Virtual Environment Navigation Assisted by Neural Networks

Georgios Kyrlitsias*
University of Cyprus
Department of Computer Sciences

Amyr Borges Fortes Neto†
University of Cyprus
Department of Computer Sciences

Panayiotis Charalambous‡
The Cyprus Insitute, CaSToRC Center

Marios Avraamides§
University of Cyprus,
Department of Psychology
Silversky3D VRT Ltd.

Yiorgos Chrysanthou¶
University of Cyprus,
Department of Computer Sciences
RISE Research Centre

ABSTRACT

Applications using Virtual Environments (VE) are becoming increasingly popular due to greater computational capacity and improvements in graphics processing units and tracking devices. As a result, much research has been carried out on various aspects of VEs, including the input devices that can be used to navigate scenes when physical movement is not permitted. Here, we test whether implementing a neural network to assist users avoid collisions with virtual obstacles, can benefit the navigation experience. Our hypothesis was that users with no gaming experience in particular, would appreciate the assistance of the neural network in navigation. However, our pilot data suggest the exact opposite: participants with video game experience liked the assisted navigation more than participants with no video game experience.

Index Terms: Human-automation navigation—Assisted navigation—Newral networks—Autonomous navigation;

1 INTRODUCTION

Until recently, interactive virtual environments (VE) were popular only in specific domains such as that of Computer Games. However, the improved performance and rendering capabilities of many consumer level devices (e.g., mobile devices), coupled with lower hardware costs, have made VEs popular in applications that go beyond gaming. With the popularization of VEs in devices, an appeal for new applications has emerged. Applications now range from architecture portfolios to applications for clinical interventions and cognitive training. A common theme in many of these VE applications is that the user must navigate the depicted environment.

In this paper, we focus on solutions that aim to help users who are not acquainted with traditional controllers such as joysticks or game pads. Specifically, we test a neural networks (NN)-assisted point and click navigation system that allows users to virtually explore buildings and houses, such as the ones found in architectural portfolios. The NN is responsible to help the user navigate towards the intended goal while at the same time avoiding collisions with obstacles such as furniture, walls and pillars. The idea is that such assisted navigation would improve the user's experience, by removing the need to carry out demanding obstacle avoidance. Although our system was initially designed for autonomous character navigation, in the pilot study we carried out we expected would ease the navigation experience of users initiating the movement themselves. The hypothesis explored was that users with no prior experience in navigating VEs (i.e., non-gamers) would find it easier to navigate with NN assistance than without it.

Although there are other algorithms for local navigation, such as ORCA [16] and RVO [14], we propose the use of NNs in this approach to develop a framework that is able to learn from human expertise. The data collected for the NN training is generated by a human user navigating the virtual environment. With this approach, we are able learn from human expertise, and apply the learned pattern for user assistance. This framework, applied here for virtual navigation, could be further extended for other specific tasks.

In Section 2 we discuss related work in neural network navigation in virtual environments and robots. In Section 3 we describe our method. Section 3.1 presents the NN topology used for this work, Section 3.2 describes the training process and Section 3.3 explains how we generated data for the training of the neural network. Section 4 presents the scenario used for the studies performed. In Section 5 we describe the pilot study we performed to evaluate our work and in Section 6 the results we obtained. Finally, Section 7 discusses possible new directions for the current research.

2 RELATED WORK

Autonomous, collision-free navigation for virtual agents is a problem that has challenged scientists for a long time. Crowd simulation researchers have proposed many approaches, from the seminal work by Reynolds on flocks and herds [13], to Helbing's Social Forces [3], to hierarchical and ruled approaches [2, 9] and geometrical and psychological approaches such as HiDAC [11], ORCA [16] and recently RVO [14]. All of these algorithms rely on geometric and empirical approaches.

Recently, many methods relying on Neural Networks and Deep Learning have been introduced to navigate both virtual agents and robots in the real world. For the virtual agents, some approaches use NNs to autonomously navigate dynamic VE in a collision-free manner [15], [4], [7]. Also, relying on sensor inputs, the navigation of land vehicles such as ALVINN [12] and robots [10], [6], [8], [5] has been described in the literature. Finally and most related to this work, NN have already been used for navigation assistance purposes in a human-automation collaboration framework [1].

Although our model was originally designed for autonomous agent navigation in dynamically changing environments, in the present research we test its suitability for assisting user navigation in VE. Specifically, we test whether it would make VE navigation easier for users who are not experts with navigating VEs using traditional controllers such as joysticks or game pads.

3 METHODOLOGY

This section starts by presenting the methodology we used to generate the data for training the NN. Then we describe the topology of the NN used in our experiment and training algorithms. As already mentioned, although here we use this NN to assist user navigation in

*e-mail: gkyrli01@cs.ucy.ac.cy
†e-mail: aborge01@cs.ucy.ac.cy
‡e-mail: ps.charalambous@cyi.ac.cy
§e-mail: mariosav@ucy.ac.cy
¶e-mail: yiorgos@cs.ucy.ac.cy

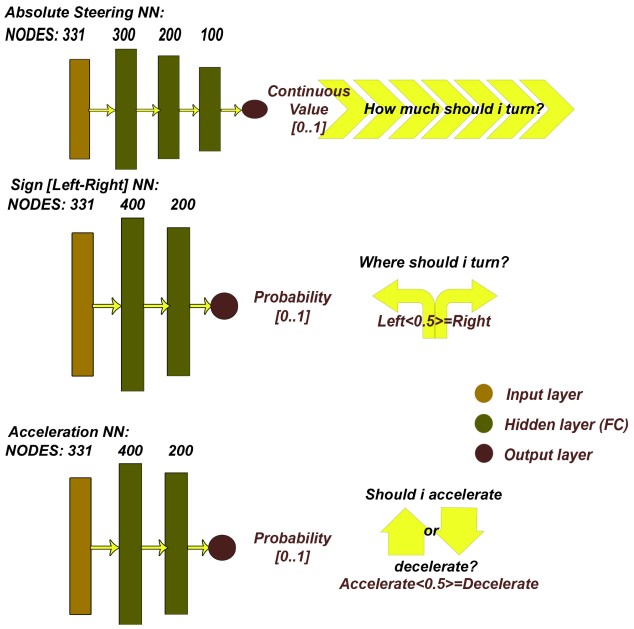

Figure 1: Neural networks and their role in agents behaviour.

a VE, this topology was first conceived for autonomous navigation of virtual agents.

3.1 The Neural Networks Architecture

In our approach we use 3 neural networks to solve the problem of collision avoidance as shown in Figure 1. We use one NN for determining the absolute value of steering angle the agent should use, one NN for the sign of the steering applied (i.e., negative or positive for left or right respectively) and one NN for deciding acceleration (one) or deceleration (zero).

3.1.1 Absolute Steering NN Topology

We used 3 hidden layers that have 300 neurons, 200 neurons and 100 neurons respectively with dropout layers between each of them and one between the last hidden layer and the output layer. The dropout rates are 0.8, 0.6, and 0.5 respectively. The output layer has one node for the steering value. The three hidden layers use the RELU activation function and the output layer uses the sigmoid function. This output is in the $[0,1]$ range; we scale it back and combine it with the sign predicted by the other NN to get the correct rotation.

3.1.2 Acceleration and Sign NNs Topology

Both decisions for acceleration and the sign of the steering value use the same topology. For this aspect of our system we used 2 hidden layers that have 400 and 200 nodes respectively with dropout layers between them with 0.6 and 0.5 dropout rates respectively. The output has one node for the respective prediction. The hidden layers use the RELU activation function and the output uses sigmoid activation function.

3.2 Training Algorithms

For both the absolute steering and the acceleration sign NNs, we decided empirically to use the ADAM optimization algorithm with default parameters provided from Keras library. For our loss functions we used Mean Squared Error for the absolute steering NN and Binary Cross Entropy for the acceleration sign NN.

3.3 Data generation

A crucial aspect of training neural networks is having good data. With that in mind, data was created using human-computer interaction. A dynamic environment was created where the user controls a character. The user aims to reach a goal avoiding obstacles that spawn in front and around him/her while the environment gets more and more difficult over time. The data describing these dynamics are then recorded and used for NN training.

During this process, 331 features were extracted and associated with 2 outputs. The features consists of the combination of 3 distance vectors, each of which measuring the distance from the agent to the closest obstacle in various directions. Every measurement is a straight line in a given angle with maximum distance of 6 meters. There are 36 angular intervals (directions) within a 180^o ahead of the agent as depicted in Figure 2 for each one of the three distance vectors. The first of the three vectors measures angles relative to agent's orientation shown in Figure 2(a). This way it detects the obstacles in the agent's path/moving direction. This results in a vector with 36 floating point values that represents the distance in meters to the closest object in that direction in an [0,6] interval.

The second vector, depicted in Figure 2(b), measures distances in a similar manner; the only difference is that while the first vector looks in front of the agent according to agent's orientation, the second vector looks towards the goal direction. In total, 72 features are measured by these two vectors.

Like the second vector, the third vector measures the same angles, computed relative to the goal direction from the agent's current position analogue to the second vector, as depicted in Figure 2(c). Thus, it detects obstacles that are present between the agent and its desired goal. Notably, this third vector is binary. That is, it obtains the value TRUE, if there is an obstacle in a given direction and FALSE if not. Furthermore, since obstacles that are not straight in the path between the agent and the goal do not impair the agent to reach the goal, this third semicircle (or ellipse) is narrowed according to Equation 1. That is, if there is a hit, the coordinates given by (x,y) must satisfy the conditions in Equation 1 to be a valid hit. The width of the ellipse is given by W and is arbitrarily given a value that is slightly greater than the agent's diameter. Some values were empirically tested for W, without much difference across tests. So, for an agent diameter of 0.123 meters, we choose to apply $W = 0.15m$. The distance D represents the scalar distance from the agent to the goal.

$$f(x,y) = \begin{cases} 1 & \text{if } \frac{x^2}{W^2} + \frac{y^2}{D^2} \leq 1 \\ 0 & \text{otherwise.} \end{cases} . \quad (1)$$

The bit-vector mentioned above helps the NN understand that it can follow its goal even though there are imminent collisions behind the goal or around the goal but not between the agent and the goal, i.e., the path to the goal is clear. Good results can be obtained without the bit-vector as well but more training data are required. With this third vector, we have a total of 108 features. On top of that, we measure the scalar distance from the agent to the goal, considering a maximum distance of 8 meters, beyond which the agent is blind. We include distance to goal in our data because people have different behaviour when they are near or far away from their goal. Besides the distance, the angle between the agent orientation and the goal direction is also measured, helping the NN to orient itself towards the goal, summing up to 110 features. We keep track of all those features for the current frame and the past two frames (first and second derivatives), summing a total of 330 features. Finally, the last feature is TRUE if the goal is directly visible by the agent at any angle, and FALSE if the goal is not visible. This is the last of the 331 features acquired to train the NN.

For our output we decided to take the users module of steering provided by the mouse X position on the screen resulting to values

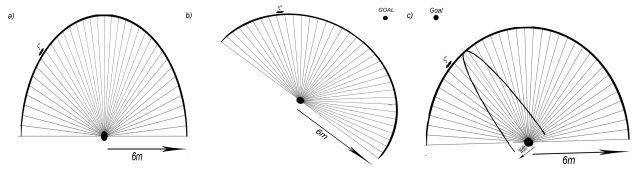

Figure 2: Measurement of obstacles' distances to generate the features.

Figure 3: Screen shot of the experiments used for evaluate the usability of our navigation assistant.

$[-0.53, 0.53]$, -0.53 meaning left, *ZERO* meaning neutral (straight ahead) and 0.53 meaning right. We also have one discrete output indicating whether the user is accelerating (zero) or decelerating (one). All features (inputs and outputs) were collected for 75 thousand frames of user-driven steering, resulting in around 120MB of training data.

3.4 Technical details

It is very important to note some details of the implementation. Our data was generated with constant 50fps. During preprocessing we generated two outputs from the user steering inputs. The first one is the absolute value of steering which was normalized afterwards to $[0, 1]$ and the second one is the sign of the value. These two outputs were used with their respective NN for training. When computing the NN predictions, we scale back the predicted value of absolute steering and combine it with the predicted sign, using a 0.5 threshold. The resulting steering value should be multiplied by a constant value of 300. Moreover, the speed of an agent should be $0 \leq speed \leq 4$. Another important aspect of the data used is that the angle to the goal should be thresholded as $-90 \leq angle \leq 90$ and the resulting angle is taken in consideration to build the goal dependent features.

4 Scenario Construction

To evaluate the usability of our NN in assisting user navigation in a virtual environment, we used a virtual model of the RISE Research Centre headquarters (http://www.rise.org.cy). Figure 3 shows the avatar navigating the virtual environment controlled by the user and assisted by our NN.

The camera rotates along the vertical axis using the mouse X axis (left and right). While the user presses the left mouse button the avatar will moves and stops when the button is released. Notably, it only moves to the direction of the camera, and if an obstacle is in the way, the avatar tries to dodge it according to the NN predictions. This action does not change the camera direction, although it changes the avatar's orientation and trajectory.

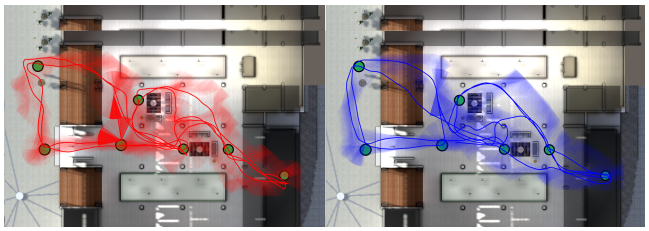

Figure 4: Trajectories of the participants during experiments with (left) and without (right) the neural network assistance.

Table 1: Trajectories distances: average and standard deviation.

	Average	Std. Deviation
With NN	161.4582	23.4751
Without NN	152.3371	15.1463

5 Model Evaluation

To evaluate the usability of our NN as an assistant to navigation, we performed three runs in the described environment for each participant. The first run was without the NN assistance, and was meant as practice for the user, allowing her/him to get used to the avatar movements and the task. This took about one minute and no data were collected.

For the next two runs the participants were asked to navigate in the virtual environment and reach a predefined sequence of goals. The sequence was always the same for all participants. We defined the goals in positions that required the participant to dodge couches, tables, and pillars in the environment. One of those runs was performed with the NN assistance, and the other run was performed without the NN assistance. The order of those runs was counterbalanced across participants. Example of trajectories are shown in Figure 4, where the green dots represents the positions of the predefined goals.

We ran 16 participants (8 females) in this task. After they carried out the two trials, participants were asked to fill out a questionnaire in which they indicated how often they engage in playing video/computer games on a scale from 1 to 5 (1=never, 5=more than 10 hours per week). This allowed us to identify how familiar participants were with controlling virtual characters in a VE. The questionnaire also asked them to indicate how hard they thought the navigation in each trial was on a scale from 1 to 5 (1=very easy, 5=very hard). Finally, we asked the participants to comment generally on the trials they carried out.

6 Results Obtained

We performed a t-test to compare data for the two types of trial (Table 2), although with a small sample and just two trials per participant, we did not anticipate a reliable output. Indeed, although there was no statistical difference in the usability difficulty across the two types of trial, numerically the navigation with NN assistance was rated as slightly more difficult than the navigations without NN assistance. The average distance and standard deviation of participants trajectories are shown in Table 1. It is possible to notice that, in average, the participants travelled longer distances with the NN-assistance than without it.

To take into account the potential mediating effect of familiarity with VE navigation, we examined the difficulty scores separately for participants with and without gaming experience. Descriptive statistics are presented in Table 3. As evident from Table 3 the pattern of results was different for participants with and without gaming experience (although the differences were not statistically reliable in an Analysis of Variance we carried out). While participants with

Table 2: T-Test analysis.

NN	Nr.	Mean	SD	SE
yes	16	2.125	1.025	0.256
no	16	1.938	0.854	0.213

Table 3: ANOVA analysis.

Play	NN	Mean	Std. Error
Never	yes	2.429	0.386
	no	1.714	0.324
Play	yes	1.889	0.340
	no	2.111	0.286

gaming experience found it easier to navigate with the NN than without, the opposite was the case for those with no gaming experience. Although an interesting finding, this contradicts our hypothesis that people not familiar with using joysticks and game pads to move in VEs would find the NN assistance beneficial. One possible explanation for this, supported by comments provided by our participants, is that these participants find the NN assistance as interfering with their control actions. In contrast, those participants who were familiar with video games and virtual environments seemed to grasp more quickly the role of the NN in the avatar movement and appreciate its contribution.

It should be emphasized that the experiment performed here is only a preliminary attempt to test the usability of the NN system. With the input we received from the participants we can implement improvements in the system (e.g., make the goals more distinguishable from the environment, include more obstacles) before we test its usability again with more trials and participants. Perhaps making navigation harder by adding further could make the NN assistance more relevant. Alternatively, it could be that such assistance systems are not suited beyond autonomous settings. Perhaps taking away the sense of control from the users who are supposed to carry out navigation themselves, increase than decreases the difficulty and acceptability of the assistance. This hypothesis will be examined in our future research.

7 Future work

For future work we expect to improve the existing scenario and test it with more participants and trials. We will add more obstacles and friction so the avatar can get stuck on obstacles. We will also examine the influence of participants' sense of control and also test a version in which the model does not have priority over the user control. Finally, we intend to train our system with data acquired from other expert users to examine the effect of different navigation styles on the users' experience.

Acknowledgments

This project is partly funded by the European Unions Horizon 2020 research and innovation programme under grant agreement No. 739578.

This work was carried out during the tenure of an ERCIM Alain Bensoussan Fellowship Programme.

References

[1] T. Adamson, M. Oishi, H.-T. L. Chiang, and L. Tapia. Busy beeway: A game for testing human-automation collaboration for navigation. In *Proceedings of the Tenth International Conference on Motion in Games*, MIG '17, pp. 9:1–9:6. ACM, New York, NY, USA, 2017. doi: 10.1145/3136457.3136471

[2] 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*, 36(2):70 – 79, 2012. Virtual Reality in Brazil 2011. doi: 10.1016/j.cag.2011.12.004

[3] D. Helbing and P. Molnár. Social force model for pedestrian dynamics. *Phys. Rev. E*, 51:4282–4286, May 1995. doi: 10.1103/PhysRevE.51.4282

[4] J. Jaafar, E. McKenzie, and A. Smaill. A fuzzy action selection method for virtual agent navigation in unknown virtual environments. In *2007 IEEE International Fuzzy Systems Conference*, pp. 1–6, July 2007. doi: 10.1109/FUZZY.2007.4295541

[5] B. Ko, H.-J. Choi, C. Hong, J. H. Kim, O. C. Kwon, and C. D. Yoo. Neural network-based autonomous navigation for a homecare mobile robot. In *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 403–406, Feb 2017. doi: 10.1109/BIGCOMP.2017.7881744

[6] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang, and J. Pan. Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning. *CoRR*, abs/1709.10082, 2017.

[7] P. Long, W. Liu, and J. Pan. Deep-learned collision avoidance policy for distributed multi-agent navigation. *CoRR*, abs/1609.06838, 2016.

[8] C. Luo and S. X. Yang. A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments. *IEEE Transactions on Neural Networks*, 19(7):1279–1298, July 2008. doi: 10.1109/TNN.2008.2000394

[9] S. R. Musse and D. Thalmann. Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):152–164, Apr 2001. doi: 10.1109/2945.928167

[10] Y.-K. Na and S.-Y. Oh. Hybrid control for autonomous mobile robot navigation using neural network based behavior modules and environment classification. *Autonomous Robots*, 15(2):193–206, Sep 2003. doi: 10.1023/A:1025597227189

[11] N. Pelechano, J. M. Allbeck, and N. I. Badler. Controlling individual agents in high-density crowd simulation. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '07, pp. 99–108. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007.

[12] D. Pomerleau. Rapidly adapting artificial neural networks for autonomous navigation. In R. P. Lippmann, J. E. Moody, and D. S. Touretzky, eds., *Advances in Neural Information Processing Systems 3*, pp. 429–435. Morgan-Kaufmann, 1991.

[13] C. W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 21(4):25–34, Aug. 1987. doi: 10.1145/37402.37406

[14] M. Rufli, J. Alonso-Mora, and R. Siegwart. Reciprocal collision avoidance with motion continuity constraints. *IEEE Transactions on Robotics*, 29(4):899–912, Aug 2013. doi: 10.1109/TRO.2013.2258733

[15] A.-H. Tan. Falcon: a fusion architecture for learning, cognition, and navigation. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, vol. 4, pp. 3297–3302 vol.4, July 2004. doi: 10.1109/IJCNN.2004.1381208

[16] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha. Reciprocal n-body collision avoidance. In C. Pradalier, R. Siegwart, and G. Hirzinger, eds., *Robotics Research*, pp. 3–19. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.