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

Rodrigo Frassetto Nogueira

May 14, 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. Then, we describe our next steps towards the use of synthetic video datasets for real object tracking and action recognition.

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 from objects of 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. Similar work has been done by [1] and [2], in which they successfully used synthetic images created from 3D models to train algorithms for pose estimation and aircraft detection, respectively. In this work, we want to demonstrate that this approach can be extended to various other applications, such as self-localization, object recognition, 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 inputs.

2 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 5000 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 as post-rendering step. This is a simple but important step because the rendered images will look more similar to real 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

3 Object Detection: Segways

First you have to upload the image file from your computer using the upload link the project menu. Then use the includegraphics command to include it in your document. Use the figure environment and the caption command to add a number and a caption to your figure. See the code for Figure 2 in this section for an example.

3.1 Why Segways?

Among the many possible objects that I could use to train the machine learning model, I chose Segways because the pre-trained neural network model that I used was never trained with images that contain Segways. 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 because Segways can be easily constructed in a 3D model.

3.2 Experiments

100 real images of a segway were downloaded from the internet and they were used for testing the accuracy of the model.

In order to have a baseline for comparisons purposes, I began testing the performance of the classifier using only real images, like the ones showed below. More precisely, I used only 58 real images of a segway for training. The classifier could correctly classify the presence of a segway in 99% of the samples.

For the second experiment, I 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 3.

Figure 1: Real Images used for testing.

Figure 2: Real Images used in training for our baseline model. Accuracy of **99%**.

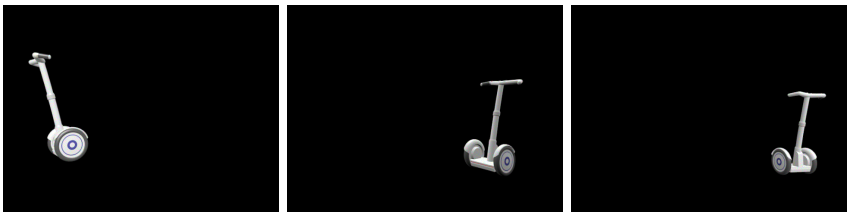

Figure 3: Images created from a 3D model of a Segway without background. Accuracy of **50%** (random guess).

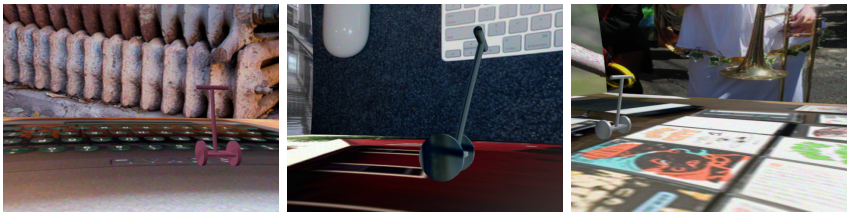

Figure 4: Simple Model with Background using 4 walls and Texture changes in the model. Accuracy of **57%**.

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

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

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 images generated from this models are showed in Figure 4. I believe that this is one the simplest models one can create to represent a segway. I 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. Although complex 3D models are likely to help training better than simpler ones, they are more expensive to be created, which decreases the appeal of using 3D models instead of real images for training.

The accuracy of the system trained using the simple model (Figure 4) and the complex model (Figure 5) are not encouraging: 57% and 60%, respectively.

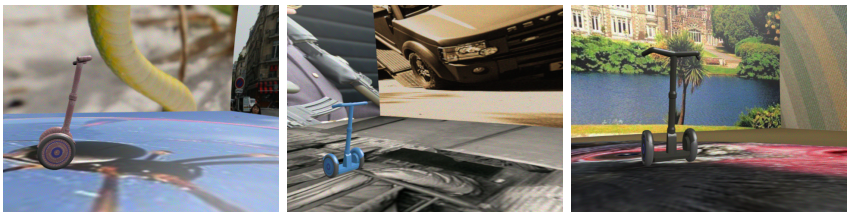

Figure 5: More complex model with Background using 4 walls and Texture changes in the model. Accuracy of **60%**.

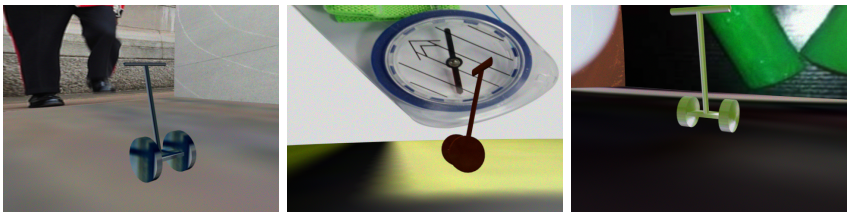

Figure 6: Simple Model with Background and Texture changes. Simple ground textures. Wheels and Handlebar with different sizes. Accuracy of **62%**.

Figure 7: Simple Model with World Background instead of 4 walls. No Ground. Segway texture changes. Wheels and Handlebar with different sizes. Accuracy of **80%**.

One minor addition was the automatic creation of 3D models with different wheel and handlebar sizes. I 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 I expect to help in the generalization of the machine learning model. The images generated from this model are shown in Figure 6. However the accuracy of the system trained with these images did not improve much: only 62% on the test set.

One major problem with the previous models is that the corners formed by the ground and the walls are structures present only in the images that contain segway. Therefore, I 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 projected 2D images (Figures 8 and 8). The result is that the accuracy increase to 80% and 92% when using the simple and the complex segway models, respectively.

Figure 8: More Complex Model with World Background instead of 4 walls. No Ground. Segway texture changes. Accuracy of **92%**.

3.3 Discussions

The Background influence

The background plays a major role in this kind of approach. I 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 I was wrong. I built two complex backgrounds, called Indoor and Outdoor. The Indoor 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, I believe that this is what the classifier learned instead of focusing on the segway.

The Outdoor environment was built using a ground with random images as texture and a random image to simulate the sky/world. Similarly to the Indoor environment, the intersection between the ground and the sky formed a corner that was the largest structure in the image, and therefore, the classifier learned it instead of the segway, I suppose.

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 for the background 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 I 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 accuracy 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 prelimiary results showed, very simple/cartoonish models can be useful when complex models are not available or are expensive to be obtained.

4 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 9 shows the 3D models of the tennis court and the baseball-field we used and some synthetic images created from them are showed in figures 10 and 11. Random images were added to the background, as well as texture to 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 images, which are in this case the x, y, z

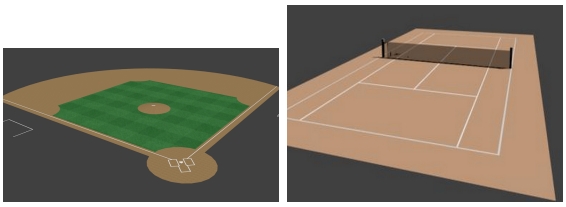

Figure 9: A simple 3D model of a Tennis Court and Baseball Field used to create the synthetic images.

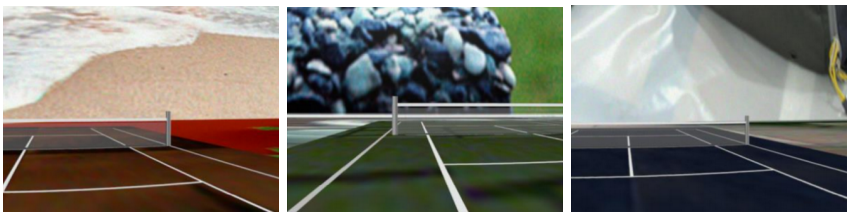

Figure 10: Synthetic images created from the simple 3D model of a Tennis Court with random images as background, camera viewpoints and illuminations.

coordinates and rotation angles of the camera, are automatically created.

Then tests were performed on 300 real images for each model. These images were have a wide range of illuminations, camera coordinates and quality. Figures **??** and 12 show some predictions of the models for baseball and tennis, respectively. In them, the 3D models are projected on the images, based on the predicted coordinates of the camera.

4.1 Discussion

The proposed approach showed that it can be used for robot navigation where the agent (robot) will operate in a delimited environment. Examples are robots designed to operate in stores, hospitals, parks or houses. The 3D model of the environment needs to be recreated, but this can be easily done using pre-build 3D models of known environments.

Using 40K synthetic images seems to be enough to achieve a good accuracy as experiments were performed using 100K images but no significant difference was noticed. Additionally, using more complex model, such as adding ground and

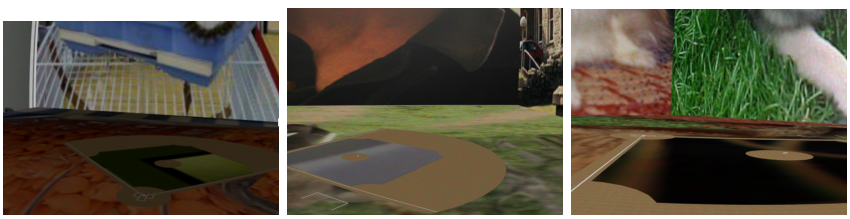

Figure 11: Synthetic images created from the simple 3D model of a Baseball Field with random images as background, camera viewpoints and illuminations.

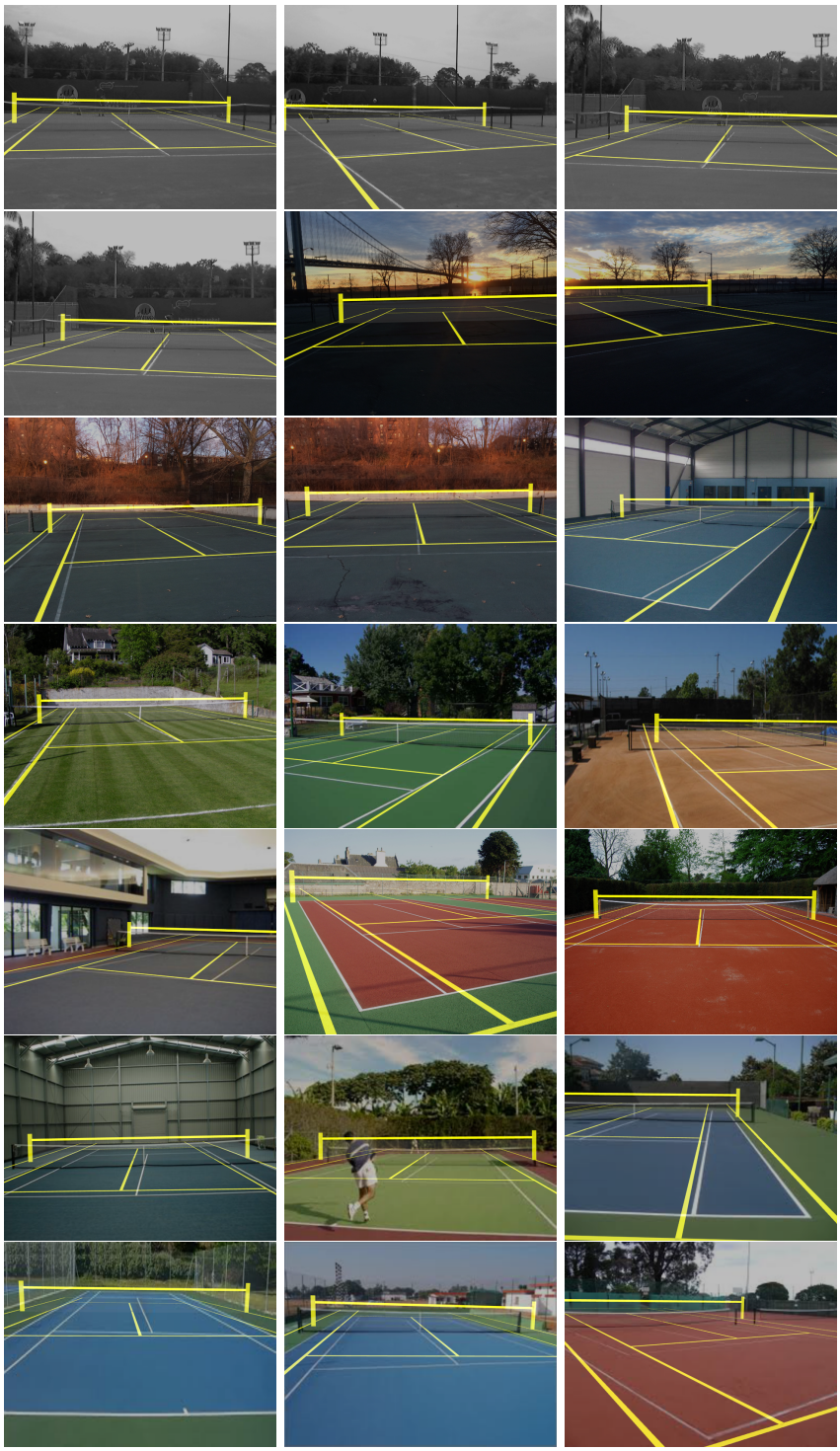

Figure 12: Predictions on the real images of tennis courts. The 3D models of the tennis court is projected on the image based on the predicted coordinates of the camera.

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. 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 operations would not be valid anymore.

5 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 and self-localization. We also showed that real images are not required for training. For the case of self-localization task, another advantage of the approach is that it uses monocular images in contrast to some other approaches that use stereo images. However, the use of stereo images could improve the accuracy and they could be easily generated from a 3D model. Additionally, contrary to common belief that only training photo-realistic 3D renderings would yield good accuracies, we demonstrated that simple 3D models can be used to obtain satisfactory accuracy. We would like to emphasize that object detection and self-localization are the first steps for automatically full scene understanding. Further work will consists in object tracking in videos using synthetic videos for training.