex models for intrinsic image decomposition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Long Beach, CA, USA, 2018).

[Bli77] Blinn J. F.: Models of light reflection for computer synthesized pictures. ACM SIGGRAPH Computer Graphics 11 (1977), 192–198.

[Bli78] Blinn J. F.: Simulation of wrinkled surfaces. ACM SIGGRAPH Computer Graphics 12 (1978), 286–292.

[BLS*11] Baker S., Scharstein D., Lewis J., Roth S., Black M. J., Szeliski R.: A database and evaluation methodology for optical flow. International Journal of Computer Vision 92, 1 (2011), 1–31.

[BSP*19] Bi S., Sunkavalli K., Perazzi F., Shechtman E., Kim V. G., Ramamoorthi R.: Deep CG2Real: Synthetic-to-real translation via image disentanglement. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Seoul, South Korea, 2019).

[BV99] Blanz V., Vetter T.: A morphable model for the synthesis of 3D faces. In Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (Los Angeles, CA, USA, 1999), pp. 187–194.

[BV03] Blanz V., Vetter T.: Face recognition based on fitting a 3D morphable model. IEEE Transactions on Pattern Analysis and Machine Intelligence 25, 9 (2003), 1063–1074.

[BWSB12] Butler D. J., Wulff J., Stanley G. B., Black M. J.: A naturalistic open source movie for optical flow evaluation. In Proceedings of European Conference on Computer Vision (Florence, Italy, 2012), Springer, pp. 611–625.

[BZS*19] Bau D., Zhu J.-Y., Strobelt H., Zhou B., Tenenbaum J. B., Freeman W. T., Torralba A.: Visualizing and understanding generative adversarial networks. In Proceedings of International Conference on Learning Representations (New Orleans, LA, USA, 2019).

[CAT] Free3D Cat. Retrieved August 4, 2020, https://free3d.com/3d-model/cat-v1-522281.html.

[CBAR17] Cheikh B. B., Bor-Angelier C., Racoceanu D.: A model of tumor architecture and spatial interactions with tumor microenvironment in breast carcinoma. In Proceedings of Medical Imaging 2017: Digital Pathology (Orlando, FL, USA, 2017), vol. 10140, International Society for Optics and Photonics, p. 101400C.

[CCC*17] Chen Y.-H., Chen W.-Y., Chen Y.-T., Tsai B.-C., Frank Wang Y.-C., Sun M.: No more discrimination: Cross city adaptation of road scene segmenters. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Venice, Italy, 2017).

[CDH*16] Chen X., Duan Y., Houthooft R., Schulman J., Sutskever I., Abbeel P.: InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. In Advances in Neural Information Processing Systems 29 (2016), D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon & R. Garnett (Eds.), Red Hook, NY, USA: Curran Associates, Inc., pp. 2172–2180.

[CFG*15] Chang A. X., Funkhouser T., Guibas L., Hanrahan P., Huang Q., Li Z., Savarese S., Savva M., Song S., Su H., Xiao J., Yi L., Yu F.: ShapeNet: An Information-Rich 3D Model Repository. Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[CGM*17] Costa P., Galdran A., Meyer M. I., Niemeiier M., Abramoff M., Mendonça A. M., Campilho A.: End-to-end adversarial retinal image synthesis. IEEE Transactions on Medical Imaging 37, 3 (2017), 781–791.

[Cha60] Chandrasekhar S.: Radiative Transfer. Dover Books on Intermediate and Advanced Mathematics. Dover Publications, New York, 1960.

[COR*16] Cordts M., Omran M., Ramos S., Rehfeld T., Enzweiler M., Benenson N., Roth S., Schiele B.: The Cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Las Vegas, NV, USA, 2016), pp. 3213–3223.

[CPK*17] Chen L.-C., Papandreou G., Kokkinos I., Murphy K., Yuille A. L.: DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence 40, 4 (2017), 834–848.

[CSVJR18] Carlson A., Skinner K. A., Vasudevan R., Johnson-Roberson M.: Modeling camera effects to improve visual learning from synthetic data. In Proceedings of The European Conference on Computer Vision (ECCV) Workshops (Munich, Germany, 2018).

[CWL*16] Chen W., Wang H., Li Y., Su H., Wang Z., Tu C., Lischinski D., Cohen-Or D., Chen B.: Synthesizing training images for boosting human 3D pose estimation. In Proceedings of Fourth International Conference on 3D Vision (3DV) (Stanford, CA, USA, 2016), IEEE, pp. 479–488.

[CZM*18] Cubuk E. D., Zoph B., Mane D., Vasudevan V., Le Q. V.: Autoaugment: Learning augmentation policies from data. arXiv preprint arXiv:1805.09501 (2018).
understanding. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (Stockholm, Sweden, 2016), IEEE, pp. 5737–5743.

[HSS*09] Hasler N., Stoll C., Sunkel M., Rosenhahn B., Siedel H.-P.: A statistical model of human pose and body shape. Computer Graphics Forum 28 (2009), 357–346.

[HTP*18] Hoffman J., Tzeng E., Park T., Zhu J.-Y., Isola P., Saenko K., Efros A., Darrell T.: CyCADA: Cycle consistent adversarial domain adaptation. In Proceedings of International Conference in Machine Learning (ICML) (Stockholm, Sweden, 2018).

[HUI13] Haltakov V., Unger C., Ilic S.: Framework for generation of synthetic ground truth data for driver assistance applications. In Pattern Recognition. J. Weickert, M. Hein and B. Schiele (Eds.), Springer, Berlin, Heidelberg (2013), pp. 323–332.

[HVG*19] Hodañ T., Vineet V., Gal R., Shaley E., Hanzelka J., Connell T., Urbina P., Sinha S. N., Guenter B.: Photorealistic image synthesis for object instance detection. In Proceedings of the IEEE International Conference on Image Processing (ICIP) (Taipei, Taiwan, 2019), IEEE, pp. 66–70.

[HZRS16] He K., Zhang X., Ren S., Sun J.: Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Las Vegas, NV, USA, 2016), pp. 770–778.

[IMS*17] Ilg E., Mayer N., Saikia T., Keuper M., Dosovitskiy A., Brox T.: Flownet 2.0: Evolution of optical flow estimation with deep networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Honolulu, Hawaii, USA, 2017).

[Ino18] Inoue H.: Data augmentation by pairing samples for image classification. arXiv preprint arXiv:1801.02929 (2018).

[IS15] Ioffe S., Szegedy C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015).

[IZZE17] Isola P., Zhu J.-Y., Zhou T., Efros A. A.: Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Honolulu, Hawaii, USA, 2017), pp. 1125–1134.

[JRBM*16] Johnson- Roberson M., Barto C., Mehta R. C. S., Skidhar S. N., Rosaen K., Vasudevan R.: Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (Singapore, 2016), 746–753.

[Kaj86] Kajinya J. T.: The rendering equation. ACM SIGGRAPH Computer Graphics 20, 4 (Aug. 1986), 143–150.

[KALL18] Karras T., Aila T., Laine S., Lehtinen J.: Progressive growing of GANs for improved quality, stability, and variation. In Proceedings of the International Conference on Learning Representations (Vancouver, BC, Canada, 2018).

[KLA19] Karras T., Laine S., Aila T.: A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Long Beach, CA, USA, 2019).

[KPL*19] Kar A., Prakash A., Liu M.-Y., Cameracci E., Yuan J., Ruxinaki M., Acuna D., Torralba A., Fidler S.: Meta-sim: Learning to generate synthetic datasets. In ICCV (2019).

[KPSC19] Khan S., Phan B., Salay R., Czarnecki K.: ProcSy: Procedural synthetic dataset generation towards influence factor studies of semantic segmentation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (Long Beach, CA, USA, 2019), pp. 88–96.

[KSG*18] Kortylewski A., Schneider A., Gerig T., Egger B., Morel-Forster A., Vetter T.: Training deep face recognition systems with synthetic data. arXiv preprint arXiv:1802.05891 (2018).

[KSH12] Krizhevsky A., Sutskever I., Hinton G. E.: ImageNet Classification with Deep Convolutional Neural Networks. In Advances in Neural Information Processing Systems 25 (2012), F. Pereira, C. J. C. Burges, L. Bottou & K. Q. Weinberger (Eds.), Red Hook, NY, USA: Curran Associates, Inc.), pp. 1097–1105.

[KW13] Kingma D. P., Welling M.: Auto-encoding variational Bayes. arXiv preprint arXiv:1312.6114 (2013).

[Lan02] Landis H.: Production-ready global illumination. In RenderMan in production: SIGGRAPH 2002 course 16. L. Gritz (Ed.). SIGGRAPH 2002 course 16 (2002) New York: ACM, pp. 87–100.

[LBD*89] LeCun Y., Boser B., Denker J. S., Henderson D., Howard R. E., Hubbard W., Jackel L. D.: Backpropagation applied to handwritten zip code recognition. Neural Computation 1, 4 (1989), 541–551.

[LBH15] LeCun Y., Bengio Y., Hinton G.: Deep learning. Nature 521, 7553 (2015), 436.

[Leg05] Legendre A. M.: Nouvelles méthodes pour la détermination des orbites des comètes. F. Didot, 1805.

[LKB*17] Litiens G., Kooi T., Bejnordi B. E., Setio A. A. A., Ciompi F., Ghafoorian M., Van Der Laak J. A., Van Ginneken B., Sánchez C. I.: A survey on deep learning in medical image analysis. Medical Image Analysis 42 (2017), 60–88.

[LLFW19] Liu Z., Lian T., Farrell J., Wandell B.: Soft prototyping camera designs for car detection based on a convolutional neural network. In Proceeding of the IEEE International Conference on Computer Vision (ICCV) Workshops (Seoul, South Korea, 2019).

[LLFW20] Liu Z., Lian T., Farrell J., Wandell B. A.: Neural network generalization: The impact of camera parameters. IEEE Access 8 (2020), 10443–10454.
Proceedings of the IEEE Conference on Computer Vision (ICCV) (Venice, Italy, 2017).

Proceedings of the 11th International Conference on Field and Service Robotics (2018), M. Hutter and R. Siegwart (Eds.), Springer International Publishing, pp. 621–635.

Proceedings of the Eleventh International Conference on Computer Vision, pp. 1221–1224.

Proceedings of the Ninth International Conference on Computer Vision (ICCV) (Shenzhen, China, 2018), Springer, pp. 909–916.

Proceedings of the 10th International Conference on Field and Service Robotics (2018), M. Hutter and R. Siegwart (Eds.), Springer International Publishing, pp. 621–635.

Proceedings of the 11th International Conference on Field and Service Robotics (2018), M. Hutter and R. Siegwart (Eds.), Springer International Publishing, pp. 621–635.

Proceedings of the 11th International Conference on Field and Service Robotics (2018), M. Hutter and R. Siegwart (Eds.), Springer International Publishing, pp. 621–635.

Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV) (Waikoloa Village, HI, USA, 2019), IEEE, pp. 1262–1270.

Proceedings of the 10th International Conference on Field and Service Robotics (2018), M. Hutter and R. Siegwart (Eds.), Springer International Publishing, pp. 621–635.
IEEE Computational Intelligence Magazine 13, 3 (2018), 55–75.

[YWB19] Yi X., Walia E., Babyn P.: Generative adversarial network in medical imaging: A review. Medical Image Analysis 58 (2019), 101552.

[YWS*18] Yue X., Wu B., Seshia S. A., Keutzer K., Sangiovanni-Vincentelli A. L.: A LiDAR point cloud generator: From a virtual world to autonomous driving. In ICMR ’18: Proceedings of the 2018 ACM International Conference on Multimedia Retrieval (New York, NY, USA, 2018), ACM, pp. 458–464.

[ZB17] Zimmermann C., Brox T.: Learning to estimate 3D hand pose from single RGB images. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Venice, Italy, 2017).

[ZCDLP17] Zhang H., Cisse M., Dauphin Y. N., Lopez-Paz D.: mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017).

[ZQC*18] Zhang Y., Qiu W., Chen Q., Hu X., Yuille A.: Unreal-Stereo: Controlling hazardous factors to analyze stereo vision. In Proceedings of the International Conference on 3D Vision (3DV) (Verona, Italy, 2018), IEEE, pp. 228–237.

[ZSY*17] Zhang Y., Song S., Yumer E., Savva M., Lee J.-Y., Jin H., Funkhouser T.: Physically-based rendering for indoor scene understanding using convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Honolulu, Hawaii, USA, 2017).

[ZYST19] Zhang S., Yao L., Sun A., Tay Y.: Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys 52, 1 (2019), 5:1–5:38.

© 2020 The Authors. Computer Graphics Forum published by Eurographics - The European Association for Computer Graphics and John Wiley & Sons Ltd