Miguel, Amblard, Barceló & Madella (eds.) Advances in Computational Social Science and Social Simulation
Barcelona: Autònoma University of Barcelona, 2014, DDD repository <http://ddd.uab.cat/record/125597>

Populating Virtual Cities with Diverse Physiology Driven Crowds of Intelligent Agents

Tomas Trescak
School of Computing,
Engineering and Mathematics
University of Western Sydney,
Sydney, Australia,
t.trescak@uws.edu.au

Anton Bogdanovych
School of Computing,
Engineering and Mathematics
University of Western Sydney,
Sydney, Australia,
a.bogdanovych@uws.edu.au

Simeon Simoff
School of Computing,
Engineering and Mathematics
University of Western Sydney,
Sydney, Australia,
s.simoff@uws.edu.au

***Abstract*—When conducting archaeological excavations of ancient cities, 3D reconstruction has become an important mechanism of documenting the findings and showing the results to general public in an accessible way. Most such reconstructions, however, mainly focus on visualising buildings and artefacts, while rarely simulating the actual people that populated the reconstructed city and aspects of their everyday life. Simulating such people and their lives in all their diversity is a costly and time-consuming exercise comparable in cost and efforts to development of a commercial video game, involving years of development and millions of dollars in funding. In this paper we present a novel approach that can significantly decrease the cost and effort required for simulating everyday life of ancient inhabitants of virtual cities, while still capturing enough detail to be useful in historical simulations. We show how it is possible to manually design a small number of individual avatars and then automatically generate a substantially large crowd of virtual agents, which will live their lives in the simulated city, perform choirs and rituals as well as other routine activities that are consistent with their social status. The key novelty of our approach that enables simulating such sophisticated crowds is the combination of physiological needs - for generating agent goals, emotions and personality - for choosing how to fulfil each goal and genetically informed propagation of appearance and personality traits - to propagate aspects of appearance and behaviour from a small sample of manually designed individuals to large agent groups of a desired size. The usefulness of our approach is demonstrated by applying it to simulating everyday life in the ancient city of Uruk, 3000 B.C.**[1]

I. Introduction

Using 3D visualisation in reconstruction of lost sites of high historical significance has become a popular way of communicating the results of years of research conducted by archaeologists and historians to general public [1]. Initially, such works were predominantly focused on reconstructing destroyed or partially destroyed architecture (e.g. Roman Colosseum) [2]. Such reconstructions help to simulate significant historical sites in all their former glory and facilitate appreciation of what remained from that glory (normally in the form of ruins). With modern advancement in research in development, we are now reaching the stage when reconstructing a heritage site can become relatively cheap. A procedural approach to generating historically informed designs of high complexity can be automated by design grammars, so that a large city can be created in a matter of days. One of the well known examples of using this approach in historical reconstructions is the Rome Reborn project [3], where a virtual reconstruction of the entire city of ancient Rome in the period of 320 AD was procedurally generated.

While 3D simulation of buildings and artefacts provides a unique possibility for general audiences to examine the architectural details of the heritage site it still does not help an observer to understand how this site has been enacted in the past. Without being able to see ordinary people performing their daily choirs and rituals the observer is unable to immerse in the actual culture of the reconstructed society and have a complete picture about their way of life. It is possible to simulate such people using so-called "virtual agents". These virtual agents are essentially autonomous computer programs that are represented by human-like 3-dimensional figures (called avatars) that move around the reconstructed environment and simulate ancient citizens of the reconstructed site. Modern video games are a good illustration in regards to possibilities that arise with employment of such virtual agents in simulating human behaviour. But the cost of developing video games is enormous. For example, the estimated cost of developing Crysis 3, one of the popular modern video games, is $66 Million [4]. It's hard to imagine such level of spending when it comes to historical simulations, so populating a historical environment with virtual agents needs to be automated.

Aiming to achieve cost saving, some researchers do not model their societies at the level of individual agents, but employ "virtual crowds" [2]. While such crowds essentially consist of a large number of virtual agents, designing a crowd normally comes down to designing a few individuals and then replicating them a desired number of times with slight modifications so that the crowd appears to be diverse [5]. In regards to agent behaviour, crowds simulation techniques predominantly rely on automated algorithms for large scale obstacle and collision avoidance and individual agent behaviour is rarely complex enough to illustrate various aspects of daily life of the reconstructed society [5]. The state of the art in using agent crowds in historical simulations is outlined in [6] where a virtual City of Pompeii is populated

[1] See the prototype video at: http://youtu.be/ZY_04YY4YRo

with a large number of simulated people, who simply walk around the city avoiding collisions. In this work the virtual agents give a perception about the appearance of the ancient people who used to populate Pompeii, but these people are not involved in historically authentic interactions. So they play a role of moving objects and can only extend the atmosphere of the culture simulation, while offering little in regards to understanding everyday life in the simulated society.

A number of crowd simulation and crowd generation approaches appear in the literature but hardly any of them advance beyond having avatars moving around and carrying objects with them. Further in the paper we show how through simulation of physiological needs and motivations together with personality traits we can achieve much more sophisticated simulations of human behaviour. Furthermore, employing genetic methods for inheriting personality traits and appearance characteristics together with connecting virtual agents with formalisations of social roles and social norms allows for a similar level of complexity in crowd based simulations as seen in commercial video games.

The remainder of the paper is structured as follows. Section II presents motivation for selecting the combination of genetics, social norms, personality and physiological motivations as a way of advancing the state of the art in historical simulations. Section III presents our methodology for creating such simulations. Section IV shows how the aforementioned methodology was applied to a particular case study: simulating everyday life in the ancient city of Uruk, 3000 B.C. In section V we analyse the results of the Uruk study.

II. Approach

The essence of our approach to building historic simulations lies in automating generation of diverse ethnic crowds in terms of the appearance and behaviour of individual avatars, while allowing for a high degree of complexity in the resulting behaviour. Simulation of life in 3D reconstructed historical cities is a costly and time-consuming process, comparable in cost and efforts to development of a commercial video game (involving years of development and millions of dollars in funding [4]). But costs and effort can be decreased with automatic generation of population. This is a two-fold process, in which we need to generate the unique appearance and the behaviour of each individual. Unique appearance can be generated by mimicking the biological reproduction, as for example in [7]. One way of automatisation of behaviour is to represent individuals as autonomous virtual agents that can generate their goals and act upon them [8]. To generate such goals, we propose to use motivation, and in particular physiological motivation, such as hunger, thirst, fatigue and comfort. In this case agents generate their goals upon physiological trigger, e.g. getting hungry. If needed, other types of motivation can be employed, such as safety, love, or self-realisation [9] [10].

The problem with classical approaches to agents driven by physiological motivation is that in a historical simulation all such agents would follow the same circadian rhythm [11] (get hungry, thirsty at the same time), what leads to undesired, uniform behaviour. To avoid this we propose to configure *motivational modifiers*, which affect the decay rate of a given motivation. For example, a hunger modifier affects the pace in which an agent gets hungry. If such modifiers are different for every agent - then every individual follows its own circadian rhythm, executing goals at various time intervals, increasing believability of the simulated population.

A. Personality and Emotions

In classical Artificial Intelligence (AI), many agent-based simulations follow the BDI model [12] which assumes individual agents being active by continuously pursuing some goals. In order to achieve a goal each agent needs a plan. Such plans can be automatically generated using traditional planning techniques [13] [14]. Such planning techniques normally model perfectly rational behaviour, which is not always suitable for simulating humans as this results in emotionless, "robotic" behaviour. To avoid this we enrich agents with personalities and emotions, which affect their decisions when creating a plan for a current goal. This approach may even lead to emergent agent behaviour that appears to be closer to human-like reasoning. As an example, imagine a fisherman agent with no personality and emotions, who catches fish when it's hungry. The agent will fish until it succeeds, or until it dies of hunger, unless we manually specify a possible change of plans when hunger level raises to a critical value. In contrast, fishermen who have various personalities and emotions may produce quite diverse behaviour, which will be more human-like and believable. For example, a phlegmatic agent may continue fishing until it succeeds, while a neurotic fisherman may get easily frustrated when hungry and unsuccessful with fishing. This neurotic agent may "decide" to stop fishing when frustration level overwhelms the rational decision for fishing and will search for alternatives to feed, such as begging or stealing food. The decision whether to beg or steal would also depend on the agent's personality.

B. Social Norms and Institution

In the above example, fishermen represents a specific *social group* of the simulated population. Social groups combine certain classes of individuals that fulfill their goals in a similar way. Combining individuals into social groups allows us to define and program actions on a group level, rather than having to do this on individual level, reducing effort in defining crowd behaviour. In human societies, it is not uncommon for members of different social groups to interact with each other and even cooperate in order to fulfill their goals. For example, imagine a fisherman who has to trade fish with a spear-maker in order to replace his broken fishing spear (see Section IV). A common technique being used in AI to facilitate the kind of interactions between different social groups as in the example above is to employ Organisation-Centred Multi-Agent System (OCMAS) [15]. The OCMAS approach is to explicitly formalise social norms of the agent population and connect those norms to the social roles, which represent different population groups. Such social norms capture rules

Fig. 1: PotMaker Planning Example: AddWater → MakeClay → Work → Trade.

and protocols that drive agent interactions. As a result, agents can use these norms in reasoning to create plans for their current goal. This provides agents the ability to automatically perform their actions depending on their assigned social group.

C. Dynamic Planning

Once the social norms are formalised, the agents can use them in combination with dynamic planning techniques to construct their plans that lead to satisfying the goals that result from individual physiological needs of the agents. Rather than having a complete recipe provided for every situation the agent can encounter - the agent is simply given the list of possible atomic actions and has to find a way of combining those to reach its goals. Our dynamic planing solution relies on environment annotation. The virtual environment contains a number of objects that can potentially be used by virtual agents and those objects can be acted upon. Through text annotations, those object specific actions are associate with pre-condition and post-conditions. So, those annotations define how an agent is potentially able to achieve its goal through atomic actions, given all possible states. Figure 2 shows a fragment of such annotation (in XML format) that the agent uses for building its plans at runtime and Figure 1 illustrates the resulting plan.

Consider an example where our agent represents a pot maker, who is responsible for making clay pots. To make his living the pot maker can trade clay pots he produces for food, water, milk and other necessary products. Now imagine a situation where our pot maker agent is supplied with a goal ”HasFood”. The result of this goal should be the agent possessing food. In the case of dynamic planning, there is no prescribed set of actions that satisfy this goal, so the agent must conduct a search for a sequence of atomic actions that would make its state evolve to the desired state (HasFood). Let’s assume that the current state of the agent is ”WaterAvailable”. So, for satisfying the goal ”HasFood” the agent must search for a sequence of actions that lead from ”WaterAvailable” to ”HasFood” state. This search is conducted using a method called ”backwards state-space search” [3] where the agent starts with finding the actions that has its goal state (”HasFood”) as a post-condition (so after completing this action the agent will satisfy the goal). In our example such action is ”Trade”. But to perform this action the agent must fulfil the necessary pre-condition of ”Trade” (which is ”WorkDone”). So the agent continues with searching for an action that has ”WorkDone” as the post-condition. Such action is ”Work”. Again, action ”Work” has the pre-condition ”HaveClay” that must be satisfied, so the agent must continue its search until it gets to the action where the pre-condition maps to the current state of the agent ”WaterAvailable” (in our example such action is ”AddWater”). As the result of its planning the agent has a sequence of actions that lead from the goal state to its current state. Thus, the agent can simply reverse this sequence and then use it for making its state evolve from the current state to the goal state. The resulting plan in our example will be as follows: AddWater→MakeClay→Work→Trade→HaveFood. Some individual steps of this plan (e.g. Work) will depend on the particular role being played by the agent and will be performed following the social norms of the underlying institution.

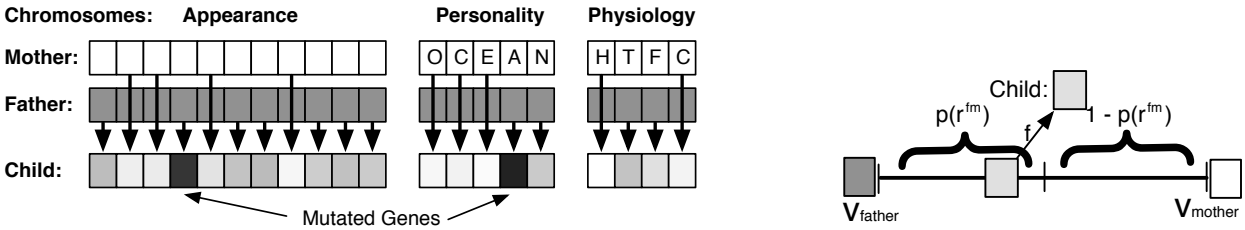
```
<?xml version='1.0' ?>
<actions>
    <action name="GetWater">
        <pre scene="" state="NoWater"></pre>
        <post scene="" state="WaterAvailable"></post>
    </action>

    <action name="AddWater">
        <pre scene="" state="WaterAvailable"></pre>
        <post scene="" state="ClayWaterAdded"></post>
    </action>

    <action name="MakeClay">
        <pre scene="" state="ClayWaterAdded"></pre>
        <post scene="" state="HaveClay"></post>
    </action>

    <action name="Work">
        <pre scene="" state="HaveClay"></pre>
        <post scene="" state="WorkDone"></post>
    </action>

    <action name="Rest">
        <pre scene="" state="WorkDone"></pre>
        <post scene="" state="Rested"></post>
    </action>

    <action name="Trade">
        <pre scene="" state="WorkDone"></pre>
        <post scene="" state="HasFood"></post>
    </action>
```

Fig. 2: Fragment of Environment Annotation XML file.

III. Methodology

Next we discuss the methodology that we suggest following for developing historical and social simulations following our

Miguel, Amblard, Barceló & Madella (eds.) Advances in Computational Social Science and Social Simulation
Barcelona: Autònoma University of Barcelona, 2014, DDD repository <http://ddd.uab.cat/record/125597>



(a) Genetic reproduction using "Fuzzy" operator

(b) Combining genes using fuzzy operator

Fig. 3: Using Genetic Operators to Form an Agent's Chromosome.

approach. This methodology is separated into several steps that facilitate automatic generation of intelligent agent crowds, where agents generate goals depending on physiological modifiers and plan their actions depending on their personality and in accordance with social norms.

A. Step 1: Design the base population

Base population represents the initial group of agents used to generate the rest of the crowd. This population has to define the fundamental visual properties of the resulting crowd. Therefore, for each ethnic group that will be generated, there must be at least one couple of avatars, where both individuals maintain the ethnicity-specific visual traits (e.g. asian eyes), while all other non-specific features (e.g. head shape) are varied. Following this approach, during genetic reproduction, ethnicity-specific features are carried on to the following generations [7], while diversity within ethnicity is assured.

This process requires a significant effort, as designers have to define all avatars with distinctive appearance and a library of related textures, clothing and attachments in order to ensure high variety. In order to reduce the effort, we propose to design and use *parametric avatars* [16] [7], which are avatars with visual features that can be modified using parametric values. For example, parameter "height" and "body fat" would modify the corresponding parameters of avatar body. Such parameter values of an avatar form genes combined in a chromosome used to reproduce children with diverse appearance.

To better understand how the diversity is achieved - we need to explain the process of genetic reproduction. In this process, an agent's appearance, motivational modifiers (in our case physiological modifiers), and its personality are encoded into "genes". As a result, these three groups of genes form three chromosomes, depicted in Figure 3a.

During reproduction, we take two parents and combine each of the three pairs of related parent chromosomes to produce the child's chromosome. We decide how many genes are inherited from the father and how many from the mother using a *father-mother ratio*. A *crossover* operator is responsible for combining chromosomes. Theory of genetic algorithms defines several crossover operators, i.e. split operator, but for our purposes, we define a specific *fuzzy operator*, that imitates the biological crossover using two pairs of chromosomes [17]:

Definition 1. *Given mother's chromosome* c^m *consisting of genes* $c^m = g_1^m g_2^m \dots g_n^m$*, the father's chromosome* c^f *consisting of genes* $c^f = g_1^f g_2^f \dots g_n^f$*, the parent gene selector function* $s^i_{r^{fm}} : 2^G \to \{0, 1\}$ *which for position* i*, where* $0 \leq i \leq n$*, selects either mother or father gene depending on probability given by the father-mother ratio* r^{fm} *and the fuzzy function* $f : \mathbb{I} \to \mathbb{R}$ *which for gene on position* i *selects a random value in the interval given by* $f(i) = [s(i), (g_i^m - g_i^f)/2]$*, we define a* ***fuzzy*** *crossover operator* $\oslash : C \times C \to C$ *as* $c^m \oslash c^f = f(1) \cdot f(2) \dots f(n)$.

Fuzzy operator creates a new gene value by selecting a random value from the interval defined by the gene values of the parents and depending on the specified father-mother ratio takes this value closer to father or mother gene. This process is depicted in Figure 3b, where r^{fm} means father-mother ratio and $p(r^{fm})$ means probability of selecting value from the interval, depending on r^{fm}.

Another important process of the biological reproduction is mutation, which is the driving mechanism of evolution and novelty in species. We mimic the mutation process by modifying the value of pre-defined number of genes to the value from outside of the previously mentioned interval. The result of genetic manipulations is a new chromosome using which we can reconstruct a new child, its appearance, physiological needs and a personality.

Once the appearance of the avatars representing the base population has been specified in a parametric fashion - a diverse crowd of a desired size can be automatically generated following the aforementioned genetic principles. The agents in the crowd will have diverse appearance, while at the same time the important ethnic features of their appearance will be preserved. In order to introduce diversity of their behaviour - further steps of the methodology need to be completed starting with the configuration of motivational modifiers.

B. Step 2: Configure motivational modifiers

Genetic approach is also used to diversify agent behaviour. For this purpose, *motivational modifiers* are encoded into genes of the chromosome. Therefore, in this step, for each member of the base population the *motivational modifiers* are specified. In case of physiological motivation, these modifiers relate to hunger, thirst, fatigue and comfort, and represent the decay rate in which agents are getting hungry, thirsty, tired and sleepy. To avoid an impression that every single agent follows the same day cycle and performs the same set of actions at

	Temptation	Gregariousness	Assertivity	Excitment	Familiarity	Altruism	Compliance	Modality	Correctness
Beg	0	0	-0.5	0	0	0	0.5	0	0.5
Work	0	0	0.5	0	0	0	0	0	1
Search	0.5	0	0.75	0.5	0	-0.25	-0.5	0	-0.5
Steal	1	0	1	1	0	-1	-1	0	-0.75

TABLE I: Personality facets of agent actions.

the same time, these values must be different for every agent from the base population. The more diverse these values are in the base population, the more diversity will be present in the circadian rhythms of the resulting crowd. As the result of completing this step we will prepare the base population for using them as the basis for the generation of highly diverse ethnic crowds. Each individual in this crowd will borrow some appearance and behaviour traces from the base population and will also use their motivational modifiers (with some degree of mutation) to introduce and element of diversity in the circadian rhythms of individual crowd members.

C. Step 3: Specify personality traits

While diverse motivational modifiers assure execution of actions at various times, agent personalities determine the kind of actions the agents will execute. In this step, for each member of the base population its personality is specified using the popular OCEAN model [18], which captures five personality traits: *openness, conscientiousness, extroversion, agreeableness and neuroticism*. Openness relates to imaginative, creative aspect of a person. Consciousness captures the ability to be organised and careful. Extroversion defines, how social and outgoing a person is. Agreeableness relates to ability to cope with people, friendliness and generosity. Neuroticism defines tendency for negative emotions and instability.

Combination of the OCEAN values defines a specific character. Explaining, how to define a specific character is out of scope of this work, therefore we direct interested readers to existing publications [19] [20]. For the purposes of this methodology, it is important that agents forming the base population have different personality values, so that during genetic reproduction their children will have a high degree of diversity in their emerging personalities. In Section V, we present how the diversity of parent personalities affects their children, and how it determines which actions they select as the result of having a certain personality type.

In order for agents to be able to select an action that is most relevant for their personality, such action has to be annotated by following *personality facets* [21]: *temptation, gregariousness, assertiveness, excitement, familiarity, straightforwardness, altruism, compliance, modesty* and *correctness*. Using values of personality facets, the agent selects an action that provides the highest utility for its personality type [19] [21]. See Table I for an examples of annotations for *work*, *beg*, *steal* and *search* actions.

Often, actions such as "work" have various meaning in the context of different social groups. Working for fishermen means to catch fish, while for pot makers it means to make pots. Therefore, in the next step of the methodology, the institution is specified, which defines all the social groups, their interactions and also defines the meaning and parameters of specific actions, e.g. determines how quickly a particular object satisfies hunger.

D. Step 4: Formalise Social Norms and Roles

To define social groups, their actions and interactions, an Electronic Institutions (EI), a well established Organisation-Centred Multi-Agent System (OCMAS) is specified. EI establishes what agents are permitted and forbidden to do as well as the constraints and the consequences of their actions [22]. In general, an EI regulates multiple, distinct, concurrent, interrelated, dialogic activities, each one involving different groups of agents playing different roles. Definition of an EI consists of the following components:

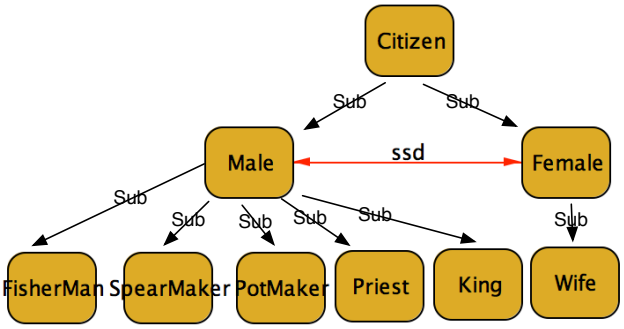


Fig. 4: Role hierarchy

First, a *dialogical framework* specifies social roles involved in the simulation and their hierarchy. Figure 4 depicts the role structure of the Uruk simulation (see Section V). Apart from the role structure, the dialogical framework defines ontology, a common language for communication between agents.

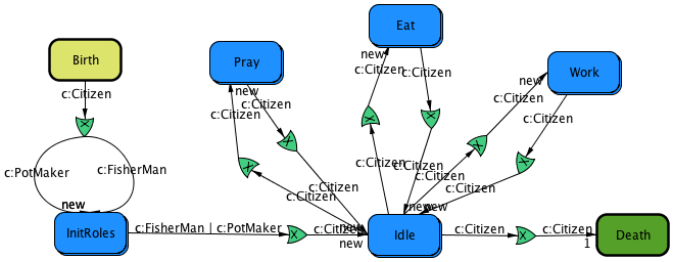
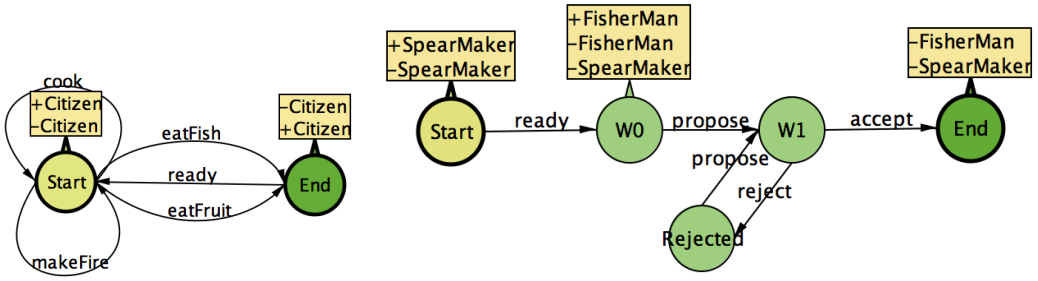


Fig. 5: Performative structure

Second, a *performative structure* isolates specific activities (also called scenes) that can be performed within an Electronic Institution. It defines how agents can legally move among different scenes (from activity to activity) depending on their role. Furthermore, a performative structure defines when new scene executions start, and whether a scene can be multiply executed at run time. A performative structure can be regarded



(a) Eat scene protocol (b) Trade scene protocol

Fig. 6: Scene protocols in an Electronic Institution.

as a graph whose nodes are both scenes and transitions (scene connectives), linked by directed arcs (See Figure 5).

The type of transition allows to express choice points (Or transitions) for agents to choose which target scenes to enter, or synchronisation/parallelization points (And transitions) that force agents to synchronise before progressing to different scenes in parallel. The labels on the directed arcs determine which agents can move to which scenes.

Third, for each activity (scene), interactions between agents are articulated through agent group meetings expresses as *scene protocols*, which follow well-defined interaction protocols, whose participating agents may change over time (agents may enter or leave). A scene protocol is specified by a directed graph whose nodes represent the different states of a dialogic interaction between roles (See Figure 6). Its arcs are labelled with illocution schemes (whose sender, receiver and content may contain variables) or time-outs.

Definition of EI is fundamental to agent reasoning and our dynamic planning algorithm that constructs a list of actions to fulfill the current goal by finding a path (sequence of actions) that make the agent go into the desired scene and reach a desired state within this scene. An institution provides agents with knowledge about possible actions that can be performed. The next step of the methodology provides means of visualising these actions in the virtual world.

E. Step 5: Adaptation and Annotation of the Environment

For purposes of visualisation, institutional actions must have corresponding objects, animations and scripts. In this step, objects of the virtual world related to such actions are created and annotated with specific meta-data, so that agents know that a connection between institutional illocutions and objects is established. Agents use annotations in their planning, which is affected by the current state of the environment. Therefore, interactive objects have to contain information on what action they provide and what are the action parameters [23].

Adaptation and annotation of the environment is the last step that requires manual input. In this last step we generate the population of the simulation and make it act within the simulated virtual environment.

F. Step 6: Generating the Population

Generation of population is a fully automatic process, where the desired number of "children" is generated from the base population using genetic approach described in Section III-A. Initially, children are only sets of chromosomes and their appearance has to be reconstructed in a given virtual world. Once connected to the virtual world, they start automatically generate goals and act upon them by using our dynamic planning approach in combination with the Electronic Institution that was defined on Step 4.

IV. Case Study: Uruk 3000 B.C.

In order to highlight the key aspects of our approach, we have applied it to simulating one of the humanity's first cities - the city of Uruk 3000 B.C. To further address the agility of our approach, we apply our methodology first to Second Life[2], a well known virtual world platform, and then to Unity 3D[3], the popular game engine. The Second Life simulation aims to present the life of Uruk to wide public, using well-know virtual world platform with many existing users. Users of Second Life can experience the simulation by conveniently connecting to it from any place in the world and enjoy the multiuser aspect and role-playing features. In contrast, the Unity simulation is intended to be used on desktop computers and doesn't have multiuser support. Due to its network dependance, the drawback of Second Life is in its lacking capability of handling large societies of intelligent agents (or non-playable characters, NPCs). On the other hand, Unity 3D provides the possibility to execute large societies. The size of the society is limited by the computational capability of the hardware, on which the game is executed. In this section we describe how our methodology facilitates deployment of sophisticated historical simulation to both platforms and compare their workload estimates.

A. Preparation: Designing the World

Before we can apply our methodology, we need to design the 3D environment of the simulation. In Second Life, we started with an existing 3D model of the city (mainly designed by our project partners from the Federation of American Scientists) that included key buildings, plants, animals and terrain. This model was informed by archaeologists and provides some level of historical accuracy.

For Unity 3D, we have recreated this 3D model in Google Sketchup and Blender. Furthermore, we have modelled historical objects used by various crafts belonging to the epoch.

[2]http://secondlife.com (last visited 03/2014)
[3]http://unity3d.com (last visited 03/2014)

Miguel, Amblard, Barceló & Madella (eds.) Advances in Computational Social Science and Social Simulation
Barcelona: Autònoma University of Barcelona, 2014, DDD repository <http://ddd.uab.cat/record/125597>

(a) Second Life (b) Unity 3D

Fig. 7: 3D Visualisation of the city of Uruk 3000 BC

These objects include beds positioned on roofs, various chairs and tables, pots for cooking, market equipment, pottery ring, spears, and spare spear parts, fisherman boats and rows. 3D design requires a lot of effort, and the preparation step took significantly longer then design and execution of the city population. Figure 7 compares the visualisations in both, Second Life and Unity 3D. Figure 8 shows the market, executed in Unity 3D, with several, custom designed objects.

Fig. 8: 3D design of the Uruk market with live avatars

With the static 3D design of the environment in place, we can start applying our methodology an populate this environment with autonomous agents.

B. Step 1: Design the base population

When defining the base population is Second Life, we considered only one ethnic group of Uruk citizens. Therefore, we designed only two members of the base population portrayed in Figure 9a and Figure 9b, using which we have generated the rest of the population.

Figure 9c, depicts a child generated with a low level of mutation. This child carries visual traits from both parents, having mother's nose, but father's mouth. Figure 9d depicts the child of the same parents, but with high level of mutation. Some visual traits are still visible (e.g. nose, jaw shape), yet, there are new emergent visual features, such as skin colour.

In Unity 3D we have applied a slightly different approach and generated the base population using the *genotype rules* [7]. Using such rules we can specify a racial or ethnic profile, which limits gene values only to the specific range. For example, we can specify what shades of skin colour can be used, what is the approximate size of the nose, what is the range of person height and so on. Yet, this approach can only generate avatars belonging to the same race/ethnic and does not allow us to generate intra ethnic avatars. Since we are generating avatars belonging to the ancient Uruk ethnic, this is not a problem.

(a) Father (b) Mother

(c) Child (d) Mutation

Fig. 9: Generating crowd appearance in Second Life

Figure 10 depicts the sample of ten avatars generated from the initial population of five avatars. In the base population we have two ethnics, Caucasian (Adam and Bea) and Sumerian

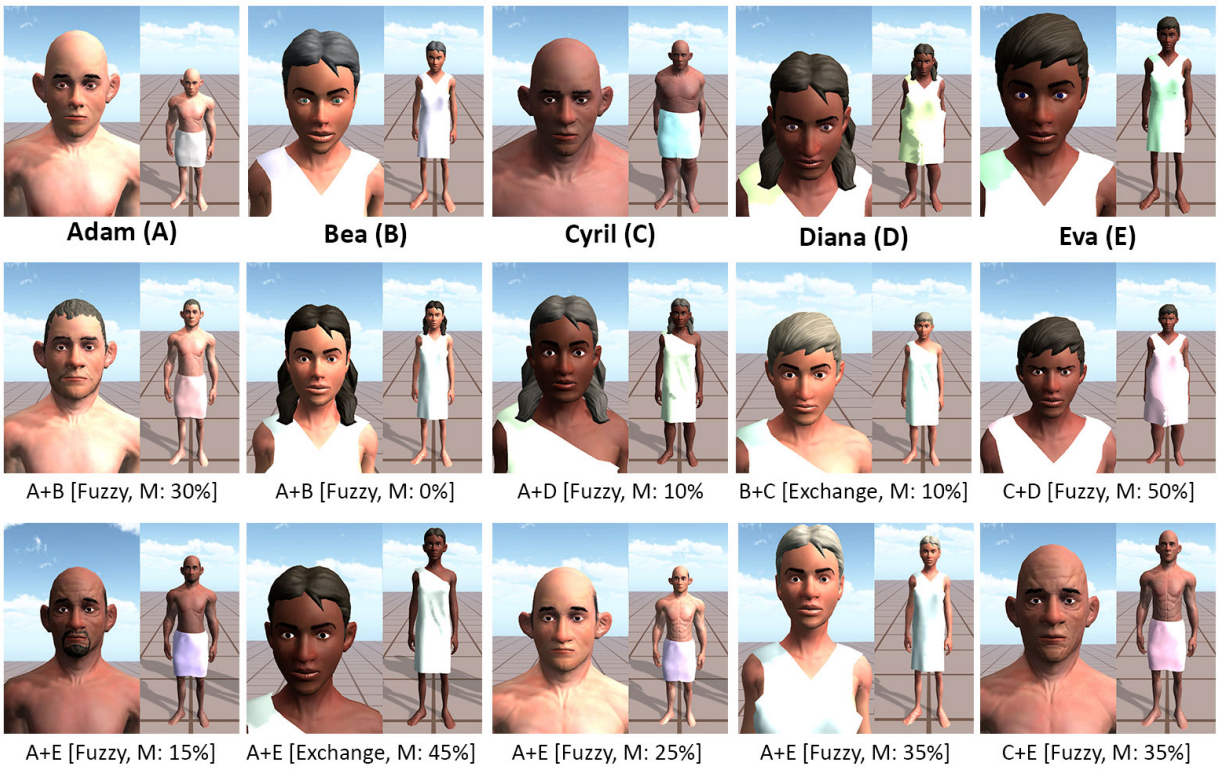


Fig. 10: Detail of the crowd generated for Unity 3D.

(Cyril, Diana and Eva). We have used western names for the Sumerian population only for convenience, in order to code them alphabetically by the first letter in their name (A-E). Generated children are named by coded names of their parents, the crossover operator and the mutation level used during generation process. To portray the preservation of ethnic features we have designed all members of the Sumerian population with bigger, distinctive noses and darker skin colour, while caucasian population has smaller noses and and lighter skin colour. Child of C+D in the first row and C+E in the second row obviously carry on only the Sumerian features, although C+D shows also very distinct features, due to the high level of mutation that has been used.

Interesting result is in the second row, where we depict four different children of A+E, each of them visually distinct, yet clearly carrying features from both father and mother. Two children are small, with lighter skin or bigger ears as their father, others are taller, or with darker skin and smaller ears as their mother.

Figure 12 depicts the society of 150 avatars, generated from the base population belonging to the same ethnic. As a result, none of the generated avatars carries caucasian traits and skin colour. The apparent difference is the overall graphic quality, which is prevailing in Unity 3D.

C. Step 2: Configure motivational modifiers.

Base population serves not only to generate avatars with unique appearance, but also with a unique (or non-uniform) behaviour. As a result, in the next step, we defined the physiological modifiers of the base population. We set various decay rates for hunger, thirst, fatigue and comfort for each member of the population. Avatars generated from the base population will obtain varied and mutated values of these modifiers. Since each modifier will have a different value, avatars will become hungry or tired in distinct intervals, executing their actions non-uniformly. Figure 11 shows the graphical user interface, that facilitates the specification of physiological properties.

D. Step 3: Specify personality traits

Physiological motivation solves the (when) problem of uniformity, when agents execute their actions at various time frames. On the other hand, having avatars with distinct personalities solves the (what) problem of uniqueness, when agents perform actions matching behavioural profile. As a result, we define personalities for each agent using the OCEAN model. Figure 13 shows the graphical user interface, that facilitates the specification of personality properties. In Section V we describe the setups for personalities that were used.

Apart from the definition of agent personalities, we annotated all actions and relate them to a specific personality, using personality facets (see Section III-C). Table I shows four actions that Uruk agents perform to satisfy the goal of "eating". In this table, there are four actions, i.e. beg, work, search and steal, and nine personality facets, e.g. temptation, gregariousness, assertivity with valued ranging from -1 (low) to 1 (high). These facets work as modifiers used to calculate

Fig. 12: Overview of the crowd generated for Unity 3D.

Fig. 11: Defining physiological properties of an avatar.

utility of a given action in relation to a specific personality. The higher the utility, the more probable is that the action will be selected. "Stealing" action is defined for agents with more aggressive personalities (very low correctness, low altruism), "begging" for agents with low-confidence (very low assertivity, higher correctness) and "working" and "searching" for more neutral personalities with varying sense of correctness.

Fig. 13: Defining personality properties of an avatar.

E. Step 4: Formalise Social Norms and Roles

Next, we defined all components of the Electronic Institution, with roles of fisherman, spear-maker, pot-maker, priest, king and wife (see Figure 4). All of these roles are sub-roles of *citizen*, which holds all common properties for all roles, e.g. inventory of owned items.

Then, we defined possible actions of agents in specific scene protocols (see Figure 6). For current roles we defined pray, eat, make spear, make pot, trade and fish protocol. Make spear, make pot and fish protocol belong to the scene "Work", and agents select the correct protocol based on their role. Most of these protocols only command a single agent what actions need to be performed to achieve its goal. The exception is the fishing protocol, which defines collaborative actions for two agents, where one agent has to row a boat, while the other is fishing with a spear. Therefore, fishermen always have to agree to go fishing in pairs.

Finally, we grouped scene protocols in a performative structure (see Figure 5), which restricts execution of actions in scenes to specific roles.

F. Step 5: Adaptation and Annotation of the Environment

For all actions and interactions, we have recorded animations, such as begging or stealing food, using motion capture and copied them in ".BHV" format to Second Life and with the help of Blender 3D, we have converted ".BVH" file to Unity 3D. Recording animations and their subsequent processing in any platform is a very delicate task and usually requires professional crew and equipment. Since we had no such possibility our own acting performance sufficed.

Moreover, since 3D object carry no meta-information on their possible purpose, we have added related objects to the virtual world model and annotated the environment so that agents can use them in their planning. For example, agents use the 3D object camp fire to cook their food. Therefore, we annotate that this 3D object provides action (illocution) "cook" from the scene "Eat" (annotated as action:Eat.cook). As a result, when agent plans its action, it knows that it has to interact with camp fire object to perform the "Eat.cook" action. In another example, we annotated a pottery ring with "action:Work.PotMaker.makePot", which defines that pottery ring provides action makePot in the scene protocol "PotMaker" from the scene "Work".

Fig. 14: Everyday Life in the City of Uruk 3000 B.C.

The underlying electronic institution puts these annotated objects into context and allows the agent to associate how its current physiological needs can be addressed using the help of particular objects, avatars or actions in the virtual environment.

G. Step 6: Generating the Population

In the last step, we generated a population of agents and integrated these into Second Life and Unity. Each agent had a unique appearance and automatically started fulfilling its goals. Agents were correctly selecting their goals based on their physiology, executing them at various intervals due to different physiological modifiers, planing their actions based on their personality and social norms, and executing them in the simulated environment. Figures 14, 15, 16 show some virtual agents from the resulting simulation.

Fig. 15: Praying Ritual

V. Evaluation

In this section we analyse our results and estimate the effort (in hours) needed to set-up and execute the simulation of Uruk in both environments. Then, we describe two experiments, that evaluate the diversity of generated agent behaviour.

A. Second Life

We estimate that the total time spent on completing the case study from Section IV was close to 7 days. The process was relatively fast as we have already had a model of the city and we focused only on generating the population. Step 1, definition of base population took us three days, where most of this time was spent on modelling clothing and attachments for avatars. Second Life provides parametric avatars with possibility to change more than 200 visual features. Therefore designing the body of the avatars took us only a couple hours per avatar. Steps 2 and 3, in our case took only one hour to complete, as the physiological modifiers and personality were defined only focusing on having wide range of values (rather than trying to achieve some pre-determined global personality skew in the resulting population). Step 4, definition of institution took 1 day, during which we designed all scenes and a performative structure and tested agent interactions. Also, we studied how to set-up the personality facets of personality-based actions. Step 5 took a lot of effort and time, in total 4 days. During this time we recorded and tuned all the animations, designed all interactive objects (e.g. pot-making ring) and scripted their behaviour. Step 6 is fully automatic, generation of 100 agents took a few seconds, visualisation of each avatar in Second Life takes about 30 seconds per avatar.

Fig. 16: Uruk School

B. Unity 3D

We estimate that the total time spent on completing the case study from Section IV was close to 25 days. The increase in time is due to the fact, that we needed to re-create manually the 3D design of the city (7 days) as well as all 3D objects (5 days) and avatar clothing (6 days). Since we were able to re-use animations from Second Life, we only needed to convert them from .BVH format to .FBX format in Blender 3D. Converted animations had to be adjusted and programmed to be used with Unity (i.e. Mecanim[4]). The conversion and animation adjustments took us 2 days. Then, we have annotated the environment with meta-data used by agents during reasoning about possible plans to accomplish their goals. In this case annotation is done directly in Unity 3D, via custom MonoBehaviour objects. Once all the platform-specific steps were taken we were able to reuse all other information from the Second Life setup. This information included the definition of the institution (which is exactly the same), personality setups for the base population, their daily plans and related cultural information for the institutional roles. It is here where our methodology proves its strong re-usability capability. As a result it took us only 1 day, to adjust steps 2-6 to Unity 3D.

C. General Methodology

Using our methodology, in combination with modern game engines and 3D virtual worlds, we significantly cut down the time to populate historical 3D simulations. The drawback of our approach is that we rely on parametric avatars with ability to modify the avatar appearance and clothing using declarative (visual) parameters. But, this is not a major issue, since we already possess the technology for Unity 3D and Second Life, and other game engines offer similar functionality, although in the form of paid plugins.

Having parametric avatars and employing our genetic approach we can generate unique, ethnic avatars in a very little time. Using motion capture, we can easily animate these avatars and believable results depend only on exact historical data and acting skills. Furthermore, using the Electronic Institution technology, we can declaratively specify the social structures and interaction protocols, used by agents to automatically reason about their possible actions. Electronic institution can be tweaked during the simulation runtime, decreasing the debugging efforts in comparison to traditional approach, where simulation has to be restarted after every change. As the result, with limited effort we receive a simulation in a semi-automatic way, where the degree of complexity of the actions the agents perform is much higher than what can be achieved with classical crowd simulation techniques.

D. Generating children of parents with diverse personalities.

To test the validity of generating agents with various behaviour, we performed two experiments. In the first experiment, we set-up diverse personalities of parents, where one parent had very low confidence, while the other was very aggressive (see Figure 17a). When hungry, the first parent would choose to beg, the second one to steal. Then, we have generated their 100 children, with father-mother ratio set to 30% (agents will have 70% of their genes closer to their mother). Figure 17b depicts the highly varying personality profiles of their children. We generated agents decide what to do when hungry and observed emerging behaviour of searching for food and working in 40% of generated children (see Figure 17c). Only a few children decided to steal as the father-mother ratio was in favour of the mother.

[4]https://unity3d.com/unity/animation (last visited 04/2014)

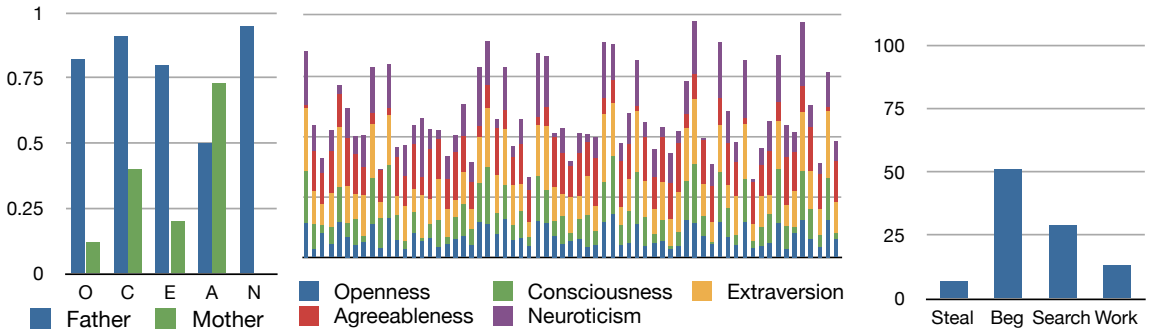
E. Generating children of parents with similar personalities.

In the second experiment we set-up father and mother with similar personalities (see Figure 18a) and during generation applied a high level of mutation (25%). We observed the children personalities and actions, depicted in Figure 18b. Generated children had very similar personalities, with occasional exceptions, due to mutations. In this experiment father choses to search for food, while mother choses again to beg. Having the same father-mother ratio (30%), most of children decide to beg, just like their mother (see Figure 18c). Several mutated children decided to work.

The above experiments showed that having a base population with diverse personalities leads to generating children with diverse behaviour. Parents with similar personalities result in their children having similar personalities and predominantly showing the same behaviour, unless they undergo mutation.

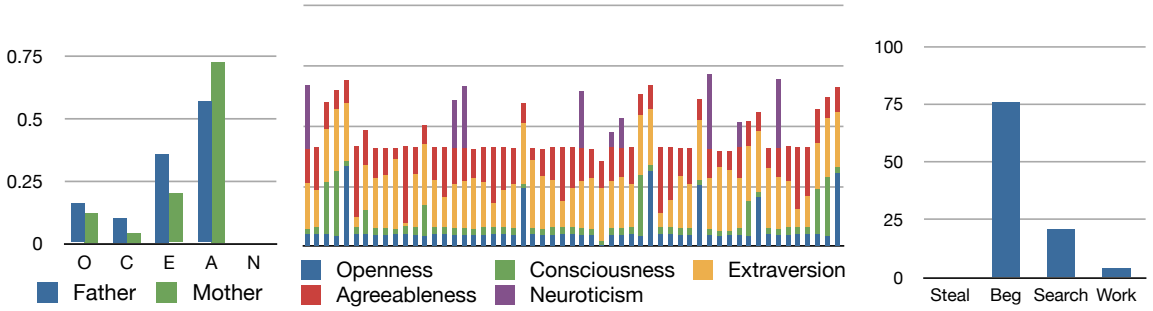
VI. Conclusions

In this work, we presented a methodology for generating crowds for the purposes of social simulations. This methodology is using genetic operations to produce individuals with unique appearance and behaviour. We have separated the methodology into six steps. First step is the definition of the base population, which specifies the visual traits of the whole population, although using mutation we may achieve novelty during generation. Second step is the definition of motivational modifiers, where motivation serves as the goal selection mechanism. In our case, we used physiological needs as the main motivation. Third step is the definition of personality traits, where personality affects agents decisions during planning and agents select actions that best match their profile. We are using well-established OCEAN model for the personality definition. In the context of social simulations, agents belong to specific social, ethnic or cultural groups and have to obey specific social norms. Therefore, fourth step is the definition of the social system and norms, in our case using Electronic Institutions. The fifth step is the adaptation and annotation of the environment that reflects all actions specified in the electronic institution. Agents are using these annotation to automatically plan their actions and interact with the environment. Following these steps results generating a diverse agent population having a high degree of variety in their appearance and behaviour, while also demonstrating substantially high degree of complexity of actions being performed by the agents.



(a) Parents (b) Children personalities (c) Actions

Fig. 17: Exp. 1: Children of parents with opposite personalities (no mutation).



(a) Parents (b) Children personalities (c) Actions

Fig. 18: Exp 2: Children of parents with similar personalities (mutation 25%).

References

[1] A. C. Addison, "Emerging trends in virtual heritage," *IEEE MultiMedia*, vol. 7, no. 2, pp. 22–25, 2000.

[2] D. Gutierrez, B. Frischer, E. Cerezo, A. Gomez, and F. Seron, "AI and virtual crowds: Populating the Colosseum," *Cultural Heritage*, vol. 8, no. 2, pp. 176–185, 2007.

[3] K. Dylla, P. Mueller, A. Ulmer, S. Haegle, and B. Fisher, "Rome Reborn 2.0: A Case Study of Virtual City Reconstruction Using Procedural Modeling Techniques," in *37th Proceedings of the CAA Conference*. Oxsford: Archaeopress, 2009, pp. 62–66.

[4] J. Gauder, "Crysis 3 cost $66 million to make, can next gen sustain such budgets?" GameChup Video Games News at http://www.gamechup.com/crysis-3-cost-66-million-to-make-can-next-gen-sustain-such-budgets/, 1 March 2013.

[5] D. Thalmann and S. R. Musse, *Crowd Simulation, Second Edition*. Springer, 2013.

[6] J. Mam, S. Haegler, B. Yersin, P. Muller, Thalmann, Daniel,, and L. Van Gool, "Populating Ancient Pompeii with Crowds of Virtual Romans," in *8th International Symposium on Virtual Reality, Archeology and Cultural Heritage - VAST*, 2007.

[7] T. Trescak, A. Bogdanovych, S. Simoff, and I. Rodriguez, "Generating diverse ethnic groups with genetic algorithms," in *Proceedings of the 18th ACM symposium on Virtual reality software and technology*, ser. VRST '12. New York, NY, USA: ACM, 2012, pp. 1–8. [Online]. Available: http://doi.acm.org/10.1145/2407336.2407338

[8] S. Vosinakis and T. Panayiotopoulos, "Simhuman: A platform for real-time virtual agents with planning capabilities," in *In Proceedings of the IVA 2001 Workshop*. SpringerVerlag, 2001, pp. 210–223.

[9] A. H. Maslow, R. Frager, and J. Fadiman, *Motivation and personality*. Harper & Row New York, 1970, vol. 2.

[10] C. P. Alderfer, "An empirical test of a new theory of human needs," *Organizational behavior and human performance*, vol. 4, no. 2, pp. 142–175, 1969.

[11] F. Levi and U. Schibler, "Circadian rhythms: Mechanisms and therapeutic implications," *Annual Review of Pharmacology and Toxicology*, vol. 47, no. 1, pp. 593–628, 2007, pMID: 17209800. [Online]. Available: http://www.annualreviews.org/doi/abs/10.1146/annurev.pharmtox.47.120505.105208

[12] A. Rao and M. Georgeff, "BDI-agents: from theory to practice," in *First International Conference on Multi-Agent Systems (ICMAS-95)*, San Francisco, USA, 1995, pp. 312–319.

[13] O. Shehory, S. Kraus, and O. Yadgar, "Emergent cooperative goal-satisfaction in large-scale automated-agent systems," *Artificial Intelligence*, vol. 110, no. 1, pp. 1–55, 1999.

[14] L. Braubach, A. Pokahr, D. Moldt, and W. Lamersdorf, "Goal representation for bdi agent systems," in *Programming multi-agent systems*. Springer, 2005, pp. 44–65.

[15] J. Vazquez-Salceda, *The Role of Norms and Electronic Institutions in Multi-Agent Systems: The HARMONIA Framework*. Birkhäuser, 2004.

[16] M. Lewis, "Evolving human figure geometry," Ohio State University, Tech. Rep. OSU-ACCAD-5/00-TR1, ACCAD, 2000.

[17] R. C. C. Vieira, C. A. Vidal, and J. B. C. Neto, "Simulation of genetic inheritance in the generation of virtual characters." in *VR*, B. Lok, G. Klinker, and R. Nakatsu, Eds. IEEE, 2010, pp. 119–126.

[18] L. R. Goldberg, "An alternative "description of personality": The Big-Five factor structure," *Journal of Personality and Social Psychology*, vol. 59, no. 6, pp. 1216–1229, Dec. 1990. [Online]. Available: http://psycinfo.apa.org/doi/index.cfm?fuseaction=showUIDAbstract&uid=1991-09869-001

[19] C. Bartneck, "Integrating the occ model of emotions in embodied characters," in *Workshop on Virtual Conversational Characters*. Citeseer, 2002.

[20] B. R. Steunebrink, M. Dastani, and J.-j. C. Meyer, "The OCC Model Revisited," in *Emotion and Computing*, D. Reichardt, Ed., Paderborn, Germany, 2009.

[21] P. J. Howard and J. M. Howard, "The big five quickstart: An introduction to the five-factor model of personality for human resource professionals." 1995.

[22] M. Esteva, "Electronic institutions: From specification to development," Ph.D. dissertation, Institut d'Investigació en Intelligència Artificial (IIIA), Spain, 2003.

[23] T. Trescak, "Intelligent generation and control of interactive virtual worlds," Ph.D. dissertation, Autonomous University of Barcelona, 2012.