Procedural Generation of Videos to Train Deep Action Recognition Networks

César Roberto de Souza[1,3], Adrien Gaidon[2], Yohann Cabon[1], Antonio Manuel López[3]
[1]Computer Vision Group, Xerox Research Center Europe, Meylan, France
[2]Toyota Research Institute, Los Altos, CA, USA
[3]Centre de Visió per Computador, Universitat Autònoma de Barcelona, Bellaterra, Spain
{cesar.desouza, yohann.cabon}@xrce.xerox.com, adrien.gaidon@tri.global, antonio@cvc.uab.es

Abstract

Deep learning for human action recognition in videos is making significant progress, but is slowed down by its dependency on expensive manual labeling of large video collections. In this work, we investigate the generation of synthetic training data for action recognition, as it has recently shown promising results for a variety of other computer vision tasks. We propose an interpretable parametric generative model of human action videos that relies on procedural generation and other computer graphics techniques of modern game engines. We generate a diverse, realistic, and physically plausible dataset of human action videos, called PHAV for "Procedural Human Action Videos". It contains a total of $39,982$ *videos, with more than* $1,000$ *examples for each action of* 35 *categories. Our approach is not limited to existing motion capture sequences, and we procedurally define* 14 *synthetic actions. We introduce a deep multi-task representation learning architecture to mix synthetic and real videos, even if the action categories differ. Our experiments on the UCF101 and HMDB51 benchmarks suggest that combining our large set of synthetic videos with small real-world datasets can boost recognition performance, significantly outperforming fine-tuning state-of-the-art unsupervised generative models of videos.*

1. Introduction

Understanding human behavior in videos is a key problem in computer vision. Accurate representations of both appearance and motion require either carefully handcrafting features with prior knowledge (*e.g.,* the dense trajectories of [71]) or end-to-end deep learning of high capacity models with a large amount of labeled data (*e.g.,* the two-stream network of [53]). These two families of methods have complementary strengths and weaknesses, and they often need to be combined to achieve state-of-the-art action recognition performance [74, 9]. Nevertheless, deep networks have the potential to significantly improve their accuracy based on training data. Hence, they are becoming the de-facto standard for recognition problems where it is possible to collect large labeled training sets, often by crowd-sourcing manual annotations (*e.g.,* ImageNet [10], MS-COCO [32]). However, manual labeling is costly, time-consuming, error-prone, raises privacy concerns, and requires massive human intervention for every new task. This is often impractical, especially for videos, or even unfeasible for ground truth modalities like optical flow or depth.

Figure 1: Procedurally generated human action videos (clockwise): push, kick ball, walking hug, car hit.

Using realistic synthetic data generated from virtual worlds alleviates these issues. Thanks to modern modeling, rendering, and simulation software, virtual worlds allow for the efficient generation of vast amounts of controlled and algorithmically labeled data, including for modalities that cannot be labeled by a human. This approach has recently shown great promise for deep learning across a breadth of computer vision problems, including optical flow [37], depth estimation [32], object detection [34, 65, 78, 59, 42], pose and viewpoint estimation [52, 41, 57], tracking [15], and semantic segmentation [23, 49, 48].

Work done while A. Gaidon was at Xerox Research Centre Europe.

In this work, we investigate *procedural generation of synthetic human action videos* from virtual worlds in order to train deep action recognition models. This is an open problem with formidable technical challenges, as it requires a full generative model of videos with realistic appearance and motion statistics conditioned on specific action categories. Our experiments suggest that our procedurally generated action videos can complement scarce real-world data. We report significant performance gains on target real-world categories although they differ from the actions present in our synthetic training videos.

Our first contribution is a *parametric generative model of human action videos* relying on physics, scene composition rules, and procedural animation techniques like "ragdoll physics" that provide a much stronger prior than just viewing videos as tensors or sequences of frames. We show how to procedurally generate physically plausible variations of different types of action categories obtained by MOCAP datasets, animation blending, physics-based navigation, or entirely from scratch using programmatically defined behaviors. We use naturalistic actor-centric randomized camera paths to film the generated actions with care for physical interactions of the camera. Furthermore, our manually designed generative model has *interpretable parameters* that allow to either randomly sample or precisely control discrete and continuous scene (weather, lighting, environment, time of day, etc), actor, and action variations to generate large amounts of diverse, physically plausible, and realistic human action videos.

Our second contribution is a quantitative experimental validation using a modern and accessible game engine (Unity®Pro) to synthesize a labeled dataset of $39,982$ videos, corresponding to more than $1,000$ examples for each of 35 action categories: 21 grounded in MOCAP data, and 14 entirely synthetic ones defined procedurally. Our dataset, called *PHAV* for "Procedural Human Action Videos" (*cf.* Figure 1 for example frames), is publicly available for download[1]. Our procedural generative model took approximately 2 months of 2 engineers to be programmed and our PHAV dataset 3 days to be generated using 4 gaming GPUs. We investigate the use of this data in conjunction with the standard UCF101 [55] and HMDB51 [29] action recognition benchmarks. To allow for generic use, and as predefined procedural action categories may differ from unknown a priori real-world target ones, we propose a multi-task learning architecture based on the recent Temporal Segment Network [75] (TSN). We call our model *Cool-TSN* (*cf.* Figure 6) in reference to the "cool world" of [64], as we mix both synthetic and real samples at the mini-batch level during training. Our experiments show that the generation of our synthetic human action videos can significantly improve action recognition accuracy, especially with small real-world training sets, in spite of differences in appearance, motion, and action categories. Moreover, we outperform other state-of-the-art generative video models [70] when combined with the same number of real-world training examples.

The rest of the paper is organized as follows. Section 2 presents a brief review of related works. In Section 3, we present our parametric generative model, and how we use it to procedurally generate our PHAV dataset. Section 4 presents our cool-TSN deep learning algorithm for action recognition. We report our quantitative experiments in Section 5 measuring the usefulness of PHAV. Finally, conclusions are drawn in Section 6.

2. Related work

Synthetic data has been used to train visual models for object detection and recognition, pose estimation, and indoor scene understanding [34, 65, 78, 77, 52, 47, 44, 50, 1, 7, 41, 42, 22, 24, 35, 58, 57, 23]. [21] used a virtual racing circuit to generate different types of pixel-wise ground truth (depth, optical flow and class labels). [49, 48] relied on game technology to train deep semantic segmentation networks, while [15] used it for multi-object tracking, [51] for depth estimation from RGB, and [54] for place recognition. Several works use synthetic scenarios to evaluate the performance of different feature descriptors [27, 2, 68, 67, 69] and to train and test optical and/or scene flow estimation methods [38, 4, 40, 37], stereo algorithms [20], or trackers [61, 15]. They have also been used for learning artificial behaviors such as playing Atari games [39], imitating players in shooter games [33], end-to-end driving/navigating [6, 80], learning common sense [66, 81] or physical intuitions [31]. Finally, virtual worlds have also been explored from an animator's perspective. Works in computer graphics have investigated producing animations from sketches [19], using physical-based models to add motion to sketch-based animations [18], and creating constrained camera-paths [16]. However, due to the formidable complexity of realistic animation, video generation, and scene understanding, these approaches focus on basic controlled game environments, motions, and action spaces.

To the best of our knowledge, ours is the first work to investigate virtual worlds and game engines to generate synthetic training videos for action recognition. Although some of the aforementioned related works rely on virtual characters, their actions are not the focus, not procedurally generated, and often reduced to just walking.

The related work of [36] uses MOCAP data to induce realistic motion in an "abstract armature" placed in an empty synthetic environment, generating $2,000$ short 3-second clips at 320x240 and 30FPS. From these non-photo-realistic clips, handcrafted motion features are selected as relevant and later used to learn action recognition models for 11 ac-

[1] Data and tools available in http://adas.cvc.uab.es/phav/

tions in real-world videos. Our approach *does not just replay MOCAP, but procedurally generates new action categories* – including interactions between persons, objects and the environment – as well as random *physically plausible* variations. Moreover, we jointly generate and learn deep representations of both action appearance and motion thanks to our realistic synthetic data, and our multi-task learning formulation to combine real and synthetic data.

A recent alternative to our procedural generative model that also does not require manual video labeling is the unsupervised Video Generative Adversarial Network (VGAN) of [70]. Instead of leveraging prior structural knowledge about physics and human actions, the authors view videos as tensors of pixel values and learn a two-stream GAN on $5,000$ hours of unlabeled Flickr videos. This method focuses on tiny videos and capturing scene motion assuming a stationary camera. This architecture can be used for action recognition in videos when complemented with prediction layers fine-tuned on labeled videos. Compared to this approach, our proposal allows to work with any state-of-the-art discriminative architecture, as video generation and action recognition are decoupled steps. We can, therefore, benefit from a strong ImageNet initialization for both appearance and motion streams as in [75]. Moreover, we can decide what specific actions/scenarios/camera-motions to generate, enforcing diversity thanks to our interpretable parametrization. For these reasons, we show in Section 5 that, given the same amount of labeled videos, our model achieves nearly two times the performance of the unsupervised features shown in [70].

3. PHAV: Procedural Human Action Videos

In this section we introduce our interpretable parametric generative model of videos depicting particular human actions, and how we use it to generate our PHAV dataset.

3.1. Action scene composition

In order to generate a human action video, we place a *protagonist* performing an *action* in an *environment*, under particular *weather conditions* at a specific *period* of the day. There can be one or more *background actors* in the scene, as well as one or more *supporting characters*. We film the virtual scene using a parametric *camera behavior*.

The protagonist is the main human model performing the action. For actions involving two or more people, one is chosen to be the protagonist. Background actors can freely walk in the current virtual environment, while supporting characters are actors with a secondary role necessary to complete an action, *e.g.,* hold hands.

The action is a human motion belonging to a predefined semantic category originated from one or more motion data sources (described in section 3.3), including pre-determined motions from a MOCAP dataset, or programmatic actions

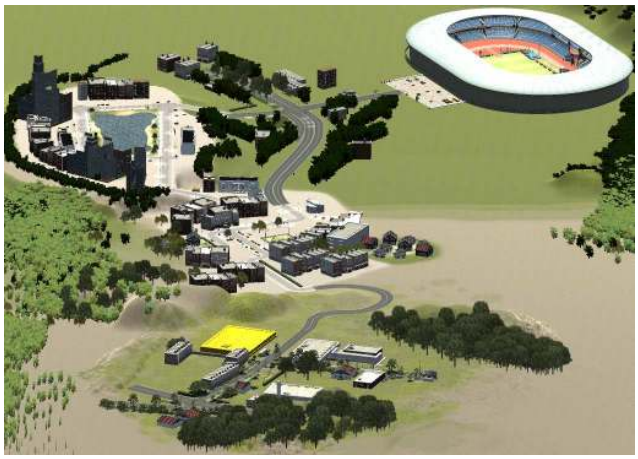

Figure 2: Orthographic view of different world regions.

defined using procedural animation techniques [11, 63], in particular ragdoll physics. In addition, we use these techniques to sample physically-plausible motion variations (described in section 3.4) to increase diversity.

The environment refers to a region in the virtual world (*cf.* Figure 2), which consists of large urban areas, natural environments (*e.g.,* forests, lakes, and parks), indoor scenes, and sports grounds (*e.g.,* a stadium). Each of these environments may contain moving or static background pedestrians or objects – *e.g.,* cars, chairs – with which humans can physically interact, voluntarily or not. The outdoor weather in the virtual world can be rainy, overcast, clear, or foggy. The period of the day can be dawn, day, dusk, or night.

Similar to [15, 49], we use a library of pre-made 3D models obtained from the Unity Asset Store, which includes artist-designed human, object, and texture models, as well as semi-automatically created realistic environments (*e.g.,* selected scenes from the VKITTI dataset [15]).

3.2. Camera

We use a physics-based camera which we call the Kite camera (*cf.* Figure 3) to track the protagonist in a scene. This physics-aware camera is governed by a rigid body attached by a spring to a target position that is, in turn, attached to the protagonist by another spring. By randomly sampling different parameters for the drag and weight of the rigid bodies, as well as elasticity and length of the springs, we can achieve cameras with a wide range of shot types, 3D transformations, and tracking behaviors, such as following the actor, following the actor with a delay, or stationary. Another parameter controls the direction and strength of an initial impulse that starts moving the camera in a random direction. With different rigid body parameters, this impulse can cause our camera to simulate a handheld camera, move in a circular trajectory, or freely bounce around in the scene while filming the attached protagonist.

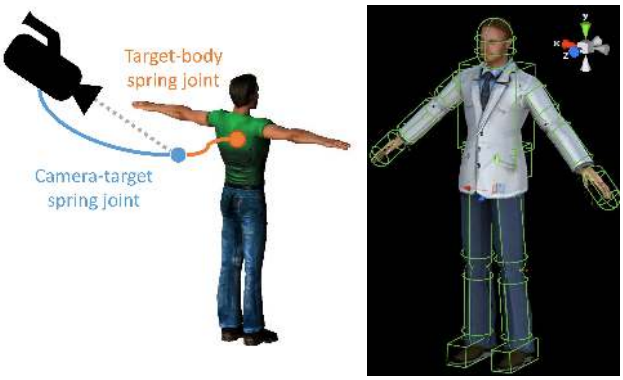

Figure 3: Left: schematic representation of our Kite camera. Right: human ragdoll configuration with 15 muscles.

3.3. Actions

Our approach relies on two main existing data sources for basic human animations. First, we use the CMU MOCAP database [5], which contains 2605 sequences of 144 subjects divided in 6 broad categories, 23 subcategories and further described with a short text. We leverage relevant motions from this dataset to be used as a motion source for our procedural generation based on a simple filtering of their textual motion descriptions. Second, we use a large amount of hand-designed realistic motions made by animation artists and available on the Unity Asset Store.

The key insight of our approach is that *these sources need not necessarily contain motions from predetermined action categories of interest, neither synthetic nor target real-world actions (unknown a priori)*. Instead, we propose to use these sources to form a *library of atomic motions* to procedurally generate realistic action categories. We consider atomic motions as individual movements of a limb in a larger animation sequence. For example, atomic motions in a "walk" animation include movements such as rising a left leg, rising a right leg, and pendular arm movements. Creating a library of atomic motions enables us to later recombine those atomic actions into new higher-level animation sequences, e.g., "hop" or "stagger".

Our PHAV dataset contains 35 different action classes (*cf.* Table 1), including 21 simple categories present in HMDB51 and composed directly of some of the aforementioned atomic motions. In addition to these actions, we programmatically define 10 action classes involving a single actor and 4 action classes involving two person interactions. We create these new synthetic actions by taking atomic motions as a base and using procedural animation techniques like blending and ragdoll physics (*cf.* Section 3.4) to compose them in a physically plausible manner according to simple rules defining each action, such as tying hands together (*e.g.,* "walk hold hands"), disabling one or more muscles (*e.g.,* "crawl", "limp"), or colliding the protagonist against obstacles (*e.g.,* "car hit", "bump into each other").

Type	#	Actions
sub-HMDB	21	brush hair, catch, clap, climb stairs, golf, jump, kick ball, push, pick, pour, pull up, run, shoot ball, shoot bow, shoot gun, sit, stand, swing baseball, throw, walk, wave
One-person synthetic	10	car hit, crawl, dive floor, flee, hop, leg split, limp, moonwalk, stagger, surrender
Two-people synthetic	4	walking hug, walk hold hands, walk the line, bump into each other

Table 1: Actions included in our PHAV dataset.

3.4. Physically plausible motion variations

We now describe procedural animation techniques [11, 63] to randomly generate large amounts of physically plausible and diverse action videos, far beyond from what can be achieved by simply replaying source atomic motions.

Ragdoll physics. A key component of our work is the use of ragdoll physics. Ragdoll physics are limited real-time physical simulations that can be used to animate a model (such as a human model) while respecting basic physics properties such as connected joint limits, angular limits, weight and strength. We consider ragdolls with 15 movable body parts (referenced here as muscles), as illustrated in Figure 3. For each action, we separate those 15 muscles into two disjoint groups: those that are strictly necessary for performing the action, and those that are complementary (altering their movement should not interfere with the semantics of the currently considered action). The presence of the ragdoll allows us to introduce variations of different nature in the generated samples. The other modes of variability generation described in this section will assume that the physical plausibility of the models is being kept by the use of ragdoll physics. We use RootMotion's PuppetMaster[2] for implementing and controlling human ragdolls in Unity®Pro.

Random perturbations. Inspired by [45], we create variations of a given motion by adding random perturbations to muscles that should not alter the semantic category of the action being performed. Those perturbations are implemented by adding a rigid body to a random subset of the complementary muscles. Those bodies are set to orbit around the muscle's position in the original animation skeleton, drifting the movement of the puppet's muscle to its own position in a periodic oscillating movement. More detailed references on how to implement variations of this type can be found in [45, 11, 46, 63] and references therein.

Muscle weakening. We vary the strength of the avatar performing the action. By reducing its strength, the actor performs an action with seemingly more difficulty.

[2] http://root-motion.com

Figure 4: Example generation failure cases. First row: physics violations (passing through a wall). Second row: over-constrained joints and unintended variations.

Action blending. Similarly to modern video games, we use a blended ragdoll technique to constrain the output of a pre-made animation to physically-plausible motions. In action blending, we randomly sample a different motion sequence (coming either from the same or from a different class) and replace the movements of current complementary muscles with those from this new action. We limit the number of blended actions in PHAV to be at most two.

Objects. The last physics-based source of variation is the use of objects. First, we manually annotated a subset of the MOCAP actions marking the instants in time where the actor started or ended the manipulation of an object. Second, we use inverse kinematics to generate plausible programmatic interactions. For reproducibility, our annotated subset of MOCAP actions, as well as the code for interacting with objects for particular actions will be available upon request.

Failure cases. Although our approach uses physics-based procedural animation techniques, unsupervised generation of large amounts of random variations with a focus on diversity inevitably causes edge cases where physical models fail. This results in glitches reminiscent of typical video game bugs (*cf.* Figure 4). Using a random 1% sample of our dataset, we manually estimated that this corresponds to less than 10% of the videos generated. Although this could be improved, our experiments in Section 5 show that this noise does not prevent us from improving the training of deep action recognition networks using this data.

Extension to complex activities. Using ragdoll physics and a large enough library of atomic actions, it is possible to create complex actions by hierarchical composition. For instance, our "Car Hit" action is procedurally defined by composing atomic actions of a person (walking and/or doing other activities) with those of a car (entering in a collision with the person), followed by the person falling in a physically plausible fashion. However, while atomic actions have been validated as an effective decomposition for the recognition of potentially complex actions [14], we have not studied how this approach would scale with the complexity of the actions, notably due to the combinatorial nature of complex events. We leave this as future work.

Parameter	#	Possible values
Human Model (H)	20	models designed by artists
Environment (E)	7	simple, urban, green, middle, lake, stadium, house interior
Weather (W)	4	clear, overcast, rain, fog
Period of day (D)	4	night, dawn, day, dusk
Variation (V)	5	$\varnothing$, muscle perturbation and weakening, action blending, objects

Table 2: Overview of key random variables of our parametric generative model of human action videos.

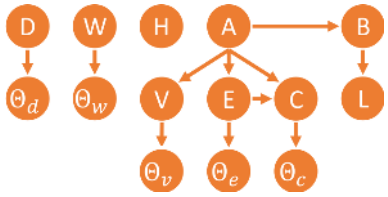

Figure 5: Simplified graphical model for our generator.

3.5. Interpretable parametric generative model

We define a human action video as a random variable $X = \langle H, A, L, B, V, C, E, D, W \rangle$, where H is a human model, A an action category, L a video length, B a set of basic motions (from MOCAP, manual design, or programmed), V a set of motion variations, C a camera, E an environment, D a period of the day, W a weather condition, and possible values for those parameters are shown in Table 2. Our generative model (*cf.* Figure 5, more details in supplementary material) for an action video X is given by:

$$\begin{aligned} P(X) = & P(H)\ P(A)\ P(L \mid B)\ P(B \mid A) \\ & P(\Theta_v \mid V)\ P(V \mid A)\ P(\Theta_e \mid E)\ P(E \mid A) \\ & P(\Theta_c \mid C)\ P(C \mid A, E) \\ & P(\Theta_d \mid D)\ P(D)\ P(\Theta_w \mid W)\ P(W) \end{aligned} \quad (1)$$

where Θ_w is a random variable (r.v.) on weather-specific parameters (*e.g.,* intensity of rain, clouds, fog), Θ_c is a r.v. on camera-specific parameters (*e.g.,* weights and stiffness for Kite camera springs), Θ_e is a r.v. on environment-specific parameters (*e.g.,* current waypoint, waypoint locations, background pedestrian starting points and destinations), Θ_d is a r.v. on period-specific parameters (*e.g.,* amount of sunlight, sun orientation), and Θ_v is a r.v. on variation-specific parameters (*e.g.,* strength of each muscle, strength of perturbations, blending muscles). The probability functions associated with categorical variables (*e.g.,* A) can be either uniform, or configured manually to use pre-determined weights. Similarly, probability distributions associated with continuous values (*e.g.,* Θ_c) are either set using a uniform distribution with finite support, or using triangular distributions with pre-determined support and most likely value. All values used are disclosed in the supplementary material.

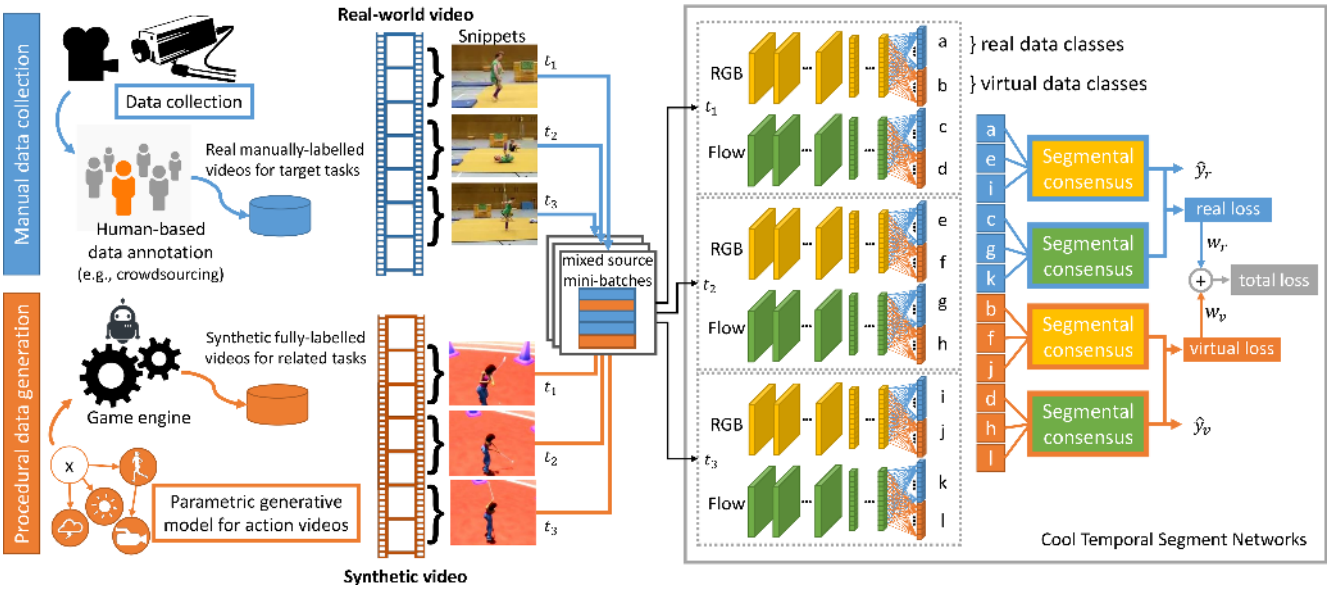

Figure 6: Our "Cool-TSN" deep multi-task learning architecture for end-to-end action recognition in videos.

3.6. PHAV generation details

We generate videos with lengths between 1 and 10 seconds, at 30FPS, and resolution of 340x256 pixels. We use anti-aliasing, motion blur, and standard photo-realistic cinematic effects. We have generated 55 hours of videos, with approximately 6M frames and at least 1,000 videos per action category. Our parametric model can generate fully-annotated action videos (including depth, flow, semantic segmentation, and human pose ground-truths) at 3.6 FPS using one consumer-grade gaming GPU (NVIDIA GTX 1070). In contrast, the average annotation time for data-annotation methods such as [48, 8, 3] are significantly below 0.5 FPS. While those works deal with semantic segmentation where the cost of annotation is higher, we can generate all modalities for roughly the same cost as RGB.

4. Cool Temporal Segment Networks

We propose to demonstrate the usefulness of our PHAV dataset via deep multi-task representation learning. Our main goal is to learn an end-to-end action recognition model for real-world target categories by combining a few examples of labeled real-world videos with a large number of procedurally generated videos for different surrogate categories. Our hypothesis is that, although the synthetic examples differ in statistics and tasks, their realism, quantity, and diversity can act as a strong prior and regularizer against overfitting, towards data-efficient representation learning that can operate with few manually labeled real videos. Figure 6 depicts our learning algorithm inspired by [53], but adapted for the recent state-of-the-art Temporal Segment Networks (TSN) of [75] with "cool worlds" [64], *i.e.* mixing real and virtual data during training.

4.1. Temporal Segment Networks

The recent TSN architecture of [75] improves significantly on the original two-stream architecture of [53]. It processes both RGB frames and stacked optical flow frames using a deeper Inception architecture [60] with Batch Normalization [25] and DropOut [56]. Although it still requires massive labeled training sets, this architecture is more data efficient, and therefore more suitable for action recognition in videos. In particular, [75] shows that both the appearance and motion streams of TSNs can benefit from a strong initialization on ImageNet, which is one of the main factors responsible for the high recognition accuracy of TSN.

Another improvement of TSN is the explicit use of long-range temporal structure by jointly processing random short snippets from a uniform temporal subdivision of a video. TSN computes separate predictions for K different temporal segments of a video. These partial predictions are then condensed into a video-level decision using a segmental consensus function $\mathbf{G}$. We use the same parameters as [75]: a number of segments $K = 3$, and the consensus function: $\mathbf{G} = \frac{1}{K}\sum_{k=1}^{K}\mathcal{F}(T_k; W)$, where $\mathcal{F}(T_k; W)$ is a function representing a CNN architecture with weight parameters W operating on short snippet T_k from video segment k.

4.2. Multi-task learning in a Cool World

As illustrated in Figure 6, the main differences we introduce with our "Cool-TSN" architecture are at both ends of the training procedure: (i) the mini-batch generation, and (ii) the multi-task prediction and loss layers.

Cool mixed-source mini-batches. Inspired by [64, 49], we build mini-batches containing a mix of real-world videos and synthetic ones. Following [75], we build minibatches

of 256 videos divided in blocks of 32 dispatched across 8 GPUs for efficient parallel training using MPI[3]. Each 32 block contains 10 random synthetic videos and 22 real videos in all our experiments, as we observed it roughly balances the contribution of the different losses during backpropagation. Note that although we could use our generated ground truth flow for the PHAV samples in the motion stream, we use the same fast optical flow estimation algorithm as [75] (TVL1 [79]) for all samples in order to fairly estimate the usefulness of our generated videos.

Multi-task prediction and loss layers. Starting from the last feature layer of each stream, we create two separate computation paths, one for target classes from the real-world dataset, and another for surrogate categories from the virtual world. Each path consists of its own segmental consensus, fully-connected prediction, and softmax loss layers. As a result, we obtain the following multi-task loss:

$$\mathcal{L}(y, \mathbf{G}) = \sum_{z \in \{real, virtual\}} \delta_{\{y \in C_z\}} w_z \mathcal{L}_z(y, \mathbf{G}) \qquad (2)$$

$$\mathcal{L}_z(y, \mathbf{G}) = -\sum_{i \in C_z} y_i \left(G_i - \log \sum_{j \in C_z} \exp G_j \right) \qquad (3)$$

where z indexes the source dataset (real or virtual) of the video, w_z is a loss weight (we use the relative proportion of z in the mini-batch), C_z denotes the set of action categories for dataset z, and $\delta_{\{y \in C_z\}}$ is the indicator function that returns one when label y belongs to C_z and zero otherwise. We use standard SGD with backpropagation to minimize that objective, and as every mini-batch contains both real and virtual samples, every iteration is guaranteed to update both shared feature layers and separate prediction layers in a common descent direction. We discuss the setting of the learning hyper-parameters (*e.g.,* learning rate, iterations) in the following experimental section.

5. Experiments

In this section, we detail our action recognition experiments on widely used real-world video benchmarks.

5.1. Real world action recognition datasets

We consider the two most widely used real-world public benchmarks for human action recognition in videos. The **HMDB-51** [29] dataset contains 6,849 fixed resolution videos clips divided between 51 action categories. The evaluation metric for this dataset is the average accuracy over three data splits. The **UCF-101** [55, 26] dataset contains 13,320 video clips divided among 101 action classes. Like HMDB, its standard evaluation metric is the average mean accuracy over three data splits. Similarly to UCF-101 and HMDB-51, we generate three random splits on our PHAV dataset, with 80% for training and the rest for testing, and report average accuracy when evaluating on PHAV.

[3] https://github.com/yjxiong/temporal-segment-networks

Target	Model	Spatial	Temporal	Full
PHAV	TSN	65.9	81.5	82.3
UCF-101	[75]	85.1	89.7	94.0
UCF-101	TSN	84.2	89.3	93.6
UCF-101	TSN-FT	86.1	89.7	94.1
UCF-101	Cool-TSN	86.3	89.9	**94.2**
HMDB-51	[75]	51.0	64.2	68.5
HMDB-51	TSN	50.4	61.2	66.6
HMDB-51	TSN-FT	51.0	63.0	68.9
HMDB-51	Cool-TSN	53.0	63.9	**69.5**

Table 3: Impact of our PHAV dataset using Cool-TSN. [75] uses TSN with cross-modality training.

5.2. Temporal Segment Networks

In our first experiments (*cf.* Table 3), we reproduce the performance of the original TSN in UCF-101 and HMDB-51 using the same learning parameters as in [75]. For simplicity, we use neither cross-modality pre-training nor a third warped optical flow stream like [75], as their impact on TSN is limited with respect to the substantial increase in training time and computational complexity, degrading only by -1.9% on HMDB-51, and -0.4% on UCF-101.

We also estimate performance on PHAV separately, and fine-tune PHAV networks on target datasets. Training and testing on PHAV yields an average accuracy of 82.3%, which is between that of HMDB-51 and UCF-101. This sanity check confirms that, just like real-world videos, our synthetic videos contain both appearance and motion patterns that can be captured by TSN to discriminate between our different procedural categories. We use this network to perform fine-tuning experiments (TSN-FT), using its weights as a starting point for training TSN on UCF101 and HMDB51 instead of initializing directly from ImageNet as in [75]. We discuss learning parameters and results below.

5.3. Cool Temporal Segment Networks

In Table 3 we also report results of our Cool-TSN multi-task representation learning, which additionally uses PHAV to train UCF-101 and HMDB-51 models. We stop training after $3,000$ iterations for RGB streams and $20,000$ for flow streams, all other parameters as in [75]. Our results suggest that leveraging PHAV through either Cool-TSN or TSN-FT yields recognition improvements for all modalities in all datasets, with advantages in using Cool-TSN especially for the smaller HMDB-51. This provides quantitative experimental evidence supporting our claim that procedural generation of synthetic human action videos can indeed act as a strong prior (TSN-FT) and regularizer (Cool-TSN) when learning deep action recognition networks.

We further validate our hypothesis by investigating the impact of reducing the number of real world training videos (and iterations), with or without the use of PHAV. Our re-

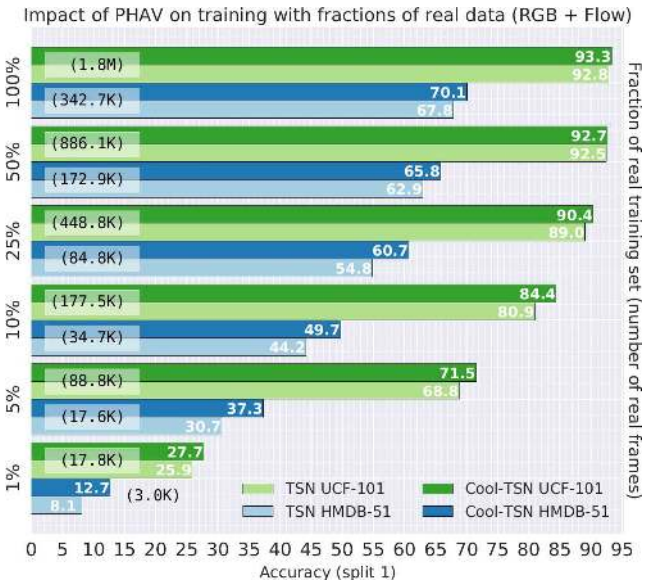

Figure 7: Impact of using subsets of the real world training videos (split 1), with PHAV (Cool-TSN) or without (TSN).

sults reported in Figure 7 confirms that reducing training data from the target dataset impacts more severely TSN than Cool-TSN. HMDB displays the largest gaps. We partially attribute this to the smaller size of HMDB and also because some categories of PHAV overlap with some categories of HMDB. Our results show that it is possible to replace half of HMDB with procedural videos and still obtain comparable performance to using the full dataset (65.8 vs. 67.8). In a similar way, and although actions differ more, we show that reducing UCF-101 to a quarter of its original training set still yields a Cool-TSN model that rivals the state-of-the-art [76, 53, 74]. This shows that our procedural generative model of videos can indeed be used to augment different small real-world training sets and obtain better recognition accuracy at a lower cost in terms of manual labor.

5.4. Comparison with the state of the art

In this section, we compare our model with the state of the art in action recognition (Table 4). We separate the current state of the art into works that use one or multiple sources of training data (such as by pre-training, multi-task learning or model transfer). We note that all works that use multiple sources can potentially benefit from PHAV without any modifications. Our results indicate that our methods are competitive with the state of the art, including methods that use much more manually labeled training data like the Sports-1M dataset [28]. Our approach also leads to better performance than the current best generative video model VGAN [70] on UCF101, for the same amount of manually labeled target real-world videos. We note that while VGAN's more general task is quite challenging and different from ours, [70] has also explored VGAN as a way to learn unsupervised representations useful for action recognition, thus enabling our comparison.

	Method	UCF-101 %mAcc	HMDB-51 %mAcc
ONE SOURCE	iDT+FV [73]	84.8	57.2
	iDT+StackFV [43]	-	66.8
	iDT+SFV+STP [72]	86.0	60.1
	iDT+MIFS [30]	89.1	65.1
	VideoDarwin [13]	-	63.7
MULTIPLE SOURCES	2S-CNN [53]	88.0	59.4
	TDD [74]	90.3	63.2
	TDD+iDT [74]	91.5	65.9
	C3D+iDT[62]	90.4	-
	Actions∼Trans [76]	92.0	62.0
	2S-Fusion [12]	93.5	69.2
	Hybrid-iDT [9]	92.5	70.4
	TSN-3M [75]	94.2	69.4
	VGAN [70]	52.1	-
	Cool-TSN	94.2	69.5

Table 4: Comparison against the state of the art.

6. Conclusion

In this work, we have introduced PHAV, a large synthetic dataset for action recognition based on a procedural generative model of videos. Although our model does not learn video representations like VGAN, it can generate many diverse training videos thanks to its grounding in strong prior physical knowledge about scenes, objects, lighting, motions, and humans.

We provide quantitative evidence that our procedurally generated videos can be used as a simple drop-in complement to small training sets of manually labeled real-world videos. Importantly, we show that we do not need to generate training videos for particular target categories fixed a priori. Instead, surrogate categories defined procedurally enable efficient multi-task representation learning for potentially unrelated target actions that might have only few real-world training examples.

Our approach combines standard techniques from Computer Graphics (procedural generation) with state-of-the-art deep learning for action recognition. This opens interesting new perspectives for video modeling and understanding, including action recognition models that can leverage algorithmic ground truth generation for optical flow, depth, semantic segmentation, or pose, or the combination with unsupervised generative models like VGAN [70] for dynamic background generation, domain adaptation, or real-to-virtual world style transfer [17].

Acknowledgements. Antonio M. Lopez is supported by the Spanish MICINN project TRA2014-57088-C2-1-R, by the Secretaria d'Universitats i Recerca del Departament d'Economia i Coneixement de la Generalitat de Catalunya (2014-SGR-1506), and the CERCA Programme / Generalitat de Catalunya.

References

[1] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3D chairs: exemplar part-based 2d-3d alignment using a large dataset of CAD models. In *CVPR*, 2014. 2

[2] M. Aubry and B. Russell. Understanding deep features with computer-generated imagery. In *ICCV*, 2015. 2

[3] G. Brostow, J. Fauqueur, and R. Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(20):88–89, 2009. 6

[4] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. *ECCV*, pages 611–625, 2012. 2

[5] Carnegie Mellon Graphics Lab. Carnegie Mellon University Motion Capture Database, 2016. 4

[6] C. Chen, A. Seff, A. Kornhauser, and J. Xiao. DeepDriving: Learning affordance for direct perception in autonomous driving. In *ICCV*, 2015. 2

[7] L.-C. Chen, S. Fidler, and R. Yuille, Alan L. Urtasun. Beat the MTurkers: Automatic image labeling from weak 3D supervision. In *CVPR*, 2014. 2

[8] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 6

[9] C. R. de Souza, A. Gaidon, E. Vig, and A. M. López. Sympathy for the Details: Dense Trajectories and Hybrid Classification Architectures for Action Recognition. In *ECCV*, pages 1–27, 2016. 1, 8

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1

[11] A. Egges, A. Kamphuis, and M. Overmars, editors. *Motion in Games*, volume 5277 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. 3, 4

[12] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional Two-Stream Network Fusion for Video Action Recognition. In *CVPR*, 2016. 8

[13] B. Fernando, E. Gavves, M. Oramas, A. Ghodrati, T. Tuytelaars, and K. U. Leuven. Modeling video evolution for action recognition. In *CVPR*, 2015. 8

[14] A. Gaidon, Z. Harchaoui, and C. Schmid. Temporal localization of actions with actoms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2782–95, nov 2013. 5

[15] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual Worlds as Proxy for Multi-Object Tracking Analysis. *CVPR*, pages 4340–4349, 2016. 1, 2, 3

[16] Q. Galvane, M. Christie, C. Lino, and R. Ronfard. Camera-on-rails : Automated Computation of Constrained Camera Paths. *ACM SIGGRAPH Conference on Motion in Games*, pages 151–157, 2015. 2

[17] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. *CVPR*, pages 2414–2423, 2016. 8

[18] M. Guay, R. Ronfard, M. Gleicher, and M.-P. Cani. Adding dynamics to sketch-based character animations. *Sketch-Based Interfaces and Modeling (SBIM) 2015*, 2015. 2

[19] M. Guay, R. Ronfard, M. Gleicher, and M.-P. Cani. Space-time sketching of character animation. *ACM Trans. Graph.*, 34(4):118:1–118:10, July 2015. 2

[20] R. Haeusler and D. Kondermann. Synthesizing real world stereo challenges. In *Proceedings of the German Conference on Pattern Recognition*, 2013. 2

[21] H. Haltakov, C. Unger, and S. Ilic. Framework for generation of synthetic ground truth data for driver assistance applications. In *Proceedings of the German Conference on Pattern Recognition*, 2013. 2

[22] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla. SynthCam3D: Semantic understanding with synthetic indoor scenes. *CoRR*, arXiv:1505.00171, 2015. 2

[23] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla. Understanding real world indoor scenes with synthetic data. In *CVPR*, 2016. 1, 2

[24] H. Hattori, V. Naresh Boddeti, K. M. Kitani, and T. Kanade. Learning scene-specific pedestrian detectors without real data. In *CVPR*, 2015. 2

[25] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 6

[26] Y.-G. Jiang, J. Liu, a. Roshan Zamir, I. Laptev, M. Piccardi, M. Shah, and R. Sukthankar. THUMOS Challenge: Action Recognition with a Large Number of Classes, 2013. 7

[27] B. Kaneva, A. Torralba, and W. Freeman. Evaluation of image features using a photorealistic virtual world. In *ICCV*, 2011. 2

[28] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 8

[29] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *ICCV*, 2011. 2, 7

[30] Z. Lan, M. Lin, X. Li, A. G. Hauptmann, and B. Raj. Beyond gaussian pyramid : Multi-skip feature stacking for action recognition. In *CVPR*, 2015. 8

[31] A. Lerer, S. Gross, and R. Fergus. Learning physical intuition of block towers by example. In *ICML*, 2016. 2

[32] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, pages 740–755, 2014. 1

[33] J. Llargues, J. Peralta, R. Arrabales, M. González, P. Cortez, and A. López. Artificial intelligence approaches for the generation and assessment of believable human-like behaviour in virtual characters. *Expert Systems With Applications*, 41(16):7281–7290, 2014. 2

[34] J. Marín, D. Vázquez, D. Gerónimo, and A. López. Learning appearance in virtual scenarios for pedestrian detection. In *CVPR*, 2010. 1, 2

[35] F. Massa, B. Russell, and M. Aubry. Deep exemplar 2D-3D detection by adapting from real to rendered views. In *CVPR*, 2016. 2

[36] P. Matikainen, R. Sukthankar, and M. Hebert. Feature seeding for action recognition. In *ICCV*, 2011. 2

[37] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convo-

lutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 1, 2

[38] S. Meister and D. Kondermann. Real versus realistically rendered scenes for optical flow evaluation. In *CEMT*, 2011. 2

[39] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing Atari with deep reinforcement learning. In *NIPS, Workshop on Deep Learning*, 2013. 2

[40] N. Onkarappa and A. Sappa. Synthetic sequences and ground-truth flow field generation for algorithm validation. *Multimedia Tools and Applications*, 74(9):3121–3135, 2015. 2

[41] J. Papon and M. Schoeler. Semantic pose using deep networks trained on synthetic RGB-D. In *ICCV*, 2015. 1, 2

[42] X. Peng, B. Sun, K. Ali, and K. Saenko. Learning deep object detectors from 3D models. In *ICCV*, 2015. 1, 2

[43] X. Peng, C. Zou, Y. Qiao, and Q. Peng. Action recognition with stacked fisher vectors. In *ECCV*, 2014. 8

[44] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3D geometry to deformable part models. In *CVPR*, 2012. 2

[45] K. Perlin. Real time responsive animation with personality. *T-VCG*, 1(1):5–15, 1995. 4

[46] K. Perlin and G. Seidman. Autonomous Digital Actors. In *Motion in Games*, pages 246–255. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. 4

[47] L. Pishchulin, A. Jain, C. Wojek, M. Andriluka, T. Thormählen, and B. Schiele. Learning people detection models from few training samples. In *CVPR*, 2011. 2

[48] S. Richter, V. Vineet, S. Roth, and K. Vladlen. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 1, 2, 6

[49] G. Ros, L. Sellart, J. Materzyska, D. Vázquez, and A. López. The SYNTHIA dataset: a large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 1, 2, 3, 6

[50] S. Satkin, J. Lin, and M. Hebert. Data-driven scene understanding from 3D models. In *BMVC*, 2012. 2

[51] A. Shafaei, J. Little, and M. Schmidt. Play and learn: Using video games to train computer vision models. In *BMVC*, 2016. 2

[52] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipmanand, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *CVPR*, 2011. 1, 2

[53] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 1, 6, 8

[54] E. Sizikova1, V. K. Singh, B. Georgescu, M. Halber, K. Ma, and T. Chen. Enhancing place recognition using joint intensity - depth analysis and synthetic data. In *ECCV, VARVAI Workshop*, 2016. 2

[55] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv:1212.0402*, December 2012. 2, 7

[56] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15:1929–1958, 2014. 6

[57] H. Su, C. Qi, Y. Yi, and L. Guibas. Render for CNN: viewpoint estimation in images using CNNs trained with rendered 3D model views. In *ICCV*, 2015. 1, 2

[58] H. Su, F. Wang, Y. Yi, and L. Guibas. 3D-assisted feature synthesis for novel views of an object. In *ICCV*, 2015. 2

[59] B. Sun and K. Saenko. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *BMVC*, 2014. 1

[60] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 6

[61] G. Taylor, A. Chosak, and P. Brewer. OVVV: Using virtual worlds to design and evaluate surveillance systems. In *CVPR*, 2007. 2

[62] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *CVPR*, 2014. 8

[63] H. van Welbergen, B. J. H. Basten, a. Egges, Z. M. Ruttkay, and M. H. Overmars. Real Time Character Animation: A Trade-off Between Naturalness and Control. *Eurographics - State-of-the-Art-Report*, xx:45–72, 2009. 3, 4

[64] D. Vázquez, A. López, D. Ponsa, and J. Marín. Cool world: domain adaptation of virtual and real worlds for human detection using active learning. In *NIPS, Workshop on Domain Adaptation: Theory and Applications*, 2011. 2, 6

[65] D. Vazquez, A. M. López, J. Marín, D. Ponsa, and D. Gerónimo. Virtual and real world adaptation for pedestrian detection. *T-PAMI*, 36(4):797 – 809, 2014. 1, 2

[66] R. Vedantam, X. Lin, T. Batra, C. Zitnick, and D. Parikh. Learning common sense through visual abstraction. In *ICCV*, 2015. 2

[67] V. Veeravasarapu, R. Hota, C. Rothkopf, and R. Visvanathan. Model validation for vision systems via graphics simulation. *CoRR*, arXiv:1512.01401, 2015. 2

[68] V. Veeravasarapu, R. Hota, C. Rothkopf, and R. Visvanathan. Simulations for validation of vision systems. *CoRR*, arXiv:1512.01030, 2015. 2

[69] V. Veeravasarapu, C. Rothkopf, and R. Visvanathan. Model-driven simulations for deep convolutional neural networks. *CoRR*, arXiv:1605.09582, 2016. 2

[70] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *NIPS*, 2016. 2, 3, 8

[71] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *IJCV*, 103:60–79, 2013. 1

[72] H. Wang, D. Oneata, J. Verbeek, and C. Schmid. A robust and efficient video representation for action recognition. *IJCV*, pages 1–20, July 2015. 8

[73] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013. 8

[74] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, 2015. 1, 8

[75] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In *ECCV*, 2016. 2, 3, 6, 7, 8

[76] X. Wang, A. Farhadi, and A. Gupta. Actions $\sim$ Transformations. In *CVPR*, 2015. 8

[77] J. Xu, S. Ramos, D. Vázquez, and A. López. Domain adaptation of deformable part-based models. *T-PAMI*, 36(12):2367–2380, 2014. 2

[78] J. Xu, D. Vázquez, A. López, J. Marín, and D. Ponsa. Learning a part-based pedestrian detector in a virtual world. *T-ITS*, 15(5):2121–2131, 2014. 1, 2

[79] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l1 optical flow. In *DAGM-Symposium*, 2007. 7

[80] Y. Zhu, R. Mottaghi, E. Kolve, J. Lim, and A. Gupta. Target-driven visual navigation in indoor scenes using deep reinforcement learning. *CoRR*, arXiv:1609.05143, 2016. 2

[81] C. Zitnick, R. Vedantam, and D. Parikh. Adopting abstract images for semantic scene understanding. *T-PAMI*, 38(4):627–638, 2016. 2