AI Perspectives

POSITION PAPER Open Access

Digital reality: a model-based approach to supervised learning from synthetic data

Tim Dahmen[1*], Patrick Trampert[1,2], Faysal Boughorbel[3], Janis Sprenger[1], Matthias Klusch[1], Klaus Fischer[1], Christian Kübel[3,4] and Philipp Slusallek[1,2]

Abstract

Hierarchical neural networks with large numbers of layers are the state of the art for most computer vision problems including image classification, multi-object detection and semantic segmentation. While the computational demands of training such deep networks can be addressed using specialized hardware, the availability of training data in sufficient quantity and quality remains a limiting factor. Main reasons are that measurement or manual labelling are prohibitively expensive, ethical considerations can limit generating data, or a phenomenon in questions has been predicted, but not yet observed. In this position paper, we present the Digital Reality concept are a structured approach to generate training data synthetically. The central idea is to simulate measurements based on scenes that are generated by parametric models of the real world. By investigating the parameter space defined of such models, training data can be generated in a controlled way compared to data that was captured from real world situations. We propose the Digital Reality concept and demonstrate its potential in different application domains, including industrial inspection, autonomous driving, smart grid, and microscopy research in material science and engineering.

Keywords: Machine learning, Synthetic training data, Data augmentation, deep learning

Introduction

Recent advances in machine learning, in particular deep learning, have revolutionized not only all kinds of image understanding problems in computer vision, but also the approach to general pattern detection problems for various signal processing tasks. Deep learning methods [1] can be applied to data that originates from almost any type of sensor, including image data from arbitrary modalities and most time-dependent data. Roughly speaking, in machine learning a very general computational model with a large number of free parameters is fitted to a specific problem during a training phase. The parameters are iteratively adjusted such that the computation performed by the model has minimal deviation from a desired result. In the case of supervised learning, the desired computation is specified by a finite set of input-output pairs, the training data. The machine learning model attempts to interpolate or extrapolate between the training data points using some concept of smoothness such that reasonable output for data not in the training set can be predicted. This is generally referred to as the ability of the model to generalize, which can be evaluated by a second set of input-output pairs, the test data. The particular success of supervised learning approaches in computer vision primarily stems from the tremendous advances in the achievable accuracy for classification tasks. Provided that the computational model has sufficient capacity and the training data set is large enough, particularly hierarchical neural networks can approximate a very wide range of functions and can be trained efficiently using backpropagation [2].

However, the availability of training data is the main problem of deep learning methods. For the task of general image understanding in computer vision, several standardized databases with millions of labelled images exist [3–5]. The databases have been created by joint efforts of the computer vision research community and constitute a considerable investment in machine learning research. For the application of deep learning methods to more specific problems, either scientific or industrial, labelled training data from in-vivo sources (see textbox for definition) does not exist in general. In-situ creation

* Correspondence: Tim.Dahmen@dfki.de
[1]Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
Full list of author information is available at the end of the article

of the data, can be a problem for numerous reasons. (1) If the data acquisition involves expensive measurement equipment or sample preparation, the cost of generating sufficient quantities of training data can be prohibitive. (2) In many application, there are ethical questions involved in acquiring training data, for example radiation exposure of patients or data from traffic accidents with human casualties. (3) Particularly for the case of semantic segmentation (per-pixel classification), labelling the training set can constitute a tremendous effort. (4) In many scientific applications, a phenomenon that was predicted from theory, but not yet observed, should be detected. In such cases, in-vivo and in-vitro training data is unavailable for principle reasons.

An additional concern with in-vivo training data relates to the clustering of data around common phenomena. In most scenarios, certain situations occur more frequently than others. In a production environment, most data will show undamaged parts while actual defects are rare. Even defective parts typically have a heterogeneous distribution where some defect types are common and others highly uncommon. Often, situations that are relevant to be detected occur rarely.

Definition: in-vivo, in-vitro, and in-silico data

In-vivo data is captured from real-life situations that were not primarily created or modified for the purpose of capturing the data. Examples are video streams from autonomous vehicles driving through a city, black-box data of accidents, and images of product defects from production.

In-vitro data is captured using physical sensors under lab conditions. Examples are footage of crash-tests, images of products that were intentionally damaged in the lab to capture the data, or images of surface materials taken in the lab under controlled lighting conditions.

In-silico data is generated without the use of physical sensors by software simulations. Examples are renderings of traffic scenes from a driving simulator, rendered images of defect products, or virtual crash-tests performed by simulations using the finite element method.

Consequently, in-vivo training data sets typically consist of large quantities of relatively uninteresting situations with rare instances of exceptional, but highly relevant situations. If used to train a machine learning system, this situation immediately translates to a class imbalance problem. In principle, this problem can be mitigated to some degree by manually filtering or selecting training data, and by some computational compensation for class imbalance. Nevertheless, the rate of occurrence of rare phenomena might be *very* low.

A vivid example of an exceptional situation is the child running in front of an autonomous driving vehicle. Cars in Germany drove $7.3 \cdot 10^{11}$ km in 2016 [6] and created 4195 severe accidents with children [7]. Consider subdividing all driven distance to chunks of 5 m length to obtain individual training data samples. One can estimate that approximately three out of 10^{11} such chunks contain images of children prior to a severe accident. For obvious reasons, in-vitro generation of the data is not possible. Resolving the class imbalance problem by capturing enough data and normalizing class balance by manual sorting is also not a valid option. Even if one could afford the sheer amount of work, the ethical implications of the approach is that one would need to wait for these 4195 severe accidents to happen to record the data required to avoid them rather than using in-silico data generation and preventing that the accidents must happen.

The core contribution of this position paper is the introduction of a concept called “Digital Reality” that solves these issues.

Figure 1) displays this generic blueprint of how machine learning models can be trained and validated using such synthetic training data. The approach applies to all data driven methods, particularly data-driven supervised or unsupervised learning with deep neural networks, and deep reinforcement learning [8].

The process starts by (1) creating partial models of reality by modeling, capturing, or learning individual aspects such as geometry, behavior, or physical properties including materials or lighting. (2) stating models are composed of parametric scenarios would invert the part-whole relation by manual configuration, data fitting, or machine learning. (3) Setting all parameters of such a scenario to fixed values creates a concrete instance of the scenario, corresponding to a simulation-ready 3D scene. (4) The scene is then rendered by a forward-simulation of the imaging process. (5) The resulting synthetic images are used to train a machine learning system.

The remainder of this paper is organized as follows: We first present a more detailed description of the individual steps of the Digital Reality concept. In section 2, we describe how partial models are obtained, and in section 3 we elaborate how training data can be generated from parametric models using sensor simulations. In section 4, we discuss how considering the parametric models from a sampling perspective can provide useful insights into data generation. In section 5, we present several Use Cases from different application areas to illustrate the Digital Reality concept on concrete examples, and in order to give some evidence that the concept is feasible.

Parametric models of the real world

The first step of the Digital Reality concept is the creation of partial models of the real world. Each of these partial models covers one specific aspect of reality in the context of a narrow field of application.

For example, in the context of defect detection in a production environment, partial models can cover the shape properties of the products, shape and characteristics of the defects, material properties of the product surfaces,

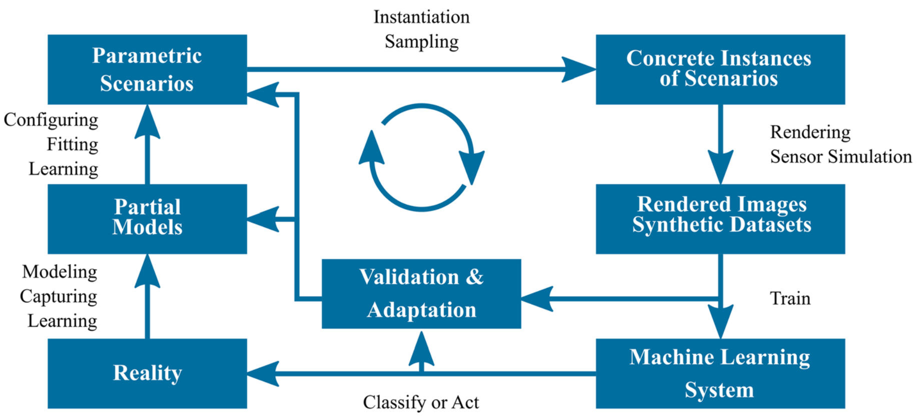

Fig. 1 The Digital Reality Concept. The process starts by creating partial models of reality by modeling, capturing, or learning individual aspects such as geometry, physical properties including materials, behavior, or lighting. The partial models are composed of parametric scenarios by manual configuration, data fitting, or machine learning. The composition is performed in such a way that all aspects of reality relevant to a specific simulation are provided. Setting all parameters of such a scenario to fixed values creates a concrete instance of the scenario, corresponding to a simulation-ready 3D scene. The scene is then rendered by a forward-simulation of the imaging process. The resulting synthetic images are used to train a machine learning system

lighting setup at the production site, or properties and physics of the imaging system. In the context of autonomous driving vehicles, a much longer list of partial models is perceivable. The list includes, among others, geometry and material properties of individual elements of a traffic scene such as roads, buildings, traffic signs, plants, vehicles, or pedestrians [9], layout of traffic scenes, behavior models of vehicles, traffic lights, and pedestrians, models of lighting conditions and weather, as well as models of the sensory systems available to the car including optical cameras, lidar, and the physics of the imaging process of each modality. The partial model is called parametric because each model is controlled by a set of input parameters and describes a part of a scene in a simulation environment as a function of these parameters.

Clearly, creating partial models of reality closely relates to science. However, there are differences between models created for the purpose of Digital Reality and general scientific models. Science aims to *understand* one aspect of reality, in the most general way possible. Therefore, models are only accepted if they are described in a form that is interpretable by humans. The value of a model is highly dependent on the range of its applicability. This means, a model is considered valuable if it can be applied to a wide variety of situations and *explains* one aspect of reality. Consequently, capturing data about a phenomenon without generating an abstract insight and interpretation is considered incomplete science.

In the context of Digital Reality, neither *understanding* nor *generality* of partial models are primary concerns. Instead, for the immediate purpose of training a machine learning system, a generative model with a narrow applicability to the problem is sufficient. The model does not necessarily need to be formulated in a way that is particular prone to human interpretation. Rather, any parametric model that is capable to generate the desired output is sufficient. Obviously, the partial models can still be created manually. The manual approach is ideal when obtaining of a deeper understanding of aspects of reality is of interest for reasons beyond machine learning. In other cases, capturing or learning a phenomenon is often more effective.

If a model has only little manually created structure but a large number of parameters that are automatically fitted against data, we refer to the process of creating the model as *capturing*. Typical examples for captured models are object geometries, surface properties of materials, emission properties of light sources, animation snippets, and many more. In the following, we give some examples of recent progress in capturing various types of models. In the special case that the architecture used for the model capturing is a neural network, we refer to the process as *learning* the model.

Capturing of appearance models

The most commonly captured type of partial model is the geometry of objects. Surface geometry is traditionally captured by 3D laser scanners. The scanners generate an unstructured set of points on the surface of an object. A model is then fitted to these points to establish connectivity and create a mesh. A viable alternative to laser scanners that is increasingly used in the computer game and movie industry is photogrammetry [10]. Apart from the obvious point that a digital camera is sufficient to perform a scan, photogrammetry has the advantage of capturing surface color and texture along with the shape. However, the captured textures include lighting information that must be removed in a non-trivial post-processing step called delighting [11]. Most 3D scanning approaches are limited to strictly diffuse objects. A fully automatic solution to handle glass and mirror surfaces has just been presented recently [12].

Geometry alone is insufficient for photorealistic rendering, as the appearance of objects strongly depends on their optical properties. The most basic model of the optical properties of material surfaces for rendering images is the Bidirectional Reflection Distribution Function (BRDF). Capturing BRDF has been a topic in computer graphics research for a long time, and various measurement devices and algorithms of different levels of complexity have been developed for this purpose [13]. In practice, most renderers use lower dimensional parametric models that are fitted against measured BRDF data. Small details in the surface geometry and in the BRDF are stored as a set of texture images for diffuse color, position (also called displacement), surface normal direction, reflectivity, roughness, and so on.

An interesting observation for many entertainment applications is that the characteristic features in color variation and the small geometric features captured in a surface normal are more important for the human perception of materials than the precise modelling of reflectance characteristics. A common workflow for capturing materials therefore consists of generating a high resolution elevation model of the surface using photogrammetry. Material textures are then generated from this model and the remaining free parameters of the material model are set manually to fit the model appearance.

Once partial aspects of the real world are modelled, the partial models can be composed to parametric scenarios in a simulation environment (Fig. 2). The scenario can be configured via a parameter space that consists of all parameters of the partial scenarios and potentially additional, scenario-specific parameters. If all parameters are set to concrete values, the generative model can produce a concrete instance of a scenario that corresponds to a simulation-ready scene.

Behavior model generation

When moving from static images to video, parametric models must include time dependent aspects, including behavior models. Particularly the behavior of digital human models (DHM) is of key importance because of their high-level of variability and the importance of correctly detecting humans. The control of DHMs can be separated into at least two major aspects. On the one hand, a controller which drives the basic mechanics of the artifact that represents the body of a human (not necessarily making use of physics for this purpose) and, on the other hand, an intelligent agent that drives the high-level behavior of the DHM.

In principle, human motion synthesis can be addressed with varying approaches and levels of detail, depending on the requirements of the specific domains. A recent overview of motion synthesis approaches is given in [14]. Current motion generation approaches for full body animation can be classified as either analytical or data-driven. Analytical motion synthesis aims to generate realistic motions based on intrinsic mathematical

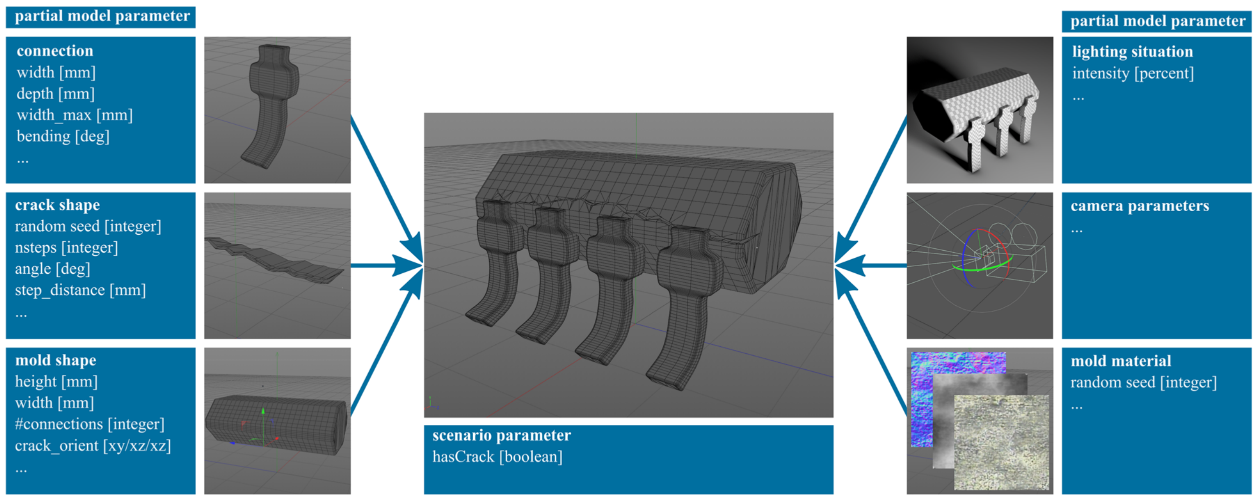

Fig. 2 An example of a parametric scenario, a microchip with a defect. The scenario is composed of different partial models that cover one aspect of reality each. Partial models concern the shape of the mold, the shape of the defect, the geometry of the connections, surface properties of mold and connections, as well as the camera and lighting setup. The parameter space of the parametric scenario is the union of the input parameters of all partial models and additional, scenario-specific parameters

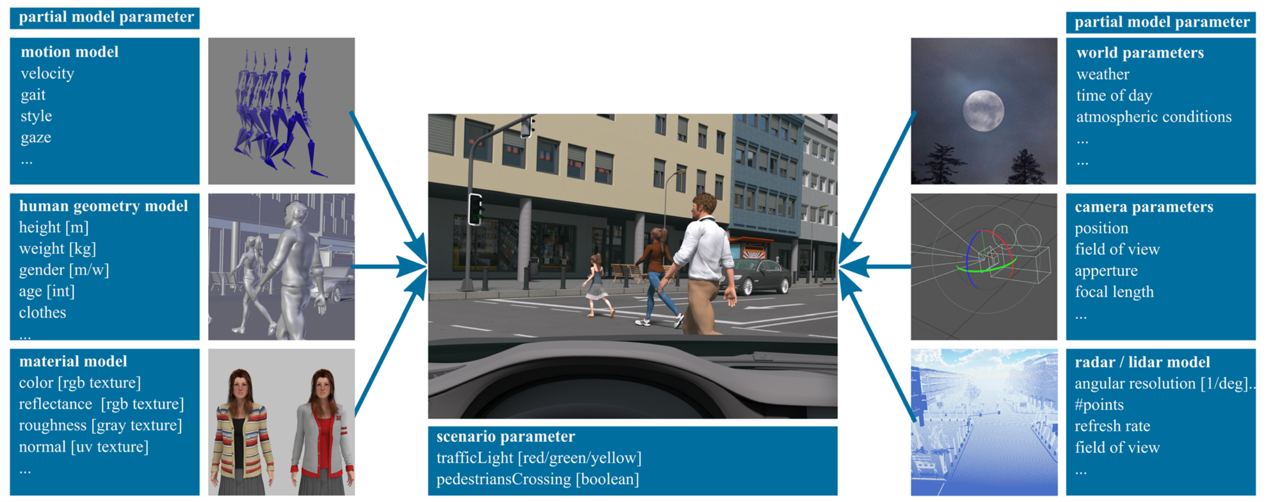

Fig. 3 A second example of a parametric scenario, a traffic scene. Again, the scenario is composed of different partial models that cover one aspect of reality each. The real-world environment exhibits a much larger number of parameters and variation compared to chip production depicted in Fig. 2. As the scenario is potentially imaged using different sensory modes (optical camera plus radar or lidar), partial models must provide the sensor-modality specific inputs for the simulation

or physics models [15]. In particular, inverse-kinematics-based approaches [16] are often utilized to manipulate motions of articulated avatars. In contrast to the analytical approaches, data driven or example-based motion synthesis approaches rely strongly on reference and example datasets, which are predominantly recorded by means of motion capturing. These approaches can be further subdivided into different categories: motion blending, which interpolates example clips [17, 18], motion graphs, which concatenate discrete segments or poses [19], and machine learning, in particular deep learning-based approaches, which approximate a function or statistical model [20, 21]. Recently, machine learning approaches using deep neural networks have shown promising results with comparatively little manual effort in the preprocessing [22, 23].

One level of abstraction above animation is the high-level control of a DHM. Work on the modelling of intelligent behavior goes back to the fifties of the past century. At that time researchers concentrated on developing general problem solvers, which worked in any given environment if it could be described in a formal manner. However, the success of such systems was rather limited because of the computational complexity of the presented problems. In the current state of the art on DHMs, behavior trees or belief, desire, intention (BDI) reasoning are mostly used in complex applications. In how far it is possible to combine these approaches with the planning from first principle approach of a general problem solver is a research question. In fact, whether and how it is possible to learn basic behavior, possibly using deep learning techniques, and combine it with symbolic reasoning approaches remains an open problem.

For autonomous driving, the behavior of pedestrians and bicyclists is the most difficult part to model, as in-vivo data is only partially available and cannot be obtained in many situations due to ethics and effort (Fig. 3). Current synthetic driving simulators either do not include pedestrians at all [24] or only display default game engine animations with predefined trajectories [25–27]. A review on models of pedestrian behavior in urban areas is presented in [28]. The work has a focus on route choice and crossing behavior. Most importantly for a Digital Reality, the authors propose a multi-level behavior model, very similar to the definition of multi-agent systems.

Shallow models in two dimensions

So far, we have considered the case that partial models are built close to the real world. In this case, models exist in a three-dimensional world space, and object properties are modelled from a relatively deep physical understanding of the measurement process. This allows the generation of in-silico images using low-level physical simulation of the measurement process, such as physics-based rendering or radar simulation. Such an approach is conceptually very clean and has clear advantages in terms of generality.

However, the capturing of all required models can constitute a tremendous effort, and the low-level sensor simulations can have very high computational cost. Though, in many situations, in-silico data can be generated in sufficient quality from more shallow models. Hereby, a typically two-dimensional model is generated purely from in-vitro or in-vivo data without the need to integrate a deeper physical understanding of the real world. Such image-based models are typically expressed in image processing terms such as intensities, frequencies and pixel-distances.

An example of such a shallow model is modeling of cracks in microchips, which can be used to train an optical inspection system. The model consists of a background texture with a crack model painted over the background, which is generated from a texture atlas using an exemplar-based inpainting approach for texture synthesis. The crack model itself consists of a polygon line of random width that extends in a primary direction, but deviates from that direction at random steps in random angles. The intensity profile of the crack is modeled by superimposing several semi-transparent lines with identical corner positions but different transparencies and line widths. All random parameters of the model are drawn from statistical distributions that were generated by manually measuring a set of 80 in-vitro images of cracks. The overall parametric model is depicted in Fig. 4.

Scientific model generation

Typical length scales relevant to production or traffic environments are the millimeter to meter scale. For many scientific applications, we are interested in much smaller or much larger length scales and sensory systems suitable for these scales. Paradoxically, our quantitative understanding of both matter and the imaging thereof is much more precise on both microscopic scale and on astronomical scales compared to the every-day environment that we live in. Using, for example, force field simulations as consistency check, we can model microstructure at the atomic level much more reliably than we can model objects at the every-day scale, such as cars, buildings, furniture, or humans. It is much easier to achieve a quantitatively correct simulation of an electron microscopy image compared to a quantitatively correct rendering of, for example, a human face at the visual optical spectrum. As Richard Feynman put it [29], it is possible to know everything about matter, all one would have to do is to look at it and see where the atoms are. With recent advances in imaging and analytical characterization techniques, it is nowadays possible to generate a good description of the atomic structure of materials.

For example, with the advent of aberration corrected transmission electron microscopy (TEM) [30] and increasingly sensitive detectors, it is possible to create a two-dimensional projection of the atomic structure of a thin object. This can be extended by in-situ TEM characterization to image the structural dynamics at the atomic level in response to external stimuli such as heat, electrical currents, strain, or specific gas environments [31–33]. The main challenge is to create a three-dimensional model from the atomic scale projections.

However, using convolutional neural networks, significant advances have been achieved, for example by identifying and tracking atomic positions in a metal nanoparticle exposed to a defined gas environment to follow the structural response of the nanoparticle [34]. Also in atom probe tomography

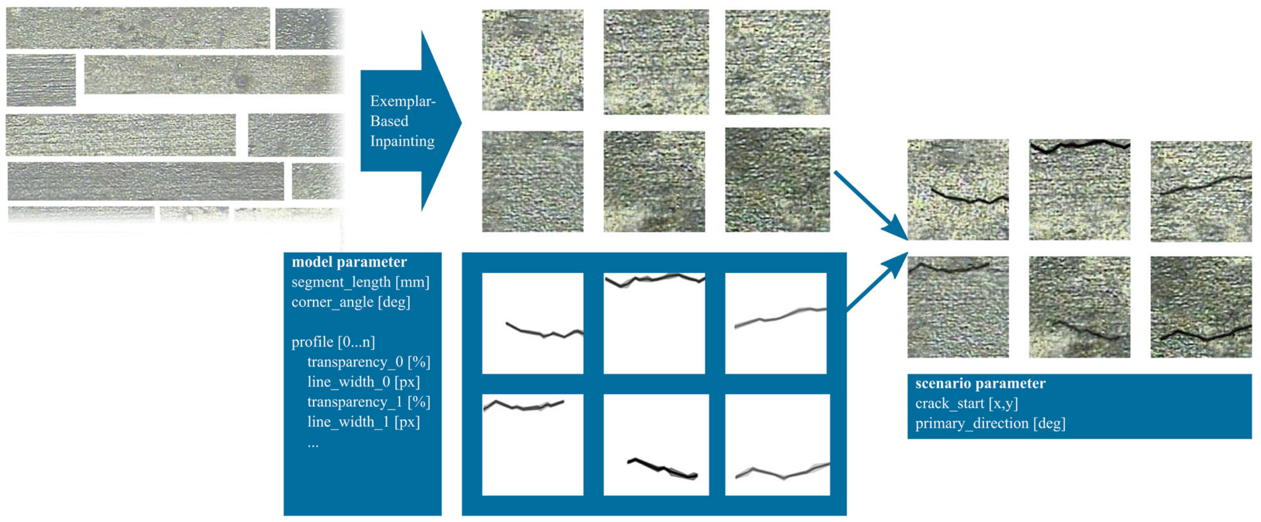

Fig. 4 An example of a shallow parametric model. The background is generated from image examples with a texture synthesis approach using exemplar-based inpainting. The crack is generated using a polygonal model. All parameters of the polygon such as step length, angle at each corner, line width and cross-section intensity profile are drawn as random variables from a statistical model. The model was generated by measuring real cracks

(APT) [35, 36], tremendous improvements have been achieved, which enable determination of the three-dimensional coordinates of around 50% of the atoms in nanoscale needle-shaped samples. With this progress, state-of-the-art analytical techniques are getting closer to fulfill Feynman's vision.

Sensor simulation and label generation

In-silico training data is generated from a scene using conventional simulation of the measurement process and sensor. In the case of optical camera systems, the simulation consists of rendering an image. Specific questions of how to render optical images accurately and efficiently have been the topic of computer graphics research for decades such that the problem is well understood and there exists a large number of both commercial and free software packages to address the task.

In the case of other sensor modalities such as lidar, radar, or X-ray imaging, the image formation is also well understood, but the range of available software for the simulation is smaller compared to optical systems. In either way, we assume that simulation software exists for all required sensor modalities.

In the case of in-vitro or in-vivo training data, the data must be labelled manually or using semi-automatic methods. Creating accurate labels per pixel is a tedious and time-consuming process. Trading reduced labelling precision for additional data is an option if sufficient images are available, but does not increase segmentation performance in general [37]. Per-pixel labels can be combined with per-image labels in suitable architectures [38]. In some cases, labels can even be generated automatically using a secondary sensory mode. For example, a network for the automatic segmentation of cells in optical microscopy images was trained using labels automatically generated by means of fluorescence microscopy [39]. This approach is very elegant but highly specific to the scenario and not available in general.

Besides generating the in-silico sensor data, the simulation system must also generate class labels. The exact format depends on the problem that should be solved by the network. In the case of supervised deep neural networks for image classification, the provided label is simply a class ID per image. For multi-object detection, the labels are a set of bounding boxes in the image space and an additional class label is assigned to each box. For semantic segmentation, the labels are a separate image channel that contains a class ID per pixel. Obviously, the required class label format corresponds to the output format of the network.

Fortunately, generating class labels directly from a parametric model is much easier than first rendering in-silico images and then generating the labels from these images. In many cases, the parametric model is constructed in such a way that one or more parameters directly correspond to class labels. In the example of the defective chip, the model contains a parameter *hasCrack* that triggers if a crack is generated in the scene or not and can directly be mapped to a class label. In the case of semantic segmentation, the rendering system can be configured to generate an object ID pass as a separate image or image channel, where the object ID is set to one for all pixels that cover the crack and to zero for all pixels not covering the crack. For multi-object detection networks, image space bounding box information can be generated trivially from the object ID images.

The generation of the class labels can be slightly more complicated than this, e.g. if one needs to consider the case that an object is generated in the scene but not visible for a specific camera position or setup, or if the model parameters do not map one-to-one to the class labels. Still, the parametric model was generated from an understanding of the problem domain, and as such, the terminology used to define the parameter space of the model is usually closely related to the desired class labels.

Sampling the parameter space

One key advantage of the Digital Reality approach for the generation of training data is that class balance can be achieved in an elegant and generic way by means of controlled sampling of the parameter space. If the parameter space is *well-behaved*, a simple uniform random sampling of the parameter space is sufficient. Each parameter is set to a random or pseudo random value, and the corresponding simulation model is then generated.

Well-behaved parameter spaces

A parameter space is called *well-behaved*, if the following conditions are met. First, all parameters should be *limited*. A parameter is *limited*, if there exists an interval [*lower*, *upper*], such that the parameter needs only be sampled inside the interval to cover all variability in the model. Enum-like parameters (i.e. parameters that are selected from a finite set of possible values) are always limited. Second, all parameters should have approximate constant influence over the entire parameter range. We define the *influence* of a parameter p by means of an image similarity measure N such as peak signal-to-noise ratio or structural similarity index. Consider a configuration C in parameter space. A second configuration C' is generated by adding a small value ε to the parameter p. The images corresponding to configuration C and C' are rendered and compared by the image similarity metric N. The influence of the parameter p is then the partial derivative dN / dp. Intuitively, this answers the question: how much influence does the parameter have on the rendered image. For the parameter space to be *well-behaved*, we demand that the influence of all parameters is approximately constant over their respective range. If this condition is violated, it can typically be compensated by measuring the influence along the parameter range of *p* and remapping the parameter *p* via some monotonous mapping function *p'* = *f(p)*. *f(p)* is chosen such that *dN / df(p)* is approximately constant. This mapping function can include a constant scaling factor to ensure that the influence of all parameters has the same magnitude.

As example for such a mapping, consider a scene consisting of an isolated object on a neutral background. In such a scene, the camera position is typically specified in a coordinate system consisting of the two Euler angles and the distance of the camera to the object. The rendering of the object covers a number of pixels that is approximately proportional to the inverse of the squared distance to the camera, a phenomenon widely known as inverse square law. The distance between two images measured by an image similarity metric is approximately proportional to the number of pixels affected for most metrics. Consequently, the distance of the camera to the object should be specified on a quadratic scale, such as *1 cm, 2 cm, 4 cm*, and so on. This ensures that increasing the distance by one step will lead to a constant amount of change in the image.

An initial goal in sampling a parameter space is to avoid class imbalance. This means that every output class of the model should be trained using approximately the same number of data points in the training set (Fig. 5a-b).

Black box adaptive sampling and importance sampling using the confusion matrix

In some situations, individual classes are more difficult to differentiate than others. In this case, a sampling that aims for a uniform number of training data points per output class might not be optimal. Consider a hypothetical scenario from autonomous driving. The system should detect several classes of traffic signs including stop and right of way signs, advertisement signs, pedestrians, and vehicles. Clearly, pedestrians exhibit a much larger variety of appearances compared to stop signs and consequently, a larger number of training samples is requred.

We propose an adaptive sampling scheme using the confusion matrix. The sampling is initialized by generating a fixed set of training samples optimized to achieve class balance as described above. The data set is used to train a network and evaluate it on a validation set. The confusion matrix for the validation set is computed. If occurrences of class A are often mistaken for class B, additional data of class A and B are generated. In the simplest case, additional samples corresponding to class A and B are generated using uniform random sampling. A more elaborate approach is to generate training data in pairs (Fig. 5e).

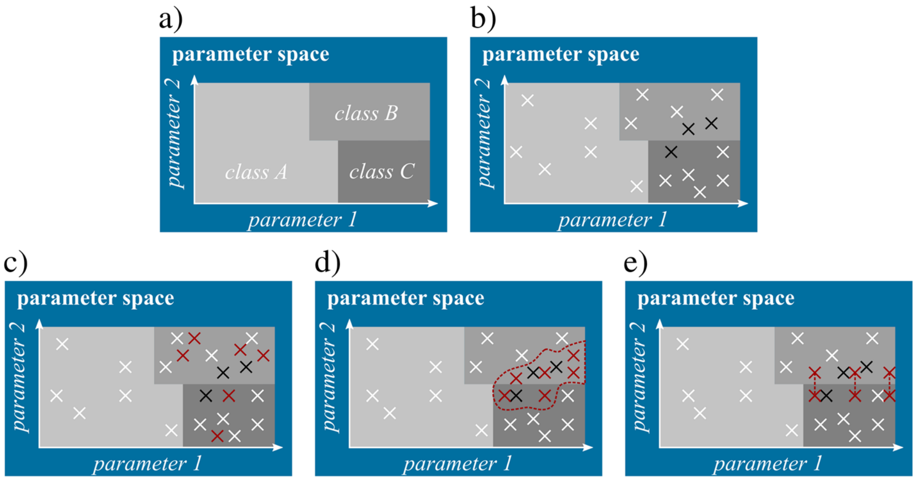

Fig. 5 Different sampling strategies to generate additional training data. **a**) Consider a parametric model with two parameters. Every point in the parameter space (white crosses) corresponds to a concrete instance of a scenario (simulation ready scene). **b**) As a first sampling strategy, class balance can be achieved by generating the same number of instances in the parts of the parameter space that corresponds to every output class. In the example, six scenarios are generated for each class **a**, **b**, and **c**. After training and running initial classification experiments, it becomes clear that the system has difficulties differentiating certain instances of class **b** and **c** (black crosses). c) A straight forward sampling strategy is to generate additional samples in the classes **b** and **c** using uniform random sampling. **d**) A more controlled approach is to use some version of importance sampling to generate additional samples close to known misclassified samples. This requires bookkeeping of the parameters for each training data point. **e**) Additional training data can also be generated in pairs. Hereby, two similar data points are generated that differ in one parameter only but fall in the different classes

Hereby, the two samples of the pair should be identical in all but one parameter, but one sample corresponds to class A, the other sample corresponds to class B. For example, if advertisements signs showing humans are frequently misclassified as pedestrians, a scenario could fix a street scene with respect to street and building layout, lighting conditions, camera position and so on, but replace one advertisement sign with a pedestrian.

If the system stores the parameters used to generate each sample in the training data set, additional importance sampling can be performed (Fig. 5d-e). Starting from the confusion matrix, the sampling algorithm can determine the parameters of individual, incorrectly classified samples. If this information is mapped to the parameter space, regions in the parameter space can be identified that have lower detection rates than others. For example, the system could determine that the detection rates for certain classes are lower in regions of the parameter space where the camera faces westwards and the sun position corresponds to late evening. Additional training data can then be generated by means of importance sampling specifically in these regions.

White box adaptive sampling

Semi-automated approaches for generating training data can also rely on inspecting the used neural network. The easiest way to visualize the inner workings of a neural network is to draw the activations during the forward pass [40]. Activation maps with zero values can help to indicate dead filters, e.g. through high learning rates. Besides activations, it is possible to display the weights, respectively the learned kernels [40]. A well-trained network usually resembles quite smooth kernels without noise. Noisy patterns could indicate low regularization, which means the model is overfitted.

Another option to visualize how a deep neural network works is to track the maximal neuron activation. In particular, it can help to understand what a neuron is looking for in its corresponding receptive field, as shown in [41] and using guided backpropagation as in [42]. One more option to identify where a classification originates from is to plot the probability of a class as function of the position of an occluding object resulting in an occlusion map [43]. Further progress towards the mentioned examples are gradient-based class activation maps [44]. The computed plots are similar to heatmaps and show the contributions of image parts to a classification, which facilitates interpretation compared to originally introduced saliency maps [45].

Impressive work on how neural networks build up their understanding of images has been collated and applied to many example images in [46]. The authors show how optimization can be used for visualizing neural networks and interactions between neurons. Going one step further, possible interfaces that arise when combining interpretability techniques are explored in [47].

All of the above mentioned inspection techniques are potentially useful the make informed decisions about what training data to add for improving the performance of a model. However, a systematic approach with clear procedures how to address this is missing today.

Use-cases

In order to illustrate the Digital Reality concept on concrete examples and to give some evidence of the feasibility of the approach, we now present several Use Cases from different domains.

Use case 'Optical Inspection in Production'

Decreasing production tolerances and increasing quality constraints in manufacturing create a demand for inline inspection. Quality assurance takes place not only on the final product but increasingly at every intermediate step of the production line. On the one hand, this trend is supported by sensors with significantly higher resolution than possible in the past, as well as an increase of computational power of the corresponding signal processing platforms. On the other hand, the demand to quickly reconfigure production lines for new or changing products to the extreme case of individualized products in an Industry 4.0 context makes the automated inspection algorithmically more difficult. How can the inspection algorithm make a decision if a deviation is a defect or an intended configuration if every product is configured individually? In this context, deep neural networks may become an integral part of inline inspection. Rather than learning the product and interpreting every deviation as defect, the system learns typical defects, such as cracks, and can then detect these defects on a wide range of similar, but not identical, products.

Use case 'Use of Synthetic Data for Simulated Autonomous Driving'

In recent years, there has been tremendous progress in the application of deep learning and planning methods for scene understanding and navigation learning of autonomous vehicles [48]. However, the accomplishment of pedestrian safety by means of trusted, verifiable, and efficient methods of artificial intelligence (AI) remains a key challenge for autonomous driving. Critical traffic scenarios with life threatening situations for pedestrians or car passengers are too rare and diverse in comparison to other road situations that can be encountered. Creating such critical scenarios for analysis is ethically impossible. Furthermore, the manual labeling of acquired in-vivo or in-vitro data may be prohibitively expensive. Labeled in-silico images for simulated critical traffic scenes can be automatically generated with appropriate frameworks

[49], driving simulators and several large collections of synthetic ground truth data for benchmarking scene analysis solutions in the context of autonomous driving are available [50]. Deep learning based semantic segmentation with a domain adapted VGG16 network over mixtures of labeled in-silico and unlabeled data can perform considerably better compared against purely using in-vivo data [51].

However, scene understanding alone does not help to solve the problem of collision-free navigation by self-driving cars. Available real or synthetic dataset of critical traffic scenes are either insufficient in quantity and quality, or not freely available. An established standard database does not exist as of today. Therefore, the training and validation of deep reinforcement learning-based methods for safe and smooth navigation of simulated autonomous cars requires the synthesizing of these scenes based on real-world studies of accidents like GIDAS [52, 53].

The Association for Standardization of Automation and Measurement Systems (ASAM) [54] already established description formats for some aspects of the street environment. OpenCRG is a description format for road surfaces, OpenDrive allows the representation of street networks and OpenScenario is a definition language for driving tasks. The language also allows the description of predefined, trajectory based motion paths. However, other aspects such as complex pedestrian and bicyclist behavior and motion are not yet describable by established formats.

Several commercial driving simulators like NVidia Drive Constellation [55], SCANeR [56]. OpenDS [27], and Tronis [24] focus on the simulation of driving scenarios, including the generation of high quality optical images. The solutions mostly lack quantitative sensor-specific simulation because of limitations in the simulation technology and the material representation. For example, available pedestrian models do not include material descriptions for radar simulation, etc.

One interesting question is, what impact a suggested new sensor has on the driving performance of an autonomous vehicle. Particularly, one would like to know whether a certain type of accident could have been avoided if the car had been equipped with a specific lidar, radar, or camera.

Use case 'In-Situ Microscopy of Graphene formation'

Graphene and graphenoid materials have attracted a lot of interest because of their unique mechanical and electrical properties [57]. While defect-free, large area preparation of graphene sheets for practical applications is technically challenging, progress has been made in preparing nanocrystalline graphene (ncg) by pyrolysis of polymer precursors [58]. In addition to the easy fabrication, the main advantage of ncg is, that it can be patterned to achieve different shapes and by varying the polymer precursor and the pyrolysis conditions, the structure and thus the properties of the thin films can be tailored.

However, a better understanding of the graphitization process and the structural evolution during pyrolysis is necessary for a targeted preparation of ncg with defined properties. Low-voltage in-situ TEM techniques have been demonstrated to enable direct imaging of the structural changes of ncg during pyrolysis close to atomic scale resolution [59]. Besides, new reaction pathways, which are strongly influenced by the high defect density in ncg have been identified [33].

Nevertheless, a more complete understanding of the reactions in ncg, which is the interaction of small graphene flakes, is necessary to tailor the pyrolysis conditions. This requires a much more automated tracking of reaction hot spots to enable a statistically meaningful analysis of various reaction sites and to follow the structural changes at the hot spots by high-speed in-situ TEM imaging. Most important, more advanced image analysis methods are required to identify the local atomic arrangement in the image series and to correlate the observed changes with the corresponding molecular modelling.

Use case 'Smart Home and Smart Grid'

Due to climatic change, renewable energy sources and higher energy efficiency are key factors of the economy. More or less all renewable energy sources are very variable so that the stability of energy networks is much more difficult to guarantee. More buffer capacity for energy storage combined with a transformation of energy networks into smart grids are possible ways to cope with the challenge.

Smart grids require intensive load balancing and a regional compensation of energy generation and consumption. Smart homes are important elements of a smart grid. For an optimized load balancing a smart grid should be enabled to control and precisely predict energy consumption and generation by smart homes and its devices such as solar panels, solar heat, large battery packs, combined heat and power generation plants, or washing machines based on individual patterns and profiles. For a good user acceptance, external controls must fit to individual consumer habits and requirements. Machine learning, in particular deep learning is considered essential for an effective extraction and prediction of those patterns, but requires training phases of at least 1 year to cover all "energy periods". Furthermore, it is hard, if not impossible, to train neural networks based on real-world data on energy-related effects of very rare weather conditions such as once-in-a-hundred-years summers or winters, long, unusual rain periods, or large-scale regional damages of solar panels due to thunderstorms. In this respect, synthetic (in-silico) data of simulated energy-related events can help to find appropriate intervention possibilities and to improve the prediction. Simulation might also be useful in generating meaningful data to predict effects of more or better equipped, higher automated, or even autonomous smart homes.

Conclusions

We present the Digital Reality concept as a generic blueprint for the training of Deep Neural Networks using in-silico training data. Aspects of the real world are represented by parametric models, which in turn can be composed to form parametric scenarios. Concrete instances of the scenarios, where all parameters are fixed, are simulation-ready scenes, which can then be used to generate in-silico training data using forward simulations of the measurement process.

The choice of parameter values allows a fine granular level of control over the composition of the training data set. By performing random sampling with a uniform sampling density, class imbalance problems can be avoided. The training can further be improved if training data is generated on-demand following a specific adaptive sampling pattern that can be obtained from careful investigation of the parameter space of the model. Adaptive sampling can start, for example, from investigating the confusion matrix, or by visual inspection of the network. Thus, the composition of the training data set can follow the requirements of the training process rather then been dictated by the arbitrary distribution of an in-vivo data generation process.

The Digital Reality approach offers an elegant and generic solution to most training data problems. It is particularly useful in cases where the in-situ generation of training data involves expensive sample acquisition, if the manual labelling of in-situ data constitutes a prohibitive effort, if training data cannot be obtained in sufficient quantity for ethical reasons, or if the phenomenon in question has been predicted, but not yet been observed. Use cases include the optical inspection in production environments in an Industry 4.0 context, autonomous driving vehicles, scientific applications for example in microscopy, or automated decision making in smart grid applications.

Acknowledgements
Not applicable.

Authors' contributions
TD write the majority of the manuscript and created the figures. All other authors contributed by providing the Use-Cases and by helping with proof-reading the text. PS contributed the Digital Reality concept as core idea of the position paper. All authors read and approved the final manuscript.

Funding
Not applicable.

Availability of data and materials
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany. [2]Saarland University, Saarbrücken, Germany. [3]ASM Pacific, Beuningen, The Netherlands. [4]Karlsruhe Institute of Technology (KIT), Eggenstein-Leopoldshafen, Germany.

Received: 19 December 2018 Accepted: 10 July 2019
Published online: 03 September 2019

References

1. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–44.
2. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature. 1986;323(6088):533–6.
3. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L. ImageNet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition; 2009. p. 248–55.
4. Netzer Y, Wang T. Reading digits in natural images with unsupervised feature learning. Proc Nips. 2011. p. 1–9.
5. Krizhevsky A. Learning multiple layers of features from tiny images. Sci Dep Univ Toronto Tech. 2009. P. 1–60.
6. Gesamtkilometer steigen um 1,4 Prozent. Bundesamt, Kraftfahrts; 2016. p. 1–2.
7. Kinderunfälle im Straßenverkehr. Statistisches Bundesamt (Destatis); 2016. p. 40.
8. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D. Human-level control through deep reinforcement learning. Nature. 2015;518(7540):529–33.
9. Ohn-Bar E, Trivedi MM. Looking at humans in the age of self-driving and highly automated vehicles. IEEE Trans Intell Veh. 2016;1(1):90–104.
10. Statham N. "Use of photogrammetry in video games: a historical overview," Games Cult. A J. Interact. Media; 2018. p. 1–19. ISSN 1555-4120.
11. Siqueira C. The state of photogrammetry in real-time graphics. In: ACM SIGGRAPH 2018 COURSES; 2018.
12. Whelan T, Goesele M, Lovegrove SJ, Straub J, Green S, Szeliski R, Butterfield S, Verma S, Newcombe R, Goesele M, Szeliski R, Butterfield S. Reconstructing scenes with Mirror and glass surfaces. ACM Trans Graph. 2018;37(4):11.
13. Guarnera D, Guarnera GC, Ghosh A, Denk C, Glencross M. BRDF representation and acquisition. Comput Graph Forum. 2016;35(2):625–50.
14. Guo S, Southern R, Chang J, Greer D, Zhang JJ. Adaptive motion synthesis for virtual characters: a survey. Vis Comput. 2015;31(5):497–512.
15. Geijtenbeek T, Pronost N. Interactive character animation using simulated physics: a state-of-the-art review. Comput Graph Forum. 2012;31(8):2492–515.
16. Grochow K, Martin SL, Hertzmann A, Popović Z. Style-based inverse kinematics. ACM Trans Graph. 2004;23(3):522.
17. Feng A, Huang Y, Kallmann M, Shapiro A. An analysis of motion blending techniques; 2012. p. 232–43.
18. Mukai T, Kuriyama S. Geostatistical motion interpolation. ACM Trans Graph. 2005;24(3):1062.
19. Kovar L, Gleicher M, Pighin F. Motion graphs. In: Proceedings of the 29th annual conference on computer graphics and interactive techniques - SIGGRAPH '02; 2002. p. 473.
20. Min J, Chai J. Motion graphs++. ACM Trans Graph. 2012;31(6):1.
21. Li Y, Wang T, Shum H-Y. Motion texture. In: Proceedings of the 29th annual conference on computer graphics and interactive techniques - SIGGRAPH '02; 2002. p. 465.
22. Holden D, Saito J, Komura T. A deep learning framework for character motion synthesis and editing. ACM Trans Graph. 2016;35(4):1–11.
23. Holden D, Komura T, Saito J. Phase-functioned neural networks for character control. ACM Trans Graph. 2017;36(4):1–13.
24. "Tronis," 2018. [Online]. Available: https://www.tronis.de/solution. Accessed 12 Apr 2019.
25. Shah S, Dey D, Lovett C, Kapoor A. AirSim : high-Fidelity visual and physical. 2017; p. 1–14.
26. Dosovitskiy A, Ros G, Codevilla F, Lopez A, Koltun V. CARLA: An Open Urban Driving Simulator; 2017.

27. "OpenDS," 2018. [Online]. Available: https://opends.dfki.de/. [Accessed: 12 Apr 2019].
28. Papadimitriou E, Yannis G, Golias J. A critical assessment of pedestrian behaviour models. Transp Res Part F Traffic Psychol Behav. 2009;12(3):242–55.
29. Feynman RP. There's plenty of room at the bottom. Caltech Eng Sci. 1960; 23(5):22–36.
30. Williams DB, Carter CB. Transmission Electron microscopy. Boston: Springer US; 2009.
31. Ziegler A, Graafsma H, Zhang XF, Frenken JWM, editors. In-situ materials characterization, vol. 193. Berlin: Springer Berlin Heidelberg; 2014.
32. Crozier PA, Hansen TW. In situ and operando transmission electron microscopy of catalytic materials. MRS Bull. 2015;40(01):38–45.
33. Shyam Kumar CKCN, Konrad M, Chakravadhanula VSK, Dehm S, Wang D, Wenzel W, Krupke R. Nanocrystalline graphene at high temperatures: insights into nanoscale processes. ACS Nano. 2018;7(1):2485–94.
34. Madsen J, Liu P, Kling J, Wagner JB, Hansen TW, Winther O, Schiøtz J. A deep learning approach to identify local structures in atomic-resolution transmission Electron microscopy images. Adv Theory Simulations. 2018;1(8): 1800037.
35. Miller MK, Kelly TF, Rajan K, Ringer SP. The future of atom probe tomography. Mater Today. 2012;15(4):158–65.
36. Kelly TF, Miller MK. Invited review article: atom probe tomography. Rev Sci Instrum. 2007;78(3):1–20.
37. Zlateski A, Jaroensri R, Sharma P, Durand F. On the importance of label quality for semantic segmentation. Cvpr. 2018:1479–87.
38. Sun T, Zhang W, Wang Z, Ma L, Jie Z. Image-level to pixel-wise labeling: from theory to practice; 2017. p. 928–34.
39. Sadanandan SK, Ranefall P, Le Guyader S, Wählby C. Automated training of deep convolutional neural networks for cell segmentation. Sci Rep. 2017; 7(1):7860.
40. Yosinski J, Clune J, Nguyen A, Fuchs T, Lipson H. Understanding neural networks through deep visualization. In: Deep learning workshop at 31st international conference on machine learning; 2015.
41. Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: 2014 IEEE conference on computer vision and pattern recognition; 2014. p. 580–7.
42. Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M. Striving for simplicity: the all convolutional net. In: ICLR (workshop track); 2015.
43. Zeiler MD, Fergus R. Visualizing and understanding convolutional networks; 2014. p. 818–33.
44. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: visual explanations from deep networks via gradient-based localization. In: 2017 IEEE international conference on computer vision (ICCV); 2017. p. 618–26.
45. Simonyan K, Vedaldi A, Zisserman A. Deep inside convolutional networks: Visualising image classification models and saliency maps. CoRR. 2013;abs/1312.6.
46. Olah C, Mordvintsev A, Schubert L. Feature Visualization. Distill. 2017;2(11).
47. Olah C, Satyanarayan A, Johnson I, Carter S, Schubert L, Ye K, Mordvintsev A. The building blocks of interpretability. Distill. 2018;3(3).
48. Schwarting W, Alonso-Mora J, Rus D. Planning and decision-making for autonomous vehicles. Annu Rev Control Robot Auton Syst. 2018;1(1):187–210.
49. Haltakov V, Unger C, Ilic S. Framework for generation of synthetic ground truth data for driver assistance applications. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). 2013; 8142 LNCS:323–32.
50. G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes."
51. Poibrenski A, Sprenger J, Müller C. Towards a methodology for training with synthetic data on the example of pedestrian detection in a frame-by-frame semantic. In: 2018 IEEE/ACM 1st Int Work Softw Eng AI Auton Syst; 2018. p. 31–4.
52. "GIDAS - German In-Depth Accident Study," 2018. [Online]. Available: https://www.gidas.org . Accessed 12 Apr 2019.
53. Spitzhüttl DF, Liers DH. Creation of pre-crash simulations in global traffic accident scenarios based on the iGLAD database. Proc 3rd Int Symp Futur Act Saf Technol Towar Zero traffic Accid. 2015:427–33.
54. "ASAM - Standardization for Automotive Development." [Online]. Available: https://www.asam.net/standards/.
55. "NVIDIA DRIVE Constellation." [Online]. Available: https://developer.nvidia.com/drive. [Accessed: 12 Apr 2019].
56. "AVSimulation." [Online]. Available: https://www.avsimulation.fr/solutions/. [Accessed: 12 Apr 2019].
57. Novoselov KS, Fal′ko VI, Colombo L, Gellert PR, Schwab MG, Kim K. A roadmap for graphene. Nature. 2012;490(7419):192–200.
58. Zhang Z, Ge B, Guo Y, Tang D, Wang X, Wang F. Catalyst-free growth of nanocrystalline graphene/graphite patterns from photoresist. Chem Commun. 2013;49(27):2789–91.
59. Shyam Kumar CN, Chakravadhanula VSK, Riaz A, Dehm S, Wang D, Mu X, Flavel B, Krupke R, Kübel C. Understanding the graphitization and growth of free-standing nanocrystalline graphene using in situ transmission electron microscopy. Nanoscale. 2017;9(35):12835–42.

Publisher's Note