Automatically Dataset Augmentation Using Virtual Human Simulation

Marcelo C. Ghilardi, Leandro Dihl, Estevão Testa, Pedro Braga, João P. Pianta, Isabel H. Manssour, Soraia R. Musse

DaVint - Data Visualization and Interaction Lab,

VHLab - Virtual Human Simulation Lab,

School of Technology, PUCRS-Pontifical Catholic University of Rio Grande do Sul,

Porto Alegre, Brazil

Abstract—**Virtual Human Simulation has been widely used for different purposes, such as comfort or accessibility analysis. In this paper, we investigate the possibility of using this type of technique to extend the training datasets of pedestrians to be used with machine learning techniques. Our main goal is to verify if Computer Graphics (CG) images of virtual humans with a simplistic rendering can be efficient in order to augment datasets used for training machine learning methods. In fact, from a machine learning point of view, there is a need to collect and label large datasets for ground truth, which sometimes demands manual annotation. In addition, find out images and videos with real people and also provide ground truth of people detection and counting is not trivial. If CG images, which can have a ground truth automatically generated, can also be used as training in machine learning techniques for pedestrian detection and counting, it can certainly facilitate and optimize the whole process of event detection. In particular, we propose to parametrize virtual humans using a data-driven approach. Results demonstrated that using the extended datasets with CG images outperforms the results when compared to only real images sequences.**

I. INTRODUCTION

In the last years, there is a growing interest in understanding the behavior of pedestrian and crowds in video sequences. It is important in many applications, but certainly one of the most relevant is the safety of pedestrians in complex buildings or in mass events. Many methodologies to detect groups and crowd events have been proposed in the literature and achieved results showing that groups, social behaviors and navigation aspects can be successfully detected in video sequences. For example, counting people in crowds [1], [2], abnormal behavior detection [3], [4], study of social groups in crowds [5], [6], understanding of group behaviors [7] and characterization of crowd features [8]. Most of these approaches are based on individual pedestrian tracking or optical flow algorithms, and in general consider features like speed, directions, and distance over time.

On the other hand, many of these applications have been also addressed with another perspective, i.e. by using huge datasets for training and testing machine learning techniques, as described in Section II, in order to reach accurate results. One of the main drawbacks of this area is the needed work to build the datasets and respective ground truth, that mainly for pedestrians and crowds is not a trivial task. Sometimes this ground truth is manually prepared, which is very time-consuming. Thus, computer graphics simulations started to be used to generate greater labeled datasets to apply as ground truth. LCrowdV [9] is a recent example of computer graphics technology to generate crowds datasets from a set of provided parameters.

In this paper, we intend to investigate the efficiency of CG images generated with our framework to simulate crowds and render Virtual Humans (VH) in a simplistic way. In particular, we are interested about semi-automatically generating CG images based on a labeled dataset, i.e. using data-driven techniques. The idea behind is to use a parameterized crowd simulator, where information from trajectories labeled in the dataset generates parameters for crowds. We used two crowd simulators to simulate virtual humans and generate automatically the ground truth. In addition, images were rendered using the Unity Engine. We also implemented the Multi-column Convolutional Neural Network (MCNN) proposed in [10] to test the CG dataset and compare the efficiency with known and used dataset UCSD [11]. The main contribution of this paper is the discussion and investigation of VH simulation used to extend crowds and pedestrian datasets in a data-driven way. Results indicate that this idea is indeed promising since it reduces the work of generating ground truth datasets manually labeled.

This paper is organized as follows: Section II describes the literature review on the topic of crowd counting and VH simulation focused on dataset generation. The proposed model is presented in Section III, while experimental results are discussed in Section IV. Finally, Section V draws some conclusions and future work.

II. RELATED WORK

In the last years, several works have been developed for crowd counting [1], [2], [12]–[15] with different purposes, as crowd control, urban planning and video surveillance [16]. This problem consists in the definition of the number of people in a crowd [17], and has been addressed over the years using several approaches [11]–[13], [18], [19], such as Support Vector Machine (SVM) classifier [20] and object detection using a boosted cascade of features [21].

Recently, trying to improve the results accuracy, different methods of Convolutional Neural Networks (CNN) have been widely used [10], [22]–[27]. Sourtzinos et al. [27] presented a method for people counting using CNN, and tested with the available Mall crowd counting dataset [28]. This dataset was annotated manually through the labeling process of head position for each pedestrian in all frames.

Zhang et al. [10], e.g., proposed a Multi-column Convolutional Neural Network (MCNN) that allows input images of arbitrary resolution. However, besides using existing datasets, they also had to collect and annotate a huge dataset to perform the experiments in order to verify the effectiveness of their method. Another large-scale dataset with annotated pedestrians for crowd counting algorithms was provided by Zhao et al. [26]. Gao et al. [24] combined the Adaboost algorithm and the CNN for head detection and used a classroom surveillance dataset also manually annotated to evaluate the proposed method.

Due to the need for this large amount of training data, Boominathan et al. [22] performed an augmentation of their training dataset cropping patches from the multi-scale pyramidal representation of each training image. Cheung et al. [9], on the other hand, claim that the task to manually label the datasets is time-consuming and error-prone, besides needing several human operators. Therefore, they proposed a procedural framework called LCrowdV, to generate labeled crowd videos.

Thus, it is possible to see that although CNN methods present excellent results, there is a need to collect and label large datasets for ground truth, which sometimes demands manual annotation. Because of this, recent research has been addressed to help the problem of generating labeled videos, as the LCrowdV developed by Cheung et al. [9]. Synthetic data has already been used to improve image recognition [29]–[31], however, this approach was not yet explored in crowd/pedestrian counting solutions.

One advantage of crowd simulation applications is the possibility to easily generate a huge dataset together with a ground truth, which fully eliminates the need for an annotated ground truth, and this fact is an important and relevant advantage. This advantage is further enhanced with the possibility to generate automatically labeled crowd videos similar to the real ones, in order to easily extend existent datasets to be used to train machine learning techniques.

III. Proposed Model

Since the focus of this paper is to discuss the automatic process of augmenting labeled datasets, we chose to use one state-of-the-art architecture [10] to conduct our research. We implemented the Multi-column Convolutional Neural Network (MCNN) [10] due to the contribution presented by the authors: their method can manage features at different scales all together in order to accurately estimate crowd counts for different images. However, first of all, we used our simulators in order to simulate virtual humans and generate Virtual

Human datasets. Section III-A describes details about this process.

The overview of our method is presented in Figure 1. It is possible to see the illustration of four used datasets. Firstly, on the top-left appears the UCSD [11] images that were used for training and testing. Such dataset contains low dense crowds and ground truth data. The dataset called "Students" was filmed in our University and presents from 0 to 30 students in a top-view camera, in an environment of 9sqm. The goal is to provide a dataset with a different camera perspective as well as different crowd density if compared to UCSD. This dataset was also used to train and test our method. We tracked the people in Students dataset using a method proposed by Bins et al. [32]. We visually analyzed all tracking data and manually corrected any possible problem, in order to generate a semi-automatic accurate ground truth for Students too. The other two datasets, "CG-CrowdSim" and "CG-BioCrowds", were generated through simulation in order to augment the training data used, as explained below.
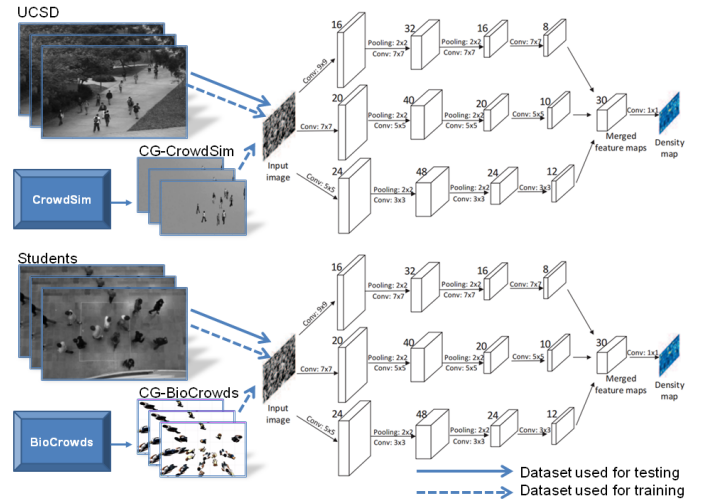

Fig. 1. Outline of the proposed methodology. The UCSD and Students datasets are used to train and test, already the CG dataset was used for only training.

In addition, we used the CG images as a dataset for only the training phase in the machine learning method. Trajectories and behaviors have been generated using two simulators (i.e. CrowdSim and BioCrowds) and rendered at Unity. These two simulators are controlled in a different way. While CrowdSim has a graphical interface that can be used to define the simulation someone is interested in, BioCrowds is a parametrized simulator, that can be data-driven. We firstly used CrowdSim because it is more comparable with LCrowdV method [9], i.e. the crowd designer should manually define initial positions, goals, speeds and etc. So, in CrowdSim case (as developed using LCrowdV) people design a crowd they are interested to simulate. In the case of this work, it is clear that we want to "imitate" the dataset, providing the augmentation. However, this imitation process is user-based since the UCSD data is not used as parameters for CrowdSim. More details about

CrowdSim is presented in Section III-A1. In order to test BioCrowds we read the information stored into the dataset Students and provide an automatic parametrization of the simulator, as discussed in Section III-A2.

In both cases, we used a very simplistic method at Unity (e.g. virtual humans do not generate shadow on the floor). In order to generate the ground truth for machine learning method (agents positions in image coordinates at a function of time), we used the clear advantages that in both simulators all virtual humans positions are known in world coordinates. So, we assumed a classical pinhole camera model $u = Px$, where u is the pixel in the image (homogeneous coordinates), P is the projection matrix 3×4 (known in CG world), e x is the $3D$ position in the world (also inhomogeneous coordinate). In the next sections, we discuss some details about crowd simulators, and then some information about how density maps were generated to be used in the MCNN.

A. Crowd Simulators

This section details the two used crowd simulators.

1) CrowdSim: CrowdSim is a rule-based crowd simulation software developed to simulate coherent motion and behaviors of virtual humans in a geometric environment. In particular, CrowdSim has been used to simulate evacuation scenarios [33]. CrowdSim simulates VH, while keeping behaviors as seek-to-goal and collision avoidance, and also generates outputs to be used in post-processing phases, such as the position of each agent at each time that can be used to visualize the characters in other platforms. In addition, CrowdSim generates statistical data that are used to estimate human comfort and safety in a specific environment, e.g. densities, velocities and etc.

Two key components are considered in CrowdSim, organized in distinct modules: *Configuration* and *Simulation*, which are respectively responsible for configuring the environment/population/routes information and for the simulation and events. For further details, please refer to [33].

Figure 2 illustrates CrowdSim environment. We can see the CrowdSim contexts (walkable regions) and connections among them (white edges) that guide agents to the pre-defined exits. S1, S2, S3, and S4 represent the exits in the simulated night club [33], that was also simulated in real life. The advantages of CrowdSim, if compared to other crowd simulators in literature, is that it has been evaluated and validated according to Galea [34] and also tested in a real scenario. The navigation graph generated by CrowdSim (edges are routes and contexts are nodes), together with the population distribution in the entry contexts (entry rooms) and the expected distributions at the decision points form the definition of our crowd motion plans.

2) BioCrowds: One agent in the environment perceives a set of markers (dots) on the ground (described through a space subdivision method) within its observational radius and move forward to its goals taking into account such markers (unoccupied and closest to this agent than any other one). This is the main aspect of BioCrowds simulator [35] which
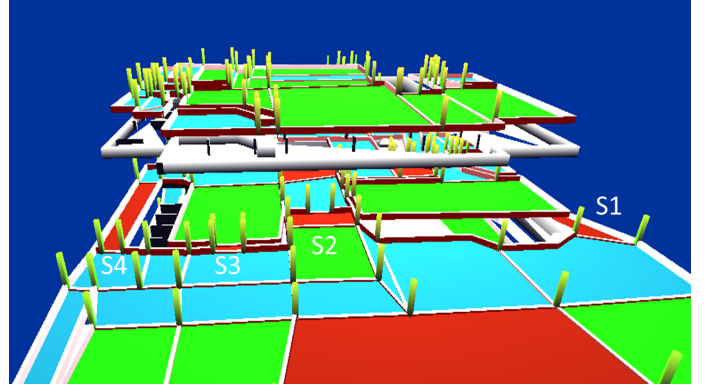

Fig. 2. CrowdSim environment example. S1, S2, S3, and S4 represent the exits in a simulated night club.
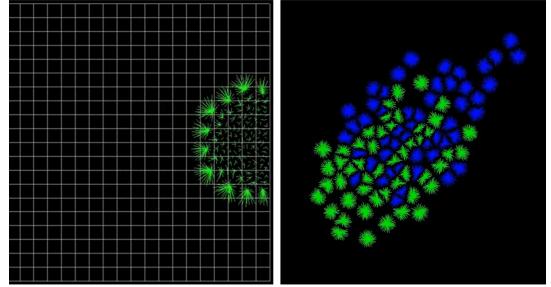

Fig. 3. BioCrowds: We show emergent phenomena being produced by our simulator in a manner similar to other crowd frameworks; on the left: arc formation and on the right: emergent lanes.

supports some of the important emergent behaviors expected in crowd simulation (illustrated in images from Figure 3), as also emergent in other crowd simulators [36], [37]. As a consequence of BioCrowds main functions, obstacles are very easy to represent as zones without any markers in space discretization method.

In order to provide data-driven control, we read information from the dataset, i.e. the number of individuals n and their positions x_i^f in image coordinates, as a function of time f. For the moment we only work with top of view cameras where perspective and coordinates do not need to be transformed. The only performed mapping to generate world coordinates is to find out the correspondent positions in pixels x_i^f to meters X_i^f, in order to compute BioCrowds parameters.

We generate following information for each person i: speed s_i^f (meters/frame) computed based on X_i, initial X_i^{if} and final X_i^{ff} positions for each individual, in frames if and ff, that represents respectively the first and last frame that individual i appeared in the video sequence. Having this data we are able to parametrize BioCrowds as follows:

- $n_B = n$, where n_B is the number of agents in BioCrowds;
- For each agent j in $[0; n_B]$;
- $X_j^{if} = f(x_i^{if})$, where i is the index of individual in a video sequence and j is the index in BioCrowds and $f(w)$

is a function that maps positions from image coordinates to world coordinates;

- Similarly, $X_j^{ff} = f(x_i^{ff})$ and s_j^f are computed.

Then, BioCrowds is able to simulate the n_B agents having their parameters defined w.r.t input dataset. As output, BioCrowds generates the position of each agent at each frame X_j^f. GT and Unity images with virtual humans are the specific output.

It is important to highlight that we chose to simulate people between initial and final frames because we want to be able to simulate the same pattern of crowd existent in the dataset, but allowing to increase or decrease the number of agents, i.e. varying the generated data. For this, we just need to replicate some positions coming from the dataset to serve as input information to agents in BioCrowds. Even if two agents have the same initial and goals positions, they adopt different motion due to the collision avoidance present in BioCrowds method.

B. Generation of Density maps for Computer Graphics Datasets

As mentioned in [10], the estimated crowd density, computed from an input image and used in the training step, is very determinant in the CNN performance. In order to provide CG dataset that can be comparable to the UCSD and [10] results, we used the same method, howeve,r adapted to data obtained from virtual human simulation.

Indeed, for CrowdSim dataset we simulated from 0 to 20 agents and their motion aimed to replicate the environment present at UCSD dataset. For BioCrowds dataset we simulated exactly 22 agents and positions from Students were used to people simulation. For both synthetic datasets, the rendering was processed in real time and 30 images were generated per second. Associated to each image, a file was generated having the position $\vec{X}_i^f$ of each agent i at each frame f in world coordinates. This set of files was used to transform coordinates from world to image, given the camera position used in CG generation, then generating the position of each agent in image coordinate $\vec{u}_i^f$.

In order to generate the density maps for CG datasets, we use distances among agents in the frame. We denote the distances from agent i to its k nearest neighbors (in image coordinates) as $d_i^1, d_i^2, ..., d_i^k$ and the average distance is $\bar{d}_i$. Therefore, to estimate the crowd density around the pixel $\vec{u}_i$, we perform a convolution $\delta(\vec{u} - \vec{u}_i)$ with a Gaussian kernel with variance γ_i proportional to $\bar{d}_i$. For more details please refer to [10]. Figure 4 illustrates images from the four datasets (on the left), the result of density maps generated for GT (middle) and the result of MCNN (right).

IV. EXPERIMENTAL RESULTS

In order to evaluate the performance of CG images in the training phase, we tested two sets of Real Dataset + CG images. Next section (IV-A) aims to discuss the results of UCSD augmented with crowd simulation manually defined.

(a) UCSD image.   (b) UCSD GT.   (c) UCSD output.

(d) CrowdSim image.   (e) CrowdSim GT.   (f) CrowdSim output.

(g) Students image.   (h) Students GT.   (i) Students output.

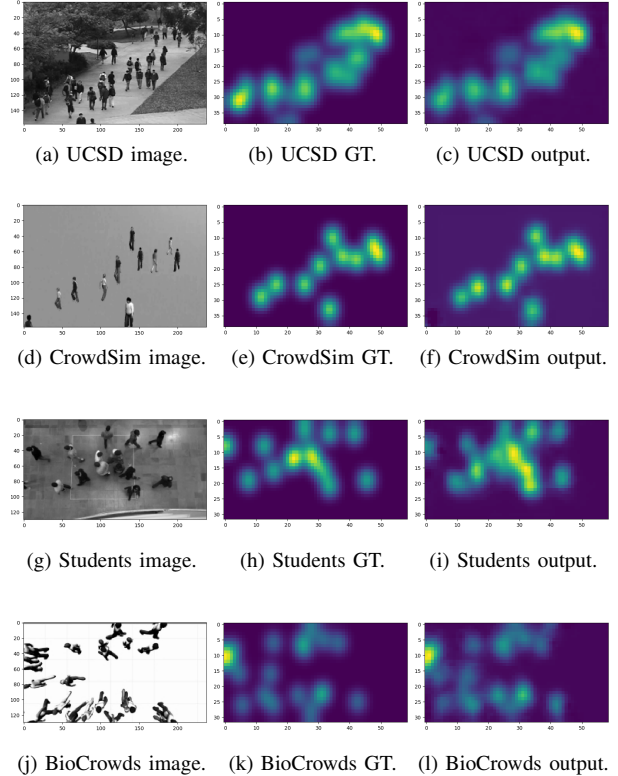(j) BioCrowds image.   (k) BioCrowds GT.   (l) BioCrowds output.

Fig. 4. Some images from the UCSD, CrowdSim, Students and BioCrowds datasets. (a), (b) and (c) illustrate the original image, the processed ground truth, and MCNN output, respectively. (d), (e) and (f) are correspondent images for a dataset using computer graphics (CrowdSim). (g), (h) and (i) are correspondent images for Students dataset. (j), (k) and (l) are correspondent images for a dataset using computer graphics (BioCrowds).

Section IV-B has the goal to show the results when a dataset Students was augmented.

The total number of images are 4630, being 2000 from UCSD, 1545 from CrowdSim (using UCSD as basis), 350 from Students (filmed in our University) and 735 from BioCrowds datasets (using Students as basis). The experimental results aim to show that even these simplistic rendering applied in CrowdSim and BioCrowds datasets can improve the performance of machine learning method. As commonly used, we adopted the MAE (Mean Absolute Error) and MSE (Mean Squared Error) metrics.[1]

First, we evaluated individually the four datasets used in this work for training and testing (see Table I). It is easy to see that the MCNN performance works better in CrowdSim and BioCrowds datasets, when compared to others. One difference among the datasets is that the CG images are more homogeneous, given the synthetic background (see Figure 4 for illustration about the datasets).

Next sections describe the performed analysis in the augmented datasets.

---

[1] Differently from [10], we used MSE and not RMSE.

Training	Testing	MAE	MSE
UCSD(40%)	UCSD(60%)	1.3745	9.4633
CrowdSim(40%)	CrowdSim(60%)	0.7898	3,4966
Students(40%)	Students(60%)	1.1087	5.5069
BioCrowds(40%)	BioCrowds(60%)	1.0981	5.5776

TABLE I
MCNN APPLIED TO THE FOUR ANALYZED DATASETS.

A. UCSD and CrowdSim

We compared the evolution of MCNN performance in two different situations: *i)* when training only with UCSD (40%) and *ii)* extending the training dataset with CG images from CrowdSim (from 0% to 100% of total 1545 images). For UCSD we used the same setting than [10], i.e. we use frames from 601 to 1400 as training data, and the remaining 1200 frames as test data. For CrowdSim dataset, we selected randomly the images until complete the required percentage of images in the dataset (from 0% to 100% of total 1545 images), to be used as training information. The tested images were always the same set of UCSD (60%) for both evaluations. Figure 5 shows the evolution in terms of computed epochs. It is easy to see that the extended dataset using CG images improved the MCNN performance.
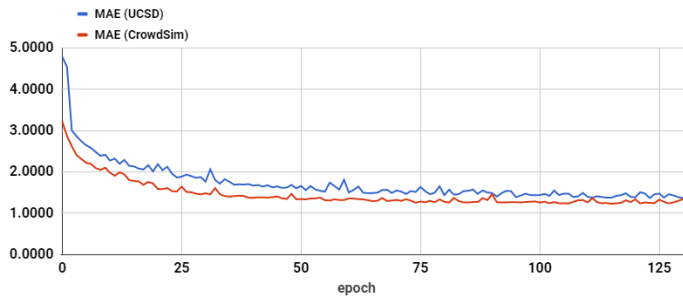
Fig. 5. MCNN performance when training with UCSD and when we extended the dataset with CG images generated using CrowdSim.

Figures 6 and 7 illustrate the results. It is easy to see that the augmentation in the training dataset with CG images provided significantly better performance than without any extension.
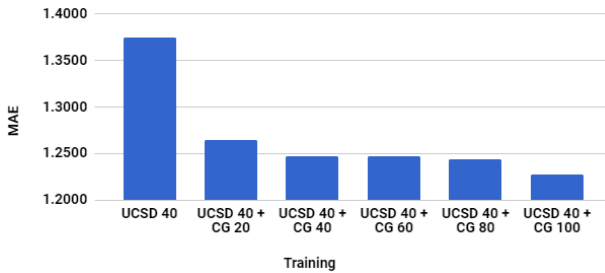
Fig. 6. Comparing the MAE metric when training dataset UCSD was extended with CG images generated using CrowdSim.

In addition, we computed the numerical improvement between the performance with and without augmentation in the UCSD dataset (see Table II). Considering the total images
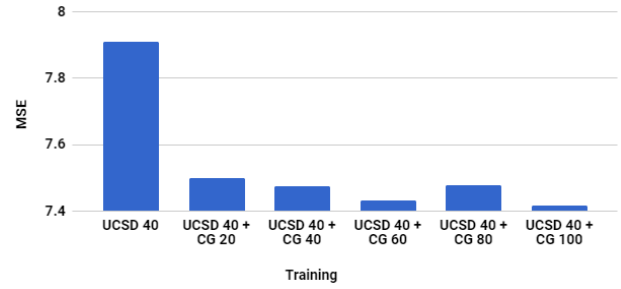
Fig. 7. Comparing the MSE metric when training dataset UCSD was extended with CG images generated using CrowdSim.

of CrowdSim the improvement was approximately 11% in comparison to the original non-augmented dataset UCSD.

Training Dataset	MAE Values	% improvement
UCSD 40%	1.3745	-
UCSD 40% + CG 20%	1.2649	7.9745
UCSD 40% + CG 40%	1.2468	9.2914
UCSD 40% + CG 60%	1.2467	9.2987
UCSD 40% + CG 80%	1.2438	9.5097
UCSD 40% + CG 100%	1.2279	10.6665

TABLE II
COMPARISON OF PERFORMANCE WHEN UCSD WAS EXTENDED WITH CG IMAGES.

As expected, training the MCNN using only CrowdSim dataset was not efficient for testing with any real dataset, indicating that it is necessary to add some real images to the training to obtain better results. The tests used 100% of samples from all datasets (Table III).

Test dataset	MAE	MSE
UCSD	4.3384	43.2086
Students	1.4851	8.3711

TABLE III
TRAINING THE MCNN USING ONLY CROWDSIM DATASET AND TESTING WITH REAL DATASETS.

B. Students and BioCrowds

As in the last section, we compared the evolution of MCNN performance in two different situations: *i)* when training only with Students (40%) and *ii)* extending the training dataset with CG images from BioCrowds (from 0% to 100% of total 735 images). For both cases, we randomly selected the used images. Figure 8 shows the evolution in terms of computed epochs. It is easy to see again that the extended dataset using CG images improved the MCNN performance.

Figures 9 and 10 illustrate the results. It is easy to see that the augmentation in the training dataset with CG images provided significantly better performance than without any extension.

We also computed the numerical improvement between the performance with and without augmentation in the Students dataset (see Table IV).

Next section presents some final considerations about this work.
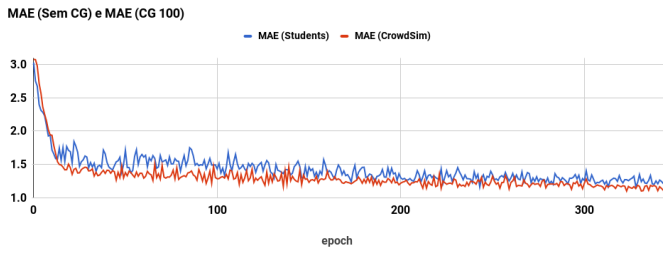
MAE (Sem CG) e MAE (CG 100)

Fig. 8. MCNN performance when training with Students and when we extended the dataset with CG images generated using BioCrowds.
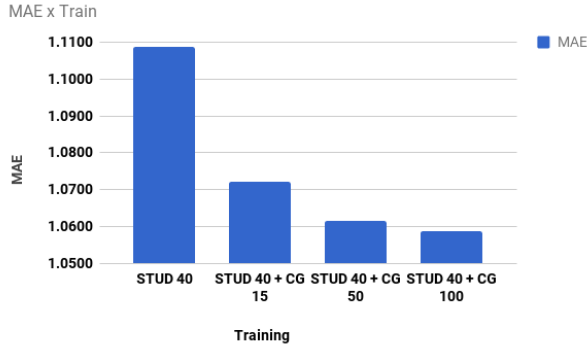

Fig. 9. Comparing the MAE metric when training dataset Students was extended with CG images generated using BioCrowds.

V. FINAL CONSIDERATIONS

In this paper, we described the results of using images of Virtual Human Simulation to extend datasets of pedestrians in order to train machine learning techniques for VH counting and detection. In order to evaluate our proposal, we implemented the Multi-column Convolutional Neural Network (MCNN) presented by Zhang [10]. We used four datasets: UCSD [11], also used in paper [10]; a new one called Student, recorded in our University; a synthetic one created with virtual human simulation similar to UCSD dataset and another synthetic dataset based on Students generated in an automatic
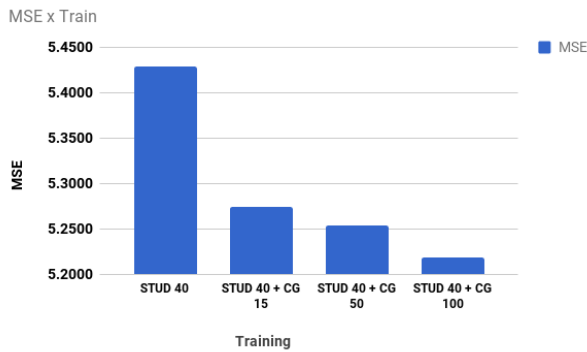

Fig. 10. Comparing the MSE metric when training dataset Students was extended with CG images generated using BioCrowds.

Training Dataset	MAE Values	% improvement
Students 40%	1.1087	-
Students 40% + CG 20%	1.0722	3.2921
Students 40% + CG 40%	1.0615	4.2572
Students 40% + CG 60%	1.0588	4.5007
Students 40% + CG 80%	1.0613	6.3768
Students 40% + CG 100%	1.0342	7.6305

TABLE IV
COMPARISON OF PERFORMANCE WHEN STUDENTS WAS EXTENDED WITH CG IMAGES. THE LAST COLUMN PRESENTS THE IMPROVEMENT IN COMPARISON TO NON-AUGMENTED DATA.

way using BioCrowds.

We trained the MCNN with different samples of the datasets: UCSD only, UCSD+CG, Students only and Students+CG. The results after training with our extended datasets outperform the results of training using just the original dataset, i.e., there was an increase in network performance using the CG extended datasets. In particular, augmenting UCSD with CG images we obtained approximately 10% of improvement in MAE values, while the improvement in Students was approximately 7%. These results were coherent with LCrowdV information when the authors said that their improvement is around 7%. Of course, this number depends on the characteristics of the augmented dataset. The performance improved for both tested datasets demonstrates the good generalization of the proposed investigation. Moreover, the possibility of automatically generating a ground truth for labeling datasets facilitates and optimizes the process of pedestrian detection and counting, decreasing the arduous task of manually labeling the videos.

We also tested two crowd simulators which main difference was the way to control the animations. While in CrowdSim we manually designed experiments imitating UCSD dataset, BioCrowds was automatically parametrized based on datasets. Although tests are necessary, the two simulators do not present differences, in the learning process, since rendering and humans visualization are in the same platform. The only difference is the required work associated with the task to animate crowds, much more easier if using BioCrowds.

For future work, we intend to create, evaluate and let available new datasets of CG images simulating several sizes and densities of crowds. We also want to provide extended datasets to other known datasets, such as the Mall crowd counting dataset [28].

REFERENCES

[1] A. Chan and N. Vasconcelos, "Bayesian poisson regression for crowd counting," in *12th IEEE ICCV*, Sept 2009, pp. 545–551.
[2] Z. Cai, Z. L. Yu, H. Liu, and K. Zhang, "Counting people in crowded scenes by video analyzing," in *9th IEEE ICIEA*, June 2014, pp. 1841–1845.
[3] E. Ermis, V. Saligrama, P. Jodoin, and J. Konrad, "Motion segmentation and abnormal behavior detection via behavior clustering," in *15th IEEE ICIP*, Oct 2008, pp. 769–772.
[4] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly detection in crowded scenes," in *CVPR 2010*, June 2010, pp. 1975–1981.
[5] J. Shao, C. Loy, and X. Wang, "Scene-independent group profiling in crowd," in *IEEE CVPR*, June 2014, pp. 2227–2234.

[6] A. Chandran, L. A. Poh, and P. Vadakkepat, "Identifying social groups in pedestrian crowd videos," in *ICAPR*, Jan 2015, pp. 1–6.

[7] B. Solmaz, B. E. Moore, and M. Shah, "Identifying behaviors in crowd scenes using stability analysis for dynamical systems," *IEEE PAMI*, vol. 34, no. 10, pp. 2064–2070, Oct. 2012.

[8] B. Zhou, X. Tang, H. Zhang, and X. Wang, "Measuring crowd collectiveness," *IEEE PAMI*, vol. 36, no. 8, pp. 1586–1599, Aug 2014.

[9] E. Cheung, T. K. Wong, A. Bera, X. Wang, and D. Manocha, *LCrowdV: Generating Labeled Videos for Simulation-Based Crowd Behavior Learning*. Cham: Springer International Publishing, 2016, pp. 709–727. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-48881-3_50

[10] Y. Zhang, D. Zhou, S. Chen, S. Gao, Y. Ma, undefined, undefined, undefined, and undefined, "Single-image crowd counting via multi-column convolutional neural network," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 00, no. undefined, pp. 589–597, 2016.

[11] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, June 2008, pp. 1–7.

[12] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, Sept 2010.

[13] L. Fiaschi, U. Koethe, R. Nair, and F. A. Hamprecht, "Learning to count with regression forest and structured labels," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, Nov 2012, pp. 2685–2688.

[14] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-scene crowd counting via deep convolutional neural networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 833–841.

[15] Y. Hu, H. Chang, F. Nian, Y. Wang, and T. Li, "Dense crowd counting from still images with convolutional neural networks," *J. Vis. Comun. Image Represent.*, vol. 38, no. C, pp. 530–539, Jul. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.jvcir.2016.03.021

[16] Z. Ma and A. B. Chan, "Counting people crossing a line using integer programming and local features," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 10, pp. 1955–1969, Oct 2016.

[17] B. Sheng, C. Shen, G. Lin, J. Li, W. Yang, and C. Sun, "Crowd counting via weighted vlad on dense attribute feature maps," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, no. 99, pp. 1–1, 2016.

[18] P. Viola, M. J. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *Proceedings Ninth IEEE International Conference on Computer Vision*, Oct 2003, pp. 734–741 vol.2.

[19] V. Rabaud and S. Belongie, "Counting crowded moving objects," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1, June 2006, pp. 705–711.

[20] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995. [Online]. Available: http://dx.doi.org/10.1023/A:1022627411411

[21] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *2001 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2001, pp. I–511–I–518 vol.1.

[22] L. Boominathan, S. S. S. Kruthiventi, and R. V. Babu, "Crowdnet: A deep convolutional network for dense crowd counting," in *Proceedings of the 2016 ACM on Multimedia Conference*, ser. MM '16. New York, NY, USA: ACM, 2016, pp. 640–644. [Online]. Available: http://doi.acm.org/10.1145/2964284.2967300

[23] E. Walach and L. Wolf, *Learning to Count with CNN Boosting*. Cham: Springer International Publishing, 2016, pp. 660–676. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46475-6_41

[24] C. Gao, P. Li, Y. Zhang, J. Liu, and L. Wang, "People counting based on head detection combining adaboost and {CNN} in crowded surveillance environment," *Neurocomputing*, vol. 208, pp. 108 – 116, 2016, sI: BridgingSemantic. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231216304660

[25] D. Oñoro-Rubio and R. J. López-Sastre, *Towards Perspective-Free Object Counting with Deep Learning*. Cham: Springer International Publishing, 2016, pp. 615–629. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46478-7_38

[26] Z. Zhao, H. Li, R. Zhao, and X. Wang, *Crossing-Line Crowd Counting with Two-Phase Deep Neural Networks*. Cham: Springer International Publishing, 2016, pp. 712–726. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46484-8_43

[27] P. Sourtzinos, S. A. Velastin, M. Jara, P. Zegers, and D. Makris, *People Counting in Videos by Fusing Temporal Cues from Spatial Context-Aware Convolutional Neural Networks*. Cham: Springer International Publishing, 2016, pp. 655–667. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-48881-3_46

[28] K. Chen, C. C. Loy, S. Gong, and T. Xiang, "Feature mining for localised crowd counting." in *BMVC*, vol. 1, no. 2, 2012, p. 3.

[29] N. P. H. Thian, S. Marcel, and S. Bengio, "Improving face authentication using virtual samples," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, vol. 3, April 2003, pp. III–233–6 vol.3.

[30] J. Zuo, N. A. Schmid, and X. Chen, "On generation and analysis of synthetic iris images," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 1, pp. 77–90, March 2007.

[31] J. Galbally, R. Plamondon, J. Fierrez, and J. Ortega-Garcia, "Synthetic on-line signature generation. part i: Methodology and algorithms," *Pattern Recogn.*, vol. 45, no. 7, pp. 2610–2621, Jul. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2011.12.011

[32] J. Bins, L. L. Dihl, and C. R. Jung, "Target tracking using multiple patches and weighted vector median filters," *MIV*, vol. 45, no. 3, pp. 293–307, Mar. 2013. [Online]. Available: http://dx.doi.org/10.1007/s10851-012-0354-y

[33] V. Cassol, J. Oliveira, S. R. Musse, and N. Badler, "Analyzing egress accuracy through the study of virtual and real crowds," in *2016 IEEE Virtual Humans and Crowds for Immersive Environments (VHCIE)*, March 2016, pp. 1–6.

[34] E. R. Galea, "A general approach to validating evacuation models with an application to EXODUS," *Journal of Fire Sciences*, vol. 16, no. 6, pp. 414–436, 1998.

[35] A. de Lima Bicho, R. A. Rodrigues, S. R. Musse, C. R. Jung, M. Paravisi, and L. P. Magalhes, "Simulating crowds based on a space colonization algorithm," *Computers & Graphics*, vol. 36, no. 2, pp. 70–79, Apr. 2012.

[36] D. Helbing and A. Johansson, *Pedestrian, crowd and evacuation dynamics*. Springer New York, 2011.

[37] J. van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *IEEE International Conference on Robotics and Automation*, May 2008, pp. 1928–1935.