Computer Vision from Computer Graphics - Large Labeled Image Datasets from 3D models

Rodrigo Frassetto Nogueira

May 19, 2015

Abstract

The recent success of machine learning techniques lies mostly on supervised learning, which requires labeled data to train the models. In this work, we propose an alternative to obtain larger annotated datasets: the use of Computer Graphics 3D models to create synthetic, fully annotated, image and video datasets. We show promising preliminary results on the tasks of object recognition and self-localization using synthetic images and pre-trained convolutional networks.

1 Introduction

Recently, deep learning techniques improved the state-of-the-art in various areas, such as computer vision, speech recognition and natural language processing. However, most of these advances, especially in computer vision tasks, used supervised learning, which requires labeled data to train the models. For instance, results from the latest ILSVCR competitions, which consists more than 1.2M natural images labeled on 1000 different classes shows that large amounts of data enable systems with impressive accuracies. However, large amounts of labeled data can be expensive to be obtained, since they must be manually acquired and annotated. Although the internet contains virtually unlimited data, it can be difficult to obtain samples for some specific applications. For example, if one wishes to construct a robot to recognize paths and whose cameras are mounted close to the ground level, images or videos from that viewpoint are scarce on the web. Or if one wants to automatically recognize defective mechanical parts in an assembly line, it can be difficult to obtain more than hundreds image samples of such items. Besides, some large annotated image datasets, like IMAGENET, are available for non-commercial purposes only. To leverage the problem of obtaining labeled images, we propose the automatic creation of labeled images and videos from 3D graphical models. we want to demonstrate that this approach can be extended to other applications, such as object recognition, self-localization, object tracking and action recognition. In summary, our main goal is train algorithms using large amounts of synthetic images and videos created from simple 3D models and apply them to perform tasks with real images or videos as inputs. However, our goal is not generating pleasantly looking synthetic data, but rather synthetic images that are effective for training purposes.

2 Related Work

Some recent works considered the use of synthetic images to train the machine learning algorithms for various computer vision tasks, specially when acquiring and/or labeling these images is costly. [8] and [6] successfully used synthetic images created from 3D models to train algorithms for pose estimation and object detection (drones and cars), respectively. The authors in [8] generate realistic synthetic depth images of humans of many shapes and sizes in highly varied poses sampled from a large motion capture database. In [6] various techniques are used to make the synthetic image look more real, such as different background images, boundaries blurring (introduction of artifacts due to the discretization of the image sensor, which produces a mixture of the intensities of the background and the target object along its boundaries), motion blurring (mimics the blurring effect that affects on fast moving objects if the shutter time of the camera is too long), random noise (emulates the shot noise) and variations in the diffuse coefficient of the materials. Synthetic images of five common objects (Coffee Pot, Duct tape, Marker, Mug, Synthetic Martini Glass) are used to train a algorithm to predict the best way to grasp objects in [7]. The correct grasping points are determined using a computer graphics ray tracer. In contrast, collecting and manually labeling a comparably sized set of real images would have been extremely time-consuming. Additionally, they randomized different properties of the objects such as color, scale, and text (e.g., on the face of a book).

Images of pedestrians in various poses and environments are used to train a pedestrian detector in [3]. The results are encouraging but the method does not take complex imaging artifacts into account. More recently, an approach to creating more realistic synthetic images by extracting people's silhouettes from real images, and superimposing them over various backgrounds was proposed [4]. However, it is very specific to pedestrian detection and requires a considerable amount of manual annotation.

It was recently shown [5] that it is possible to use a 3D car model to first extract appearance information from real images of cars, and use this information to synthesize novel views. Using these images for training purposes improves performance but this approach does not account for other artifacts such as motion blur and is only applicable to objects with relatively simple geometry.

Optical Character Recognition systems also make used of synthetically generated image patches in [2] and [1]. However, they use only small image patches, which do not resemble large natural images as the ones used this work.

Our work differs from the previous ones that, instead of using complex models and various post-processing rendering operations, we investigate to what extend simple models can be as effective as more complex ones. Although a algorithm trained with complex 3D models is likely to be better than one trained with simpler models, complex models are more expensive to be created, which decreases the appeal of using 3D models instead of real images for training.

3 Creation of Artificial Images from 3D Models

Blender, an open-source 3D computer graphics software (http://www.blender.org/), was used to create images from the 3D models. Python scripts were written to

automatize the process of creating these images, under a variety of illuminations and poses. The background was changed using randomly selected images from a set of 5,000 images. This is necessary to make sure that the classifier will focus only in the object of interest instead of focusing on the background. Noise and lens distortion were added to the image in the post-rendering phase. This is a simple but important step because the rendered images will look more similar to images taken with real cameras.

In order to show the efficacy of the proposed approach, two different set of experiments were conducted:

1. Object Detection: Segways
2. Self-localization in Tennis Courts and Baseball Fields

4 Object Detection: Segways

Our first part of the this work consists in detecting if an specific type of object is present in a image. This is modeled as a simple binary classification problem where images that do not contain the object of interest are negative samples and the ones that contain it are the positive samples. The machine learning algorithms used to this task are the popular Convolutional Neural Networks, which had recent successes in many computer vision benchmarks.

4.1 Why Segways?

Among the many possible objects that we could use to train the machine learning model, we chose Segways because the pre-trained neural network model that we used was never trained with images that contain this type of object. This is important if one wants to evaluate how the model learns new objects. Additionally, real images for comparison could be easily downloaded from the internet and Segways can be easily constructed as a 3D model.

4.2 Experiments

For testing the accuracy of the models, 180 real images of Segways were downloaded from the internet (Figure 1). For the negative samples (that is, images that do not contain segway) we used 180 images like the ones showed in Figure 2.

In order to have a baseline for comparisons purposes, we began testing the performance of the classifier using only real images, like the ones showed in Figure 3. More precisely, we used only 58 real images of a segway for training the classifier. It correctly classifies the presence of a segway in 99% of the testing samples.

For the second experiment, we downloaded a 3D model of a Segway from the internet and generated some images with various poses and illuminations but no background, as showed in Figure 4.

The machine learning model accuracy was 50%, which is random guess. We believe it couldn't learn because the images used for training are too different from testing, mainly due to the absence of background in the training images.

In the next experiment, we added background, represented as four walls and a ground floor whose textures are changed using randomly selected images.

Figure 1: Real Images used for testing.

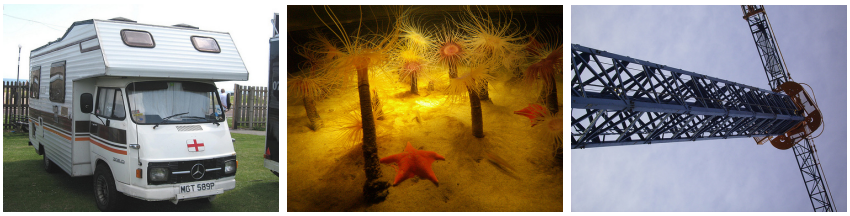

Figure 2: Images used as negative samples in the testing phase.

Figure 3: Real Images used for training our baseline model. Accuracy of **99%**.

Figure 4: Images created from a 3D model of a Segway without background. Accuracy of **50%** (random guess).

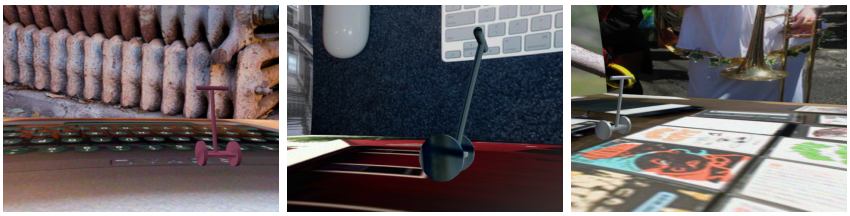

Figure 5: Simple Model with Background using four walls. Accuracy of **57%**.

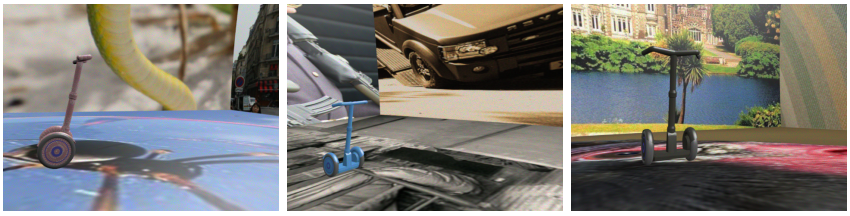

Figure 6: More complex model with Background using four walls. Accuracy of **60%**.

Additionally, a very simple model of a Segway was built using only two cylinders to represent the wheels, one rectangle to represent the supporting platform, one cylinder to represent the handlebar and another to represent supporting rod. The texture of these objects are randomly chosen from the pool of texture images. The images generated from this models are showed in Figure 5. We believe that this is one the simplest models one can create to represent a Segway. We did this on purpose since the goal of these experiments is to verify if it is possible to train machine learning models to detect real objects using only simple 3D models.

The accuracy of the system trained using the simple model (Figure 5) and the complex model (Figure 6) are not encouraging: 57% and 60%, respectively.

One minor addition was the automatic creation of 3D models with different wheel and handlebar sizes. We simple randomly changed the x, y, z scale of the cylinders that represents the wheels and handlebar. The result is a wide range of different 3D models that we expect to help in the generalization of the machine learning model. The images generated from this model are shown in Figure 7. However the accuracy of the system did not improve much: only 62% on the test set.

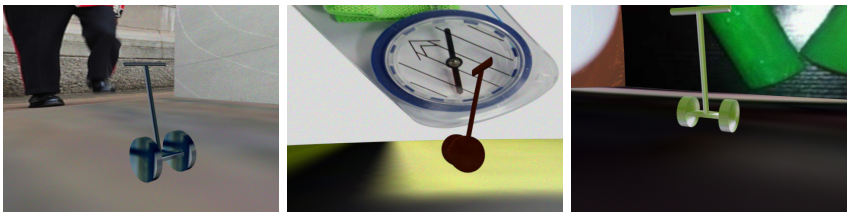

Figure 7: Simple Model with Background. Simple ground textures. Wheels and Handlebar with different sizes. Accuracy of **62%**.

Figure 8: Simple Model with World Background instead of four walls. No Ground. Wheels and Handlebar with different sizes. Accuracy of **80%**.

Figure 9: More Complex Model with World Background instead of four walls. No Ground. Accuracy of **92%**.

One major problem with the previous models is that the corners formed by the ground and the walls are present only in the images that contain Segway. Therefore, we suspected that the classifier learned that positive images (Segways) contains the corners while the negatives (no Segway) does not contain corners. In order to avoid this, the walls and the ground were replaced by a background with a projected image (Figures 9 and 9). With this modification, the accuracy increased to 80% and 92% when using the simple and the complex Segway models, respectively.

4.3 Discussions

The Background influence

The background plays a major role in this kind of approach. We first thought that using more complex backgrounds that tried to simulate indoor and outdoor environments would help the classifier during learning but the experiments proved that we were wrong. The environment was built using four walls and a ground that display random images as texture. Since the corners formed by these walls were the largest structure present in the image, we believe that this is what the classifier learned instead of focusing on the segway.

It turned out that the best way to build the environment was using a simple image as background, that is, no ground, walls or sky. Using this model the accuracy increased dramatically, which strengths the hypothesis that the classifier was learning the corners instead of the 3D segway model in the previous environments.

Model complexity

One of the main questions that we want to answer is how complex the 3D model needs to be in order to train a good detection system. The results show that even using a very simple model of a segway the system achieved an accu-

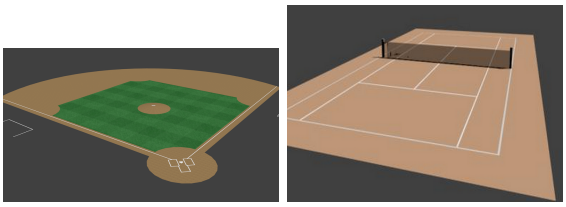

Figure 10: A simple 3D model of a Tennis Court and Baseball Field used to create the synthetic images.

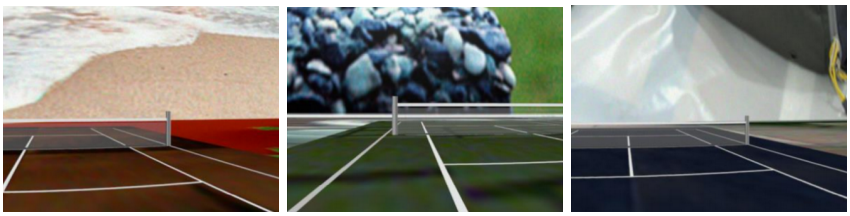

Figure 11: Synthetic images created from the simple 3D model of a Tennis Court with different images as background, camera viewpoints and illuminations.

racy as high as 80%. Using a more complex model the accuracy reached 92%, a significant increase. This indicates that using photo-realistic models would help to increase accuracy even further. However, as this preliminary results showed, very simple/cartoonish models can be useful when complex models are not available or are expensive to be obtained.

5 Self-localization in Tennis Courts and Baseball Fields

Self-localization is the task of identifying an agent's location (the camera) relatively to other objects in the environment. This is a important task in robotics, since the robots must know their position with respect to the environment. In this work, we explore the use of synthetic images to train supervised algorithms to recognize where the camera is positioned related to a tennis court and a baseball field. We used the pre-trained model that took the second place the ILSVRC-2014 object detection competition (VGG 19 layers) to train the network to estimate where a camera is positioned within a tennis court or baseball field using only synthetic images created from a very simple 3D model. Figure 10 shows the 3D models of the tennis court and the baseball-field we used. Some synthetic images created from them are showed in Figures 11 and Figure 12. Random images were added to the background and random texture was added to the ground. Camera position and illuminations were randomly changed. We created 40,000 synthetic images from each model. One of the main advantages of the proposed technique is that the labels for the samples (in this case, the x, y, z coordinates and rotation angles of the camera) are automatically created.

Then, testing was performed on 300 real images for each model. These images have a wide range of illuminations, camera coordinates and quality.

Figure 12: Synthetic images created from the simple 3D model of a Baseball Field with different images as background, camera viewpoints and illuminations.

Figure 13 shows some predictions of the tennis model. In them, the 3D models are projected on the images, based on the predicted coordinates of the camera.

Figure 14 shows the some predictions made by the model on real baseball field images. In the top left map, the yellow square represents the camera position related to the field and the red line represent the vector that the camera is pointing to. We didn't use the projection of the 3D model in the real image (like in the tennis court prediction) due to a problem with prediction of the z (elevation) coordinate. We are working to fix this issue.

5.1 Discussion

The results showed that the proposed approach can be used for robot navigation where the agent (robot) will operate in a delimited/known environment. Examples are robots designed to operate in stores, hospitals, parks or houses. The 3D model of the environment can be easily created using pre-build 3D models of known environments and modern computer graphics softwares.

Using 40,000 synthetic images seems to be enough to achieve a good accuracy as experiments performed with 100,000 images did not improved the performance. Additionally, using more complex 3D models, such as adding ground and walls with different textures, does not affect the accuracy. Therefore, similar to the Segway experiments, using simple models can be as effective as more complex ones.

It was also noticed in the experiments that training one model for each parameter (x, y, z, pitch, yaw and roll) yields better accuracy than training a single model that jointly predicts all parameters. However, we believe that this is mainly because of the size of the model. As the models get larger, the joint model would perform better than a separate model of the same size because it will use an jointly agreement of where is the camera positioned.

The popular technique called dataset augmentation, which artificially multiply the number of samples by cropping and mirroring the original images, was disabled since the labels of the images transformed by these operations would not be valid anymore.

6 Conclusion

This work demonstrated that synthetic images from simple 3D models can be used to train machine learning models in two different tasks: object detection

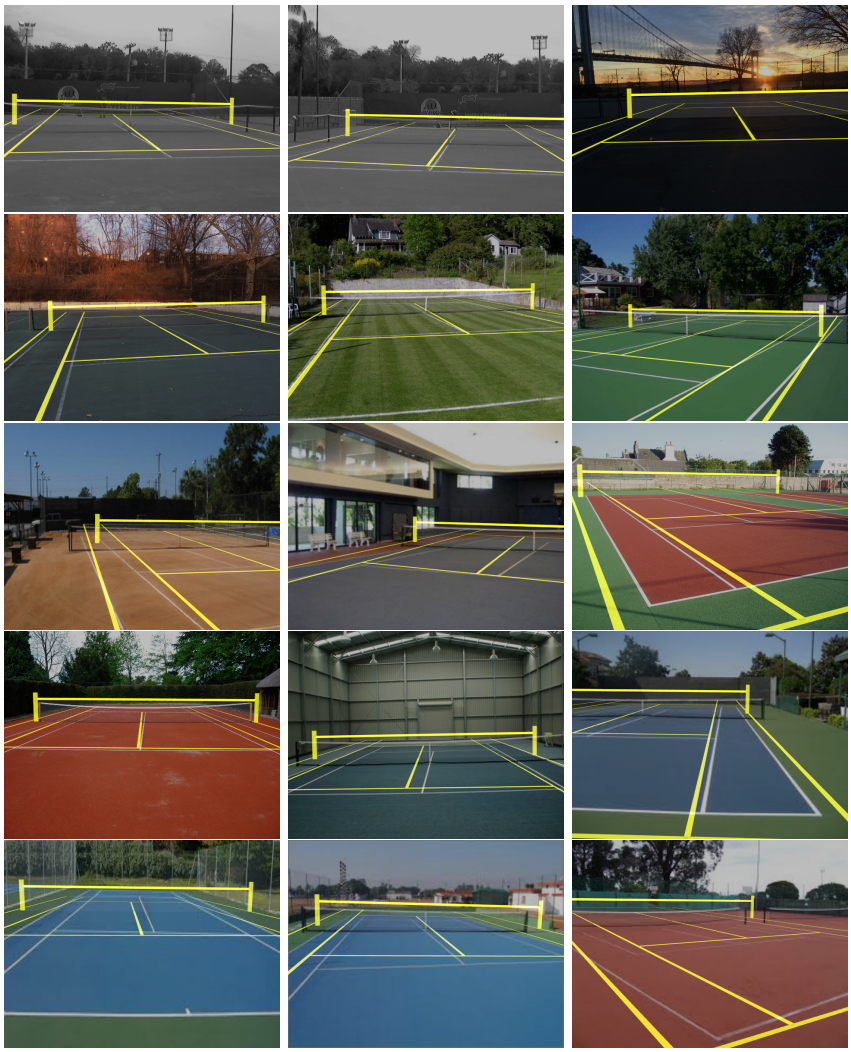

Figure 13: Predictions on real images of tennis courts. The 3D model of the tennis court is projected on the image based on the predicted coordinates of the camera.

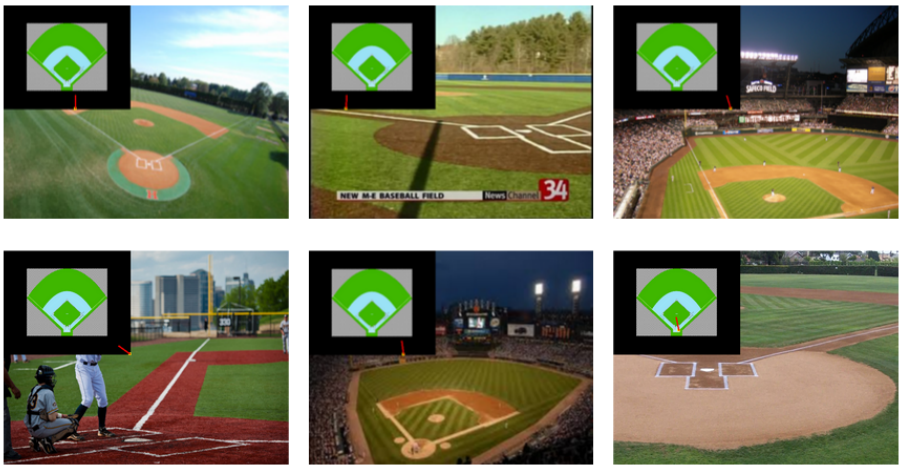

Figure 14: Predictions on real images of baseball fields. In the top left map, the yellow square represents the camera position related to the field and the red line represent the direction that the camera is pointing to.

and self-localization. Contrary to common belief that only training with photo-realistic 3D renderings would yield good accuracies, we demonstrated that simple 3D models can be used to obtain satisfactory results. We also showed that real images are not required for training. For the self-localization task, another advantage of the approach is that it uses monocular images in contrast to some other techniques that use stereo images. However, the use of stereo images could improve the accuracy and they could be easily generated from a 3D model.

7 Future Work

We would like to emphasize that object detection and self-localization are the first steps for automatically full scene understanding. Further work will consist in object tracking using synthetic videos for training. We believe that this approach will make the usage of videos in machine learning more attractive since the labels are automatically created.

More specifically, we will work on baseball and tennis ball tracking using synthetic videos. The simulation of the ball motion can be implemented through build-in functions in softwares like Blender. Figure 15 show some frames of the ball moving trough the baseball court. As it can be seen, the ball is pretty small and hard to see sometimes. Therefore we believe that inputting multiple continuous frames to the model will yield better detectors than using a single a frame.

References

[1] Dan Ciresan, Alessandro Giusti, Luca M Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron mi-

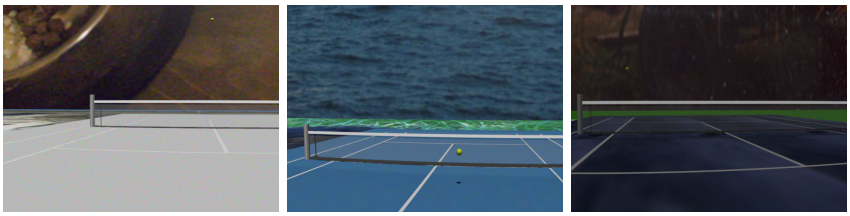

Figure 15: Sample frames extracted from synthetic videos that simulate ball motion. Notice that it is hard to spot the ball in single images due to its small size. Therefore we believe that using videos will make the task of small object tracking easier to the detector.

croscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012.

[2] Vincent Lepetit and Pascal Fua. Keypoint recognition using randomized trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(9):1465–1479, 2006.

[3] Javier Marin, David Vázquez, David Gerónimo, and Antonio M López. Learning appearance in virtual scenarios for pedestrian detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 137–144. IEEE, 2010.

[4] Leonid Pishchulin, Arjun Jain, Mykhaylo Andriluka, T Thormahlen, and Bernt Schiele. Articulated people detection and pose estimation: Reshaping the future. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3178–3185. IEEE, 2012.

[5] Konstantinos Rematas, Tobias Ritschel, Mario Fritz, and Tinne Tuytelaars. Image-based synthesis and re-synthesis of viewpoints guided by 3d models. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3898–3905. IEEE, 2014.

[6] Artem Rozantsev, Vincent Lepetit, and Pascal Fua. On rendering synthetic images for training an object detector. *Computer Vision and Image Understanding*, 2015.

[7] Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. Robotic grasping of novel objects using vision. *The International Journal of Robotics Research*, 27(2):157–173, 2008.

[8] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.