Moving Virtual Agents Forward in Space and Time

Gabriel F. Silva
Virtual Humans Lab
PUCRS
Porto Alegre, Brazil
gabriel.fonseca94@edu.pucrs.br

Paulo Knob
Virtual Humans Lab
PUCRS
Porto Alegre, Brazil
paulo.knob@edu.pucrs.br

Douglas A. Schlatter
Virtual Humans Lab
PUCRS
Porto Alegre, Brazil
gouglas.schlatter01@edu.pucrs.br

Carlos G. Johansson
Virtual Humans Lab
PUCRS
Porto Alegre, Brazil
carlos@ufcspa.edu.br

Soraia R. Musse
Virtual Humans Lab
PUCRS
Porto Alegre, Brazil
soraia.musse@pucrs.br

***Abstract*—This article proposes an adaptation from the model of Bianco for fast-forwarding agents in crowd simulation, which enables us to accurately fast forward agents in time. Besides being able to jump from one position to another, agents are able to stay inside their track, it means, the new position is calculated taking into account the original global path the agent would follow, if not being fast-forwarded. Obstacles and other agents around are also taken into account when calculating the new position. In addition, we included a personality aspect on agents, which affect their behaviors and, also, be taken into account when jumping to a future time and space. We conducted some experiments to validate our model, which shows that it was able to indeed fast forward agents from a position to another, in a coherent time, sticking to a given global path while avoiding collisions. Finally, we present a use case, showing that our method can fit inside a ”Fog of War” system.**

***Index Terms*—crowd simulation, virtual agents, fast forwarding**

I. Introduction

Since the pioneer work proposed by Thalmann and Musse [1], many other methods were proposed for crowd simulation, each one with a significant contribution. There are methods that deal with crowds from a microscopic point of view [2], [3], as well methods that deal with a macroscopic point of view [4], [5] and, even, methods which combine both microscopic and macroscopic simulation strategies [6]. Others explored high dense crowds [2], [7], heterogeneous behaviors [8], navigation control [9] and personality traits for agents [10], [11].

Despite the great number of methods proposed for the most varied range of subjects concerning crowd simulation, only very few of them tackled the problem of fast-forwarding a simulation or, in other words, instantly jumping agents from a position A in time X to another position B in time Y, while maintaining a coherent path and with a minimum error. Such a feature is especially useful if performance is taken into account. For example, if a simulation is fast-forwarded from frame 200 to frame 300, this interval of 100 frames does not need to be simulated, relieving computational resources. In their work, Bianco et al. [12] proposed a method to estimate the future position of agents in crowd simulations. A Pedestrian Dead Reckoning (PDR) method is used, evaluating the prior positions, velocities, and goals of each agent. The final positions are then estimated based on a global environment complexity factor, that aims to impact the fast-forwarding process, as well as the interaction among agents. Their model was extended by Bianco et al. [13] to allow the inclusion of events (e.g. adding an obstacle) during the jumping period.

In games, movement estimation of agents in virtual environments is a common topic of research, especially regarding computer-controlled units in Real-Time Strategy (RTS) games. Due to ”Fog of War” systems (i.e. vision restricted to only the player units), that estimation must be implemented considering a partially observable environment. Hagelback and Johanso [14] use potential fields to control a bot with imperfect information (i.e. affected by the Fog of War) to explore and navigate the environment. Cho et al. [15] explore the effect of Fog of War system in predictions made by machine learning algorithms. Another common approach includes giving perfect information about the environment to computer-controlled units, allowing the movement estimation to deal with obstacles and terrain deformation hidden by the Fog of War. This approach can be combined with a fast forward method, reducing the computational cost of units navigating through the fog.

Although interesting, the method of Bianco et al. [12] has space to improve. The global environment complexity factor is too general, which means that the more complex the environment, the more chances to have a high error rate and, thus, agents jumping to positions far away from where they should be. Therefore, in this work, we aim to adapt the model of Bianco et al. [12] to make it more accurate. So, given an initial and final time/frame, each agent is able to teleport from its current position A to a future position B, following their own path and time constraints. Thus, we are not only able to teleport agents, but to avoid collision with obstacles and maintain a coherent path based on the agent current position and its goal. Also, our agents can have personality traits, which impact their behaviors and are, also, taken into account in the fast-forwarding method. Finally, as a use case, we integrate our adapted model with a Fog of War system, placing occluded agents in a ”suspended” state until the estimated position is reached or a visible area obstruct its path.

This paper is divided as follows: Section II presents the related work regarding position estimation of agents in crowd simulations and real-world pedestrians, along with methods regarding crowd simulation optimizations during movement prediction. Section III presents the methodology used for integrating a path planning algorithm and personality traits of agents within the fast-forwarding model, as well as how we include personality traits to our agents. Section IV presents the results achieved in the integration process and a use case of our proposed method in a fog of war system. Section V presents the final considerations and future work of our method.

II. Related Work

Due to the increasing demands of simulated environments, such as a higher number of agents and larger scenarios, several methods have been presented to reduce the computational cost of agent movement, collision detection, and collision avoidance on crowd simulations. Pettre et al. [16] presented a method where a level of simulation (LOS), similar to a level of detail (LOD), is given to different sections of the navigation mesh based on the camera viewpoint. Agents closer to the camera are updated with higher frequency. Farther away and occluded agents are given lower priority. Osborne and Dickinson [17] presented a similar method for grouping agents using a hierarchical level of detail (LOD). A group can be defined as a set of agents of the same type, goal, and nearby positions. The LOD of a group is defined based on its distance to the camera. Guy et al. [18] proposed a parallel method for local collision avoidance using the concept of velocity obstacles. Each agent takes into consideration the current velocity of nearby agents to create a set of cones, each containing the possible directions that will cause a collision. After that, the direction of each agent is adjusted seeking a point not included inside the cones, which avoids collision between them.

Different methods and navigation techniques to estimate the position of real-life pedestrians, which were later adapted for virtual agents, are presented in the literature. Beauregard and Haas [19] used a pedestrian dead reckoning (PDR) method combined with acceleration sensors to estimate indoor positions. Taia et al. [20] combines a PDR technique with A* path planning algorithm [21] to provide a more precise approximation of pedestrian routes. Yi et al. [22] proposed a method for estimating the travel time for pedestrians in crowded scenes. Their model defines a group of regions of interest (ROI) for pairings of both sources and destinations, calculating the traffic flow and densities for each region. Environmental elements and stationary persons are taken into account as obstacles. Abnormal behaviors, such as wandering pedestrians, can be identified based on the deviation from their estimated travel time.

Bianco et al. [12] presented a method for estimating the future individual position of virtual agents in crowd simulations. Their work uses a PDR method to define the prior positions of each agent, taking into consideration their goal and speed at the frame where the jump occurs. Future positions are adjusted based on a global environment complexity factor and interaction with other agents. This model was later extended by Bianco et al. [13] to allow the inclusion of events during the time jump period. Events are defined as changes in the environment (i.e. adding, moving of removing obstacles and goals). Also, a metric for comparing crowds was presented, taking into consideration the local densities of uniform regions in the environment to define a global error estimation.

Differing from methods that focus on collision detection optimization or levels of detail, we propose a method that integrates the Pedestrian Dead Reckoning (PDR) position estimation with a global path planning algorithm, allowing us to simulate virtual agents that are aware of obstacles in the environment, including during the fast-forwarding. We also included personality traits in agents that affect their behavior during the simulation, aiming to evaluate their impact on the path planning algorithm and the fast forward process.

III. Proposed Model

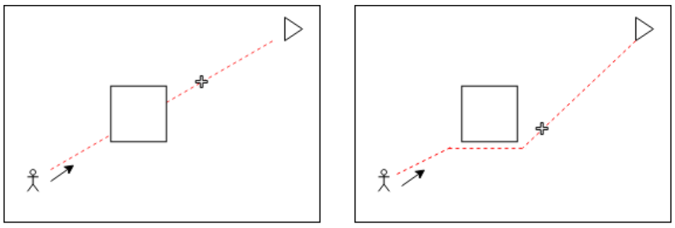


Fig. 1. Comparison between the model presented by Bianco et al [15] (left) and our proposed method (right). An agent is placed on the bottom-left corner of the environment, aiming to reach its goal (triangle). The square represents an obstacle obstructing the agent's path. The position of the agent after the fast forward is represented by a plus sign. Our method introduces a global path planning algorithm, that adjusts the agent's path based on obstructing obstacles, in order to fast forward the agent position in time and space.

The main goal of this work is to adapt the model of Bianco et al. [12] to make it more accurate, regarding the impact of obstacles and motion trajectories. To do so, we propose to remove the global environment complexity factor, that was computed taking into account the global free space to move in the environment. Instead, we control the jumping using a navigation method. In other words, such a navigation method would deliver the global path each agent should follow to reach its goal. When the agent needs to jump in time, its new position is calculated taking such a path into account. Also, we include a personality factor in our agents, which should influence their behavior and be taken into account when fast-forwarding the simulation. Section III-A presents our method to fast forward the simulation, while Section III-B presents our method to include personality on agents and how we deal with it when fast-forwarding a simulation.

A. FF Method Adaptation

As mentioned before, we chose to adapt the model proposed by Bianco et al. [12] to make agents instantly jump from one position to another in time. In short, their method is comprised of four main steps: IP (Interaction with People factor), EC (Environment Complexity factor), PDR (Pedestrian Dead Reckoning), and Repositioning in the space. IP step is responsible for checking how the speeds of each agent should be affected by the presence of nearby agents. EC step is responsible for taking into account the environment complexity (e.g. total area of obstacles) when estimating new positions for the agents. PDR step represents the dead reckoning process on the position estimation, i.e. to where agents go only based on Physics. Finally, the Repositioning step is responsible to reallocate agents in space, following their estimated new positions, but avoiding obstacles and occupied spaces. Their model uses the crowd simulation method known as BioCrowds [23] and, thus, we also do.

We made some adaptations in the method proposed by Bianco et al. [12]. First, their model relies on an Environment Complexity factor (EC) to comply with the free region and the presence of obstacles in the way. However, the area of an obstacle is taken into consideration even when not obstructing the path of an agent. We chose to remove such a factor from the model. Instead, we made two alterations: the inclusion of a navigation method, and the position estimation within the defined path.

Biocrowds [23] is a crowd simulation model that aims to control the local navigation of agents (i.e. define each agent's next position within a pre-defined radius). It can, therefore, have problems to define such next position if an obstacle is in the way. To solve this problem, we chose to use a global navigation method alongside BioCrowds. We implemented the method known as A*, proposed by Hart et al. [21], which is a method to traverse a graph following the shortest path available from an initial to a destiny position. This method also considers obstacles, allowing agents the circumvent them. So, we use A*, to calculate each agent path, and then we compute the total magnitude of the jump in a straight line, as proposed by Bianco et al. [12], to have a value of expected traveled distance based on the speed and position of the agent. Then, we simply project this distance value into the global path of the agent, identifying the point with the same traveled distance. As the A* method takes the obstacle into account, the future position is guaranteed as a point out of any obstacle.

Fig. 1 presents a comparison between the method present by Bianco et al. [12] (left) and ours (right). The previous method establishes a straight line between an agent's position and its goal (triangle). When the fast-forwarding is applied, the agent is repositioned to a target point (plus sign). If this estimated position is inside the area of an obstacle, Bianco et al. [12] identify a nearby available space to place the agent. The introduction of a global path planning algorithm allows the identification of obstacles beforehand and removes the possibility of placing an agent inside an obstacle.

Algorithm 1 presents our Fast Forward Adaptation (FFA). For a given simulation, a stop-frame t and a target frame $t+\Delta t$ are defined, where t represents the frame in which the continuous simulation is interrupted and $t+\Delta t$ represents the frame in which the simulation is resumed after the fast-forwarding. The first step is to estimate the "future" position of an agent $(pos^i_{t+\Delta t})$ using the PDR method, taking into consideration its current position (pos^i_t), movement direction, speed and objective (i.e. final goal). The future position is penalized by an IP (Interaction with People) factor, that considers the presence of other agents. We used the Weibull distribution [12]. The magnitude of vector between $pos^i_{t+\Delta t} - pos^i_t$ is projected in agent i path, identifying the point in space with the same travel distance (following the path). The agent is repositioned at that point and its path planning is updated, removing sub-goals that where "skipped" during the FFA and identifying the next immediate step towards the final goal.

Algorithm 1: Fast Forward Adaptation

```
**Data:** Stop frame ($t$); Target frame ($t + \Delta t$);
Continuous Simulation stops at frame $t$;
**for each** agent $i$ at frame $t$:
    $pos^i_{t+\Delta t}(x^i_{t+\Delta t}, y^i_{t+\Delta t}, z^i_{t+\Delta t}) \leftarrow PDR(x^i_t, y^i_t, z^i_t)$,
    IP($x^i_{t+\Delta t}, y^i_{t+\Delta t}, z^i_{t+\Delta t}$)
    Compute($|(pos^i_{t+\Delta t} - pos^i_t)|$)
    Repositioning($x^i_{t+\Delta t}, y^i_{t+\Delta t}, z^i_{t+\Delta t}$);
    UpdatePathPlanning($i$);
Resume Continuous Simulation from frame $t + \Delta t$;
```

B. Personality Traits

In order to include personality traits to our agents, we chose the OCEAN (Openness, Conscientiousness, Extraversion, Agreeableness, Neuroticism) psychological traits model, proposed by Goldberg [24], once it is the most accepted model to define the personality of a person.

It is important to understand each OCEAN factor individually, so we can define its influence on agents. Therefore, we list the definition of each factor, in short: i) Openness (O) = [0;1]: reflects the degree of curiosity, creativity and a preference for novelty and variety; ii) Conscientiousness (C) = [0;1]: reflects the tendency to be organized and dependable; iii) Extraversion (E) = [0;1]: reflects the sociability and talkativeness; iv)Agreeableness (A) = [0;1]: reflects the tendency to be cooperative and compassionate with others; and v) Neuroticism(N) = [0;1]: reflects the degree of emotional stability.

Following the previous definition, we defined the influence that such traits have on the behavior of our agents, during the FFA method. To do so, we follow the method proposed by Knob et al. [11] to create a relationship between OCEAN traits and behaviors, as following defined:

- Walking Speed [1,2]: defined as a function of Extraversion (E), as follows: $\psi = E + 1$, where ψ is the Walking Speed and E is the Extraversion value;
- Leadership [0,1]: defined as a function of Extraversion (E) and Neuroticism (N), as follows: $\omega = (W * E) + ((1 - W).(1 - N))$, where ω is the Leadership, W is the weight (defined as 0.5), N is the Neuroticism value and E is the Extraversion value;
- Impatience [0,1]: defined as a function of Conscientiousness (C), Agreeableness (A) and Extraversion (E), as follows: $\beta = (W_E * f_E) + (W_{AC} * (1 - A)) + (W_{AC} * (1 - C))$, where β is the Impatience, E is the Extraversion value, C is the Conscientiousness value, A is the Agreeableness value, W_E is a weight (defined as 0.1), W_{AC} is another weight (defined as 0.45) and f_E can assume two different values: if E is higher or equal 0.5, it is calculated as follows: $f_E = (2 * E) - 1$. Otherwise, it assumes the value 0.

These behaviors (i.e. Walking Speed, Leadership, Impatience) are used to define group features among agents. In other words, such groups would present a default behavior based on the features calculated by the OCEAN trait values. Therefore, we define two group features:

- Cohesion (ζ_g) [0,3]: defines how much a group g tends to stay together. The more cohesive the group is, the more agents inside it tend to stay close to each other. At the same time, the less cohesive the group is, the more spread agents of such a group can become. It is calculated in function of the Impatience behavior, as follows: $\zeta_g = (1 - \beta) \times 3$;
- Desired Speed (Ψ_g) [0,1.2]: defines the desired speed for agents inside the group. It is calculated in function of the Walking Speed behavior, as follows: $\Psi_g = 1.2 \times (\psi - 1)$.

Moreover, the Leadership behavior is associated with each agent and defines a given agent that acts as a leader for the other agents of the group. Such a decision is made as follows: the Leadership value of each agent is tested against a threshold (empirically defined as 0.9). If any agent is above such threshold, it is chosen as the leader of the group. If more than one agent surpasses the threshold, one of them is randomly chosen. Finally, if no agent can surpass the threshold, the group has no leader. Agents inside the group follow the behavior of the leader if he is present. Thus, if a strong leader is present, agents inside this group ignore their features and assume the features of the leader. For example, if the leader walks faster, agents of the group will also walk faster.

In short, we link the OCEAN traits to group features which can be easily taken into account for the FFA method. For example, if an agent has a faster pace (defined by Extraversion OCEAN trait), our fast-forwarding method makes this agent "jump" a higher distance, when compared with agents with a lower pace. It happens so because our method predicts the next position of the agent taking into account, besides other factors, the speed of the agent. Therefore, agents with higher speed are going to jump farther than agents with lower speeds.

IV. Results

This section presents the results achieved by our method. For all tests, we performed five simulations of each case and calculated the mean values of them. Then, errors were computed as defined in (1), as follows:

$$\sum_{a=1}^{n} \frac{d(BC^a_{t+\Delta t}, FFA^a_{t+\Delta t})}{n}, \qquad (1)$$

where BC^a and FFA^a represent the positions of agent a, respectively in the simulation, without fast forward method, and in our method, at frame t. d stands for Euclidean distance and n is the total number of agents in the tested case. The error is the average distance that represents the difference between the continuous simulation and our method, in meters.

Section IV-A shows the accuracy of the fast forward method in a obstacles-free environment, while Section IV-B shows how our method behave when obstacles are present in the environment. Section IV-C shows how our the personality traits can affect FFA, and finally, Section IV-E shows a use case for our model.

A. *Our method: Fast Forward Adaptation (FFA)*

We aim to evaluate if the FFA method is working as intended, it means, agents should be able to jump from its positions in time X to a future position in time Y, in the most accurate way possible. To do so, we modeled two different scenarios. First, a 30x30 ($900m^2$) meters scenario with just one goal. Two simulations were run, varying the number of agents (1 agent and 5 agents). Second, a 30x30 meters scenario with two goals. Again, two simulations were run: first, with two groups of 5 agents each; second, with two groups of 10 agents each. Each group wants to reach a different goal (e.g. group 1 wants to reach goal 1 and group 2 wants to reach goal 2). Table I shows all of these simulations. Also, for all simulations, in both scenarios, we ran the same experiment with and without the FFA method. The idea is to check if our model is working as intended (i.e. calculating valid positions for agents in the simulation, where valid positions can be understood as positions nearby the position the agent would be in the simulation without the fast forward method), independently of the number of agents/groups in the simulation and the number of goals present in the environment (the more goals, the more are the chances of the path of the agents to intersect each other). It is worth mentioning that during the execution of our tests, we define that agents should be fast-forwarded from frame 600 to frame 1000. We chose this interval because, in the continuous simulation with two goals, it is the range of time where agents from different groups cross/intersect each other.

Table II presents the results of these simulations. The columns Time, AvgSpeed (average speed), AvgAngVar (average variation of the agent's direction along the path), and AvgDist (average distance between agents) are relative of the Continuous Simulation (i.e. without out FFA method). Time

FFA is the total time that the simulations with our FFA method took, while Error represents the relative error between the positioning of the agents in the Continuous Simulation and the FFA simulations.

TABLE I
SCENARIOS OF THE SIMULATIONS OF THE FFA METHOD. WE VARY THE AMOUNT OF AGENTS, THE AMOUNT OF GROUPS AND THE AMOUNT OF GOALS. TO COMPARE THE RESULTS, WE ALSO RUN THE FOUR SCENARIOS WITHOUT OUR FFA METHOD (CONTINUOUS SIMULATION).

Simulation ID	#Agents	#Groups	#Goals
1	1	1	1
2	5	1	1
3	10	2	2
4	20	2	2

TABLE II
RESULTS OF THE FFA METHOD SCENARIOS. THE COLUMNS TIME, AVGSPEED, AVGANGVAR, AND AVGDIST ARE RELATIVE OF THE CONTINUOUS SIMULATION (I.E. WITHOUT OUT FFA METHOD). TIME FFA IS THE TOTAL TIME THAT THE SIMULATIONS WITH OUR FFA METHOD TOOK, WHILE ERROR REPRESENTS THE RELATIVE ERROR BETWEEN THE POSITIONING OF THE AGENTS IN THE CONTINUOUS SIMULATION AND THE FFA SIMULATIONS.

	Continuous Simulation					
Sim	Time (s)	AvgSpeed (m/s)	AvgAngVar (º)	AvgDist (m)	Time FFA (s)	Avg Error (m)
1	8.26	1.49	3.24	-	6.77	1.85
2	13.73	1.37	19.75	0.41	11.32	1.96
3	15.53	1.13	2.64	0.54	12.03	1.17
4	24.84	1.26	16.98	0.44	19.56	1.20

As presented in Table II, it can be observed a difference between the times of the simulations with and without the FFA method. Continuous simulations presented higher values of Time than simulations with the FFA method, it means, when using the FFA method, simulations were able to finish faster. It leads to the belief that the FFA method can, indeed, help simulations to run faster by avoiding to simulate a defined interval of frames/time. Besides that, the average error between the positioning of agents in the Continuous Simulation and the FFA method is lower than 2 meters for all cases. Such error is calculated as an average displacement in the positioning of all agents present in the simulation, when compared with a continuous simulation. It means that in our $900m^2$ environment, agents presented, when fast forwarded, a displacement in their positioning below 2 meters. This demonstrates that our method is capable to reduce simulation time without needing to compensate with a big loss in precision.

B. Fast Forward with Obstacles

In this section, we aim to evaluate how our fast forward method behaves with the presence of obstacles in the environment. To do so, we modeled the same two scenarios presented in Table I, but with one difference: we added three obstacles in the environment. The environment setup is shown in Fig. 2. We expect that agents are able to jump following their own paths (as defined by the path planning model, explained in Section III-A) and avoiding collision with obstacles in such path, as explained in Section III-A.

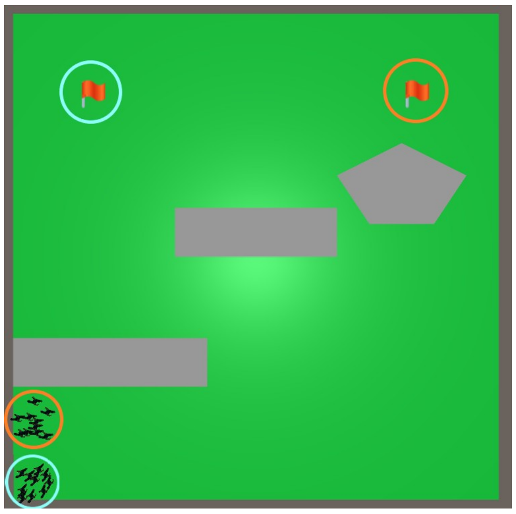


Fig. 2. Scenario used for the simulations presented in Table I. Obstacles are represented by gray polygon. Goals are represented by flags. Agents are initially place at the bottom-left corner. The agents inside the highlighted circles aims for the goals highlighted in the same color. The goal highlighted in orange was used for the simulations containing only one goal.

TABLE III
RESULTS OF SIMULATIONS WITH OBSTACLES. METRICS ARE THE SAME AS PRESENTED IN TABLE II.

	Continuous Simulation					
Sim	Time (s)	Avg Speed (m/s)	Avg AngVar (º)	Avg Dist (m)	Time FFA (s)	Error (m)
1	5.32	1.31	13.81	-	4.12	0.25
2	9.16	1.49	11.29	0.36	9.01	1.18
3	11.89	1.37	7.92	0.45	10.44	1.23
4	27.51	1.36	11.22	1.36	18.92	1.87

Table III shows the results achieved in the simulations with obstacles. As it was already observed in Table II, continuous simulations presented higher values of Time than simulations with the FFA method. It shows that obstacles are having little or no impact on the behavior of the FFA method. Moreover, Error values were kept below 2 meters, as it was already observed in Table II. It is another evidence that the FFA method can deal with obstacles with little or no impact on the path of the agents.

C. Fast Forward with Personality Traits

In this section, we aim to evaluate if the fast forward method is being influenced by the personality of the agents. To do so, we modeled the scenario of Simulation 3 presented in Table I, with the three obstacles used in Section IV-B. We ran four simulations with fast forward to check for personality variations in groups and we add to our tests the difference data (Diff) in relation to the FFA method with obstacles on the same scenario of simulation, so that we can see the differences in the positions of the agents in the simulations and demonstrate the effect of personalities within the proposed model, as presented in Table IV. In simulations 1 and 2, we tested the Leadership behavior of the group, while in simulations 3 and

4 we tested the Impatience behavior. OCEAN traits values were defined following the behavior definitions explained in Section III-B. For simulations 1 and 2, we expect that groups with a strong leader (Simulation 1) are able to reach their goals faster when compared with groups without a strong leader (Simulation 2), because all agents of the group should follow the leader. Also, since Extraversion OCEAN trait is important to define the mean speed of the agents of a group, high values of this trait reflect in higher velocities as seen in III-B. For simulations 3 and 4, we expect that groups with higher levels of Impatience (Simulation 3) present a more disordered behavior when compared with groups with lower levels of Impatience (Simulation 4). Besides that, we expect that groups with high Impatience present slightly higher speeds.

TABLE IV
SCENARIOS FOR THE SIMULATIONS WITH PERSONALITY TRAITS. O, C, E, A AND N REPRESENT THE FIVE TRAITS OF OCEAN.

Simulation	O	C	E	A	N	Test
1	0.5	0.5	0.8	0.5	0.8	Leadership
2	0.5	0.5	0.2	0.5	0.8	W/o Leadership
3	0.5	0.8	0.2	0.8	0.5	Impatient
4	0.5	0.2	0.8	0.2	0.5	Patient

Table V shows the total simulation times for all four simulations. It is possible to notice that the total time of the simulation with a high value of Leadership (Sim 1) is slighter lower than the total time of the simulation with a low value of Leadership (Sim 2). It seems to confirm what we expected: agents inside a group with a strong leader tend to follow its lead and walk at a quicker pace, arriving at their goals faster. Moreover, it can be seen that the total time of the simulation with a high value of Impatient (Sim 3) is greatly lower than the total time of the simulation with a low value of Impatient (Sim 4). Although we did not expect to find so great of a difference, it also seems to confirm what we expected: impatient agents walked at a quicker pace than patient agents, trying to arrive at their respective goals as soon as possible.

TABLE V
RESULTS OF THE SIMULATIONS WITH PERSONALITY TRAITS. METRICS ARE THE SAME AS PRESENTED IN TABLE II, EXCEPT FOR DIFF, WHICH REPRESENTS THE DIFFERENCE IN AGENTS' POSITION BETWEEN THE SIMULATIONS WITH ONLY THE FFA METHOD, AND SIMULATION WITH THE FFA METHOD ALONG WITH PERSONALITY TRAITS.

	Simulation with OCEAN					
Sim	Time (s)	Avg Speed (m/s)	Avg AngVar (°)	Avg Dist (m)	Time FFA (s)	Diff (m)
1	42.56	0.44	10.59	-	4.12	6.00
2	151.49	0.22	24.90	0.82	9.01	10.62
3	21.76	1.03	30.05	1.24	10.44	2.24
4	107.92	0.25	27.58	0.71	18.92	11.07

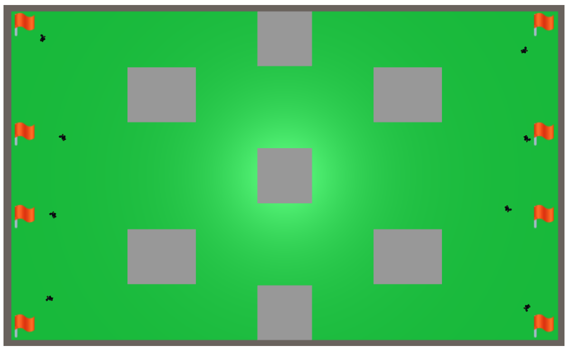
It is important to highlight that simulations 2 and 4 of Table V takes more than double the execution time (i.e. number of frames required for every agent achieve their goal) of simulations 1 and 3 of the same table. In the case of simulation 2, with the aim of the groups to not have a strong leader it is necessary to lower the expressiveness of the OCEAN status, in this case, 0.2 out of 1.0, simultaneously this affects the walking speed characteristic of the agents which influence their mean speed. This can be seen when we compare simulation 1 with simulation 2, in the first agents follow the velocity of it strong leader, this makes the agents at this simulation twice as fast second one, as can been seen in table V, therefore taking more time to reach each group destination goal and increasing simulation time.

In the case of simulation 3 and 4, the degree of impatience of a group determines its cohesion as a group. A high impatience degree culminates in a low cohesion value (i. e a high average distance between agents as seen in simulation 3). The impatience of the agents makes them give more importance for positions near its goal then close from its original group. Furthermore, in order to simulate an impatient behavior, we have to increase the expressiveness of the agents. Mutually, this increases it's walking speed, and hence increasing agents mean velocity. Simulation 4 has a patience behavior, so with more cohesion, which is visible because has the lowest average distance between all tests, and less walking speed, also detectable with a low average speed, this results in a more simulation time for the same reasons as simulation 2. The gaps in time between tests also present in the fast-forwarding method, due to it be coherent with the influences of personality behaviors. An example is that agent 0 in the simulation 1, with a strong leader, in the moment of the fast forward is in the mean position (6.62,4.32), while in simulation 2, without a strong leader, the same agent, in the same frame, does the fast forward and get the position (3.05,3.05). This coherence demonstrates the efficiency of the fast forward method coexists with the effects of personality behavior, through adapting the fast forward for each personality.

D. Comparison with the state-of-the-art

In order to validate the accuracy of our model, we performed a comparison with the model we based our work, which means, the model proposed by Bianco et al. [12]. To do so, we ran the experiments presented in [12] with a similar setup, using a single environment of 40x23 meters (920m^2) containing the following obstacle configurations: no obstacles; 2 obstacles with 109.71m^2 total area; 7 obstacles with 128.68m^2 total area; and 4 obstacles with 582.22m^2 total area. Also, simulations were run with 8, 80, and 160 agents in total.

Fig. 3(a) presents the environment used in the 7-obstacle configuration. Agents are initially placed in the horizontal extremities, needing to cross the environment horizontally to achieve their goals (flags). The same setup for agents and goals were used in the simulations containing 2 and no obstacles. Obstacles were intentionally placed in positions where they would obstruct the direct path (i.e. straight line towards the goal) of agents, converging different groups toward similar areas. This setup creates a higher density of agents, especially during moments where opposing groups intersect each other's paths. Fig. 3(b) presents the environment for the simulation containing 4 obstacles, where the free space is highly reduced. Also, agents are placed in four cardinal extremities of the environment, aiming to reach the opposite extremity.


(a) Scenario containing 7 obstacles.

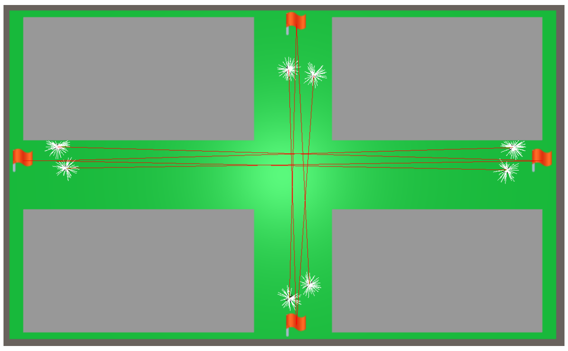

(b) Scenario containing 4 large obstacles.

Fig. 3. Scenarios used for comparison with the method presented by Bianco et al. [12]. Obstacles are represented by gray rectangles. Goals are represented by flags. In (b), the paths defined for each agent in highlighted in red lines. White lines represent a connection between the center of an agent to their auxins (as presented in the BioCrowds [23] model).

Similar to previous simulations, each scenario was executed five times. Then, we calculate the relative error in position (difference) for each agent, in meters, using the same definition as presented by Bianco et al. [12]. The difference formula is defined in (2) as follows:

$$Dif^{a}_{t \to t+\Delta t} = \frac{d(BC^{a}_{t+\Delta t}, FFA^{a}_{t+\Delta t})}{d(BC^{a}_{t}, BC^{a}_{t+\Delta t})}, \qquad (2)$$

where BC^a is the position of agent a at a given frame of the continuous simulation. $FFA^{a}_{t+\Delta t}$ represents the estimated position of agent a at the target frame using our proposed method. d stands for Euclidean distance. The main distinction from (1) is the inclusion of the distance between the positions of an agent a at the stop frame and the target frame of the simulation.

Following the definitions of Bianco et al. [12], the continuous simulation in all scenarios where stopped at frame 200 ($t = 200$) and resumed at frame 400 ($t + \Delta t = 400$). The 4-obstacle setup is an exception due to its goal definition. In this scenario, the values $t = 200$ and $t + \Delta t = 370$ were used. The relative error in position (m) for a single execution is defined by the average Dif of all agents in the simulation. An average between the five executions is presented in Fig. 4 in comparison with the results achieved by Bianco et al. [12].

It can be observed that our proposed FFA method achieved a similar error in most scenarios, indicating that the precision of position estimation is maintained with the integration of a global path planning algorithm. The error increases according to the number of agents in the simulation due to the local movement of an agent being affected by nearby agents, making it harder to correctly predict the future position in scenarios with large intersecting groups. Our method maintained this increasing error behavior in all scenarios, differing from Bianco et al. [12] in the 7-obstacle setup with 8 and 80 agents. We believe that due to the high number of obstacles in this setup, our path planning algorithm was able to better identify immediate obstructions in the agent's routes, allowing a more precise estimation when compared to Bianco et al. [12].

E. *Use Case: Fog of War*

As a use case, we integrated our method with a simple Fog of War system. In this system, a "CPU Player" controls the movement of a set of enemy units (i.e agents) in the environment. The "Human Player" (i.e. user) is then presented with a scenario containing a set of obstacles. Also, the environment is covered in fog, hiding enemy units from the user. The user is also presented with a set of "watch towers" placed in the environment, revealing areas in the fog. We simulated the behavior of "CPU" controlled units using our proposed method. The A* algorithm [21] was used to calculate the path of a unit towards a selected goal, and the FFA model was used to estimate the unit's positions within the path at a set of future frames.

The BioCrowds [23] model uses a grid of cells to represent the space subdivision of the environment. Based on these cells, we create a set of "fog cells" that cover the environment. Each fog cell hides any enemy unit contained inside its area from the user. Also, a subdivision value (s) was used to allow a fog of higher density, therefore, each BioCrowd's cell contained a total of s^2 fog cells. A fog cell contains a state that indicates if it is inside the vision range of a "watch tower" or a dynamic vision area. Watch towers represent stationary elements (e.g. structures and building of the player) that provides vision of a defined area and reveal a set of fog cells. Dynamic vision areas represent any element that provides vision at different moments of the simulation, also being able to move within the environment (e.g player units). CPU units concealed by the fog (i.e. inside a fog cell that is outside vision) are placed in a "suspension" state, where no additional calculations are required until this agent leaves such state.

Each fog cell belonging in the agent's path receives a callback that contains the estimated frame where the unit will enter and leave the cell, along with the respective positions at these frames. A callback is activated when the fog cell is within the vision range and the current simulation frame is within the estimated range. Whenever a callback is activated, the agent leaves the suspension state and is placed back in the environment within the limits of the fog cell. That position is

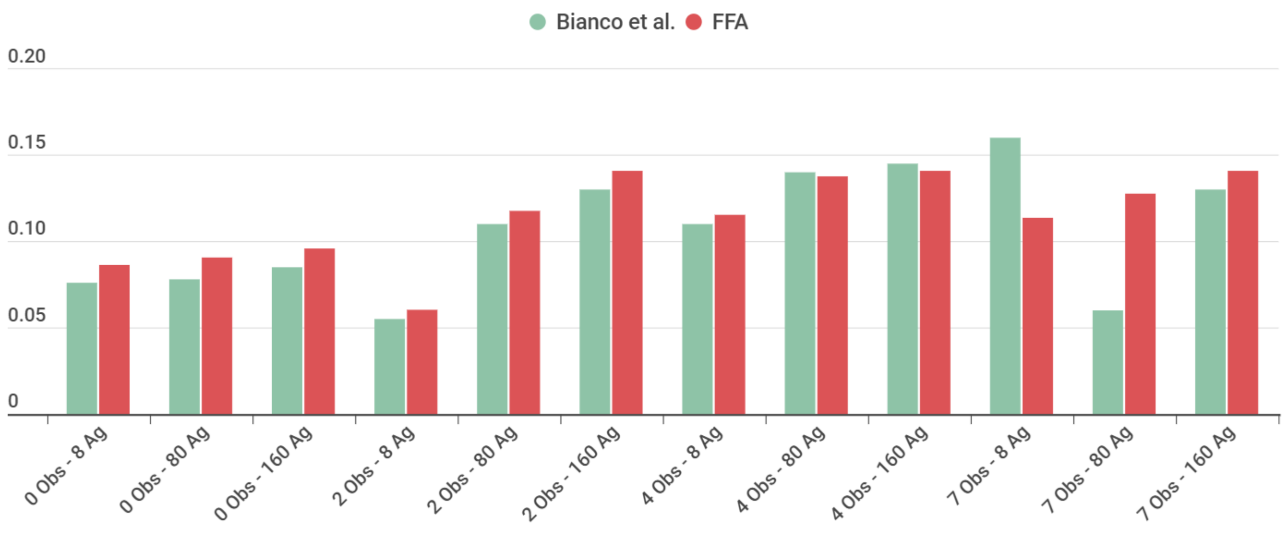


Fig. 4. Comparison of relative error in position (m) between the model presented by Bianco et al. [12] and our proposed FFA method. Values presented are an average between five executions. The value of a single execution is defined by the average Dif of all agents in the simulation, as defined by (2). The number of frames affected by the FFA for each scenario is described in Section IV-D.

defined using a interpolation between the minimum and maximum estimated position for that cell, taking into consideration the frame of the activation. All remaining callbacks for that agent are deactivated at that point. If the agent leaves vision, it is possible to repeat the process, sending it to the suspension state, and reactivating the callbacks. If the target frame (Δt) of the FFA is reached without a callback activation, the agent is placed at the final estimated position.

Fig. 5(a) presents the setup for the environment, containing 30 x 30 meters ($900m^2$). An agent was placed at the bottom of the scene, aiming to reach a goal (flag) at the top. The path defined by the A* algorithm is highlighted in red, connecting a set of cell centers towards the goal. Two “watch towers” were placed near the corners of the obstacle: one revealing a circular area; one revealing a square area. The continuous simulation is interrupted at frame 100 ($t = 100$) and the final position is predicted with target frame 3500 ($t + \Delta t = 3500$). Differing from our standard method, the simulation continues from frame 100 onwards. The agent is placed in a suspended state until the frame 3500 is reached or a callback is activated.

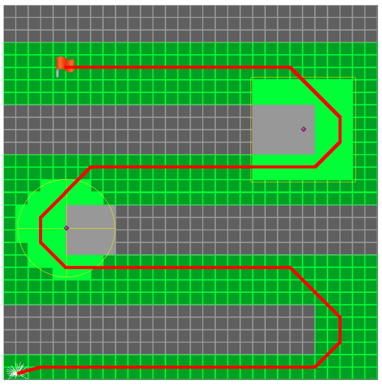
In Fig. 5(b), the agent is removed from the suspension state due to a fog cell, that belongs to the path (i.e. contains a callback), being in view-range. The agent is placed in an estimated position for that cell at the current simulation frame, which is around frame 1650 for that environment. At that point, the agent returns to the continuous simulation until it leaves the revealed area, when it enters into suspension state again. In Fig. 5(c), the “watch towers” are removed and no dynamic vision areas are used. Therefore, all the environment is concealed by the fog of war, and the agent suspension state is not interrupted. In that scenario, the agent enters the suspension state at frame 100 and leaves at frame 3500, close to its goal.

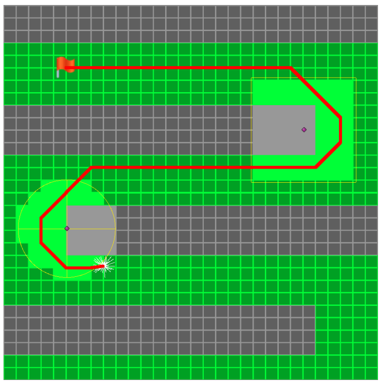
V. Final Considerations

This work proposed an adaptation from the model of Bianco for fast-forwarding agents in crowd simulation, which enables us to accurately fast forward agents in time. The future positions of the agents are calculated taking into account the original global path the agent would follow, if not being fast-forwarded. This way, obstacles and other agents around are also taken into account when calculating the new position. Besides that, we included a personality aspect on agents which is taken into account when jumping to a future position.

We ran several tests with our model. The results achieved show that our fast-forwarding method can be a valuable asset for crowd simulation, especially in some given domains. Most notable is the gain in time and resources: since the interval of frames/time where agents are being fast-forwarded does not need to be simulated, the simulation is faster and computational resources can be relieved or spent in other tasks. Also, we integrated our model with a Fog of War system, showing an example of how our model could be allocated in a game.

As for future work, there are several avenues to be followed. We can modify our path planning method (i.e. A*) to consider the number of agents in each cell present in the path of a given agent. It could be used to avoid jams, where a great number of agents become locked by each other. In this case, lanes could be formed by the paths to avoid such jams. Also, in the same way Bianco uses a complexity factor to ”punish” the jumping in their method, we could use the complexity of the path to ”punish” the jumping in our method. It would help agents to avoid eventual deviation from their original routes (for example, when recalculating their path after the jump). To do both of the cited future works, D* algorithm could be used instead of A*, since it allows dynamic weights for


(a) Defined path before suspension.


(b) Suspension interrupted by a revealed area.

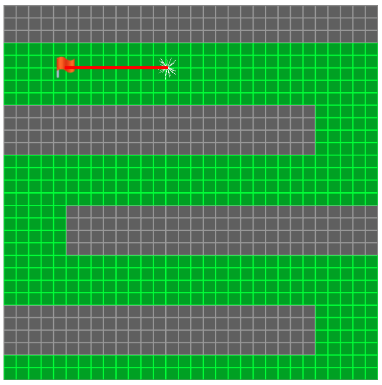

(c) Suspension not interrupted.

Fig. 5. FFA integration with a Fog of War system. In (a), the defined path of an agent is highlight int red. In (b), the suspension is interrupted due to a "watch tower" revealing a segment of the path. The agent continues within the path until he leaves vision, entering in another state of suspension. In (c), the set of "watch towers" is removed, allowing the entire section to be skipped. The agent is then positioned at the predicted point when the target frame is reached.

each node of the path. Finally, in our model, we adopted the group behavior from the work of Knob. Although, we did not simulate the fast forward method for groups of agents. Such a feature would allow us to fast forward entire groups of agents (e.g. armies in a game), which would cost less computational resources than to fast forward each agent separately.

References

[1] S. R. Musse and D. Thalmann, "A model of human crowd behavior : Group inter-relationship and collision detection analysis," in *Computer Animation and Simulation '97*, D. Thalmann and M. van de Panne, Eds. Vienna: Springer Vienna, 1997, pp. 39–51.

[2] 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*, ser. SCA '07. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 99–108. [Online]. Available: http://dl.acm.org/citation.cfm?id=1272690.1272705

[3] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 25–34, Aug. 1987. [Online]. Available: http://doi.acm.org/10.1145/37402.37406

[4] R. L. Hughes, "A continuum theory for the flow of pedestrians," *Transportation Research Part B: Methodological*, vol. 36, no. 6, pp. 507–535, 2002.

[5] A. Treuille, S. Cooper, and Z. Popović, "Continuum crowds," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1160–1168, Jul. 2006. [Online]. Available: http://doi.acm.org/10.1145/1141911.1142008

[6] A. D. S. Antonitsch, D. H. M. Schaffer, G. W. Rockenbach, P. Knob, and S. R. Musse, "Bioclouds: A multi-level model to simulate and visualize large crowds," in *Computer Graphics International Conference*. Springer, 2019, pp. 15–27.

[7] R. Narain, A. Golas, S. Curtis, and M. C. Lin, "Aggregate dynamics for dense crowd simulation," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 122:1–122:8, Dec. 2009. [Online]. Available: http://doi.acm.org/10.1145/1618452.1618468

[8] L. Zheng, D. Qin, Y. Cheng, L. Wang, and L. Li, "Simulating heterogeneous crowds from a physiological perspective," *Neurocomput.*, vol. 172, no. C, pp. 180–188, Jan. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2014.12.103

[9] S. Paris, J. Pettré, and S. Donikian, "Pedestrian reactive navigation for crowd simulation: a predictive approach," *Comput. Graph. Forum*, vol. 26, pp. 665–674, 2007.

[10] F. Durupinar, N. Pelechano, J. Allbeck, U. Güdükbay, and N. I. Badler, "How the ocean personality model affects the perception of crowds," *IEEE Computer Graphics and Applications*, vol. 31, no. 3, pp. 22–31, 2009.

[11] P. Knob, M. Balotin, and S. R. Musse, "Simulating crowds with ocean personality traits," in *Proceedings of the 18th international conference on intelligent virtual agents*, 2018, pp. 233–238.

[12] C. M. D. Bianco, A. Braun, S. R. Musse, C. R. Jung, and N. I. Badler, "Fast-forwarding crowd simulations," in *Intelligent Virtual Agents - 16th International Conference, IVA 2016, Los Angeles, CA, USA, September 20-23, 2016, Proceedings*, ser. Lecture Notes in Computer Science, D. R. Traum, W. R. Swartout, P. Khooshabeh, S. Kopp, S. Scherer, and A. Leuski, Eds., vol. 10011, 2016, pp. 208–217. [Online]. Available: https://doi.org/10.1007/978-3-319-47665-0_19

[13] C. M. D. Bianco, S. R. Musse, A. Braun, R. P. Caetani, C. R. Jung, and N. I. Badler, "Predicting future crowd motion including event treatment," in *Intelligent Virtual Agents - 17th International Conference, IVA 2017, Stockholm, Sweden, August 27-30, 2017, Proceedings*, ser. Lecture Notes in Computer Science, J. Beskow, C. E. Peters, G. Castellano, C. O'Sullivan, I. Leite, and S. Kopp, Eds., vol. 10498. Springer, 2017, pp. 101–104. [Online]. Available: https://doi.org/10.1007/978-3-319-67401-8_11

[14] J. Hagelback and S. J. Johansson, "Dealing with fog of war in a real time strategy game environment," in *2008 IEEE Symposium On Computational Intelligence and Games*, 2008, pp. 55–62.

[15] H. Cho, H. Park, C.-Y. Kim, and K.-J. Kim, "Investigation of the effect of "fog of war" in the prediction of starcraft strategy using machine learning," *Comput. Entertain.*, vol. 14, no. 1, Dec. 2016. [Online]. Available: https://doi.org/10.1145/2735384

[16] J. Pettre, P. de Heras Ciechomski, J. Maïm, B. Yersin, J.-P. Laumond, and D. Thalmann, "Real-time navigating crowds: Scalable simulation and rendering," *Computer Animation and Virtual Worlds*, vol. 17, pp. 445–455, 07 2006.

[17] D. Osborne and P. Dickinson, "Improving games ai performance using grouped hierarchical level of detail," 01 2010.

[18] S. J. Guy, J. Chhugani, C. Kim, N. Satish, M. Lin, D. Manocha, and P. Dubey, "Clearpath: Highly parallel collision avoidance for multi-agent simulation," in *Proceedings of the 2009 ACM SIGGRAPH/Eurographics*

Symposium on Computer Animation, ser. SCA '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 177–187. [Online]. Available: https://doi.org/10.1145/1599470.1599494

[19] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication (WPNC'06)*, 01 2006.

[20] F. Taia Alaoui, D. Bétaille, and V. Renaudin, "Pedestrian dead reckoning navigation with the help of a $*$ -based routing graphs in large unconstrained spaces," *Wireless Communications and Mobile Computing*, vol. 2017, pp. 1–10, 07 2017.

[21] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[22] S. Yi, H. Li, and X. Wang, "Pedestrian travel time estimation in crowded scenes," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 3137–3145.

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

[24] L. R. Goldberg, "An alternative" description of personality": the big-five factor structure." *Journal of personality and social psychology*, vol. 59, no. 6, p. 1216, 1990.