

SIMULATING GAZE BEHAVIOR OF VIRTUAL CROWDS BY PREDICTING INTEREST POINTS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

By
Umut Ağıl
July 2016

SIMULATING GAZE BEHAVIOR OF VIRTUAL CROWDS BY
PREDICTING INTEREST POINTS

By Umut Ađıl

July 2016

We certify that we have read this thesis and that in our opinion it is fully adequate,
in scope and in quality, as a thesis for the degree of Master of Science.

Uđur Gdkbay (Advisor)

zgr Ulusoy

Veysi İřler

Approved for the Graduate School of Engineering and Science:

Levent Onural
Director of the Graduate School

ABSTRACT

SIMULATING GAZE BEHAVIOR OF VIRTUAL CROWDS BY PREDICTING INTEREST POINTS

Umut Ağıl

M.S. in Computer Engineering

Advisor: Uğur Güdükbay

July 2016

Creating realistic crowd behavior is one of the major goals in crowd simulations. Simulating gaze behavior and predicting interest points of virtual characters play a significant role in creating believable scenes, however this aspect has not received much attention in the field. This study proposes a saliency model, which enables virtual agents to produce gaze behavior. The model measures the effects of distinct pre-defined saliency features that are implemented by examining the state-of-the-art perception studies. When predicting an agent's interest point, we compute the saliency scores by using a weighted sum function for other agents and environment objects in the field of view of the agent for each frame. Then we determine the most salient entity in the virtual scene according to the viewer agent by comparing the scores. We execute this process for each agent in the scene, thus agents gain a visual understanding about their environment. Besides, our model introduces new aspects to crowd perception, such as perceiving characters as groups of people, gaze copy phenomena and effects of agent velocity on attention. For evaluation, we compare the resulting saliency gaze model with real world crowd behavior in captured videos. In the experiments, we simulate the gaze behavior in real crowds. The results show that the proposed approach generates plausible gaze behaviors and is easily adaptable to varying scenarios for virtual crowds.

Keywords: crowd simulation, saliency, gaze behavior, perception, interest point detection, gaze copy.

ÖZET

İLGİ NOKTALARINI TAHMİN EDEREK SANAL KALABALIKLAR İÇİN BAKIŞ DAVRANIŞI SİMÜLASYONU

Umut Ağıl

Bilgisayar Mühendisliği, Yüksek Lisans

Tez Danışmanı: Uğur Güdükbay

Temmuz 2016

Gerçekçi kalabalık insan grubu davranışı yaratmak kalabalık simülasyonlarının en temel amaçlarından birisi olarak yer alır. Sanal karakterlerin bakış davranışını simüle etmek ve ilgi çekici noktaları tahmin etmek inandırıcı sahneler oluşturmada önemli rol oynamaktadır. Fakat bu yaklaşım bu zamana kadar bilgisayar grafiği alanında yeteri kadar ilgi çekmemiştir. Bu çalışma sanal karakterlerin bakış davranışı üretmesini sağlayan bir dikkat çekicilik modeli sunmaktadır. Model algı üzerine yapılmış güncel çalışmaları ele alarak dikkat çekici unsurlar belirler ve bu unsurların etkilerini ölçer. Bir karakterin ilgisini çeken noktayı tahmin ederken sahnede karakterin görüş alanında bulunan diğer karakterler ve nesnelerin dikkat çekme skorları bir ağırlıklı toplam fonksiyonu ile hesaplanır. Ardından bu skorlar karşılaştırılarak izleyici karaktere göre sahnedeki en dikkat çekici varlık bulunur. Bu işlem sahnedeki her bir karakter için hesaplanır ve böylece tüm karakterler çevreleri hakkında görsel bir algıya sahip olurlar. Bunun yanında, modelimiz kalabalık algısına yeni bakış açıları da getirmektedir. Örnek olarak, kalabalıkların gruplar halinde ele alınması, bakış kopyalama, hızın dikkate etkisi verilebilir. Önerilen modelin değerlendirmesi için elde ettiğimiz dikkat çekme modelini videolardan elde ettiğimiz gerçek dünya kalabalık davranış verileri ile karşılaştırdık. Hazırlanan senaryolarda, gerçek kalabalık davranışlarının simülasyonunu yaptık. Elde edilen sonuçlar modelin gerçekçi bakış davranış sunan ve farklı senaryolara kolaylıkla uygulanabilir olduğunu göstermektedir.

Anahtar sözcükler: kalabalık simülasyonu, dikkat çekme, bakış davranışı, algı, ilgi çekici noktaların tespit edilmesi, bakış kopyalama.

Acknowledgement

First and foremost, I would like to express my gratitude to my supervisor Prof. Dr. Uğur GÜDÜKBAY. Without his guidance and assistance, it would not be possible to complete this thesis.

I also would like to thank to rest of my thesis committee, Prof. Dr. Özgür ULUSOY and Prof. Dr. Veysi İŞLER for evaluating this thesis.

Besides, I am grateful to Dr. Erdal Yılmaz for his advice with the development of the framework and the evaluation of the model.

Thanks to Tarık Yalçın and Seçkin Yalçın for their assistance on creating virtual scenes and preparing videos.

I would like to thank to Onur Polat for his companionship during this process and for his help in capturing videos.

I am grateful to my friends Volkan, Faruk, Emre, Kubilay, Gülfem, İstemi, Olcay, Serkan, Yusuf and all the people who have shared their valuable time.

Special thanks to the members of "Motor Oil Co.", who have always kept my spirit high, for all the goy they have given.

Finally, I must express my sincere gratitude to my family, who have supported me throughout my years of study and through the entire thesis process. This accomplishment would not have been possible without their unconditional love.

Contents

1	Introduction	1
1.1	Motivation	2
1.2	Research Questions	4
1.3	Contributions	4
2	Background and Related Works	6
2.1	Saliency in Computer Graphics	6
2.2	Crowd Simulations and Crowd Models	7
2.3	Attention Models and Applications on Crowds	9
2.4	Human Visual System and Gaze Behavior	11
2.5	Discussion	11
3	Simulating Interest Point Detection and Gaze Behavior	13
3.1	Interest Point Detection	14
3.1.1	Attributes of Characters and Objects	15

3.1.2	Saliency Score Computation Parameters	16
3.1.3	Saliency Scoring	17
3.2	Adjusting Gaze	25
3.2.1	Gaze Copy	27
3.2.2	Gaze Shifting Animation	28
3.2.3	Gaze Duration	28
4	Implementation of the Framework	30
4.1	Graphics Models	30
4.2	Animation	31
4.3	Architecture of the Simulation	32
5	Evaluation and Results	35
5.1	Scenario Simulations	35
5.1.1	Scenario 1	36
5.1.2	Scenario 2	36
5.1.3	Scenario 3	37
5.1.4	Scenario 4	37
5.1.5	Scenario 5	39
5.1.6	Scenario 6	41

5.1.7	Scenario 7	41
5.1.8	Scenarios 8	42
5.1.9	Scenarios 9	43
5.2	Performance	43
6	Conclusion	48
	Bibliography	50

List of Figures

1.1	Gaze behavior in popular computer games	3
4.1	The relationships between agent components.	32
4.2	Gaze decision logic.	33
5.1	Effects of proximity and periphery: (a) video, (b) simulation, and (c) simulation (weight=0).	36
5.2	Effects of agent velocity: (a) video, (b) simulation, and (c) simu- lation (weight=0).	37
5.3	Effects of agent attractiveness: (a) video, (b) simulation, and (c) simulation (agent's attractiveness=0.4).	38
5.4	Effects of distinctiveness, curiosity and height: (a) video, (b) simu- lation, (c) distinctiveness = 0, (d) curiosity=0, and (e) child height = 1.65.	39
5.5	Still frames showing the effect of agent shyness: (a) video frames, (b) shyness = 0.25, (c) shyness = 0.7.	40
5.6	Effects of object distinctiveness: (a) video, (b) simulation, and (c) simulation (weight=0).	41

5.7	Still frames showing the effects of gaze copy: (a) video frames and (b) simulation frames.	42
5.8	Effects of periphery: first row: the real video, second row: the simulation result.	43
5.9	Still frames from a 40 second video: left column: the real video, right column: the simulation result.	44
5.10	Graphs depicting (a) maximum frame computation times (msecs) and (b) frame rates (frames per second) for different number of agents.	46
5.11	A crowd simulation with 100 agents. Gaze behavior scripts are (a) disabled and (b) enabled.	47

List of Tables

3.1	Parameters for saliency scoring.	19
3.2	Parameters for gaze adjustment.	27
5.1	Simulation performance measurements where gaze behavior scripts are disabled.	45
5.2	Simulation performance measurements where gaze behavior scripts are enabled.	45

Chapter 1

Introduction

Real-time crowd simulation has been a great interest since years and creating highly realistic virtual crowds is a challenging topic in the area of computer graphics. Both in industrial and academic research studies that are concerned with virtual simulation and entertainment, generating more believable virtual characters has always been an important issue. Currently, with the advancement in computer science and technology, virtual scenes that contain hundreds of agents acting in a crowd can be simulated in real-time. And behaviors of these agents play a major role in improving the realism of the simulation. Most of the studies about crowd behavior are related to path planning, collision avoidance, autonomous agents, social interactions among groups, and personality models. However, the simulation of gaze behavior, which can contribute to the realism of virtual crowds, is still an area that needs further investigation.

The development of character behavior models that imitate humans in real-life situations have obtained a great attention in video games and in special effects for film, and most of the emphasis has been placed on path planning and agent navigation algorithms. Characters move, walk, run, fight and interact more realistically with evolving studies, however gaze behavior remains as a lacking area in these simulations. Especially, in video-games, even in the most popular ones, such as Assassin's Creed, Elder Scrolls and Witcher series (cf. Figure 1.1),

gaze behavior of non-playable-characters looks unrealistic. In animation movies production, artists need to determine the interest points of virtual characters, which is then adjusted frame by frame. In addition, working with crowds of large size brings a trade-off between realism and computational costs. Despite these problems, there are only a few studies dealing with the analysis and synthesis of gaze behavior.

The purpose of this research is to bring out more emphasis on the mentioned topic and to contribute to this area of computer graphics by providing a new gaze behavior model for virtual crowds. The proposed approach focuses on crowd simulation in virtual environments, preferably where characters are viewed in close range in the final rendered scene. Our proposed model is implemented as a module that can be attached to the Unity game engine.

1.1 Motivation

In simulating crowds having characters that are aware of their environment and other characters has a significant impact on enhancing the realism. Navigation and path planning algorithms can ensure this awareness up to a certain point by ensuring that characters avoid collisions with each other or scene objects. However, in the current literature most computer games simulating crowds neglect the implementation of virtual characters' gaze behavior or apply pre-adjusted gaze animations to characters.

The lack of realism in gaze behavior can be observed in non-playable characters (NPCs), which are controlled by AI mechanisms. In our observations on computer games with crowd simulations, we observe two types of gaze behaviors applied to NPCs. One approach is forcing characters to look at their forward direction and implementing head movements via blended animations. In this way, characters look like they continuously look at their environment. The other approach is making NPCs be aware of the playable main character, which enables characters to look at the player. Even though these methods can generate crowds with

gazing ability, resulting simulations still lack realism. NPCs either look at their environment randomly without considering the saliency of surrounding entities or they direct their gaze towards the player’s character for unrealistic durations. Moreover, gaze interactions between NPCs are generally omitted, except for pre-adjusted event animations, such as two characters stand for talking to each other. Figure 1.1 shows examples of gaze behavior in three popular games where the first two images (a, b) show characters that just look forward, the third image (c) contains NPCs that look at the player agent continuously, and the fourth image (d) shows a scene with pre-adjusted event and animations where passing agents direct their gaze to the predetermined area.



Figure 1.1: Gaze behavior in popular computer games: Assassin’s Creed (©Ubisoft, 2014), Elder Scrolls Skyrim (©Bethesda, 2011), and Witcher 3 (©CD Projekt RED, 2015). (a), (b) Agents look only forward or random directions. (c) NPCs continuously look at the player character. (d) A scene where events, animations and gaze behavior are pre-adjusted.

In development of animation movies, gaze behavior of characters are generally manually adjusted by an animation artist, by setting the gaze direction in certain frames. Certainly, artists can predict where the characters should look at, however

doing this process manually can be time consuming. Besides, a computational model provides a realistic prediction for interest points, which can be used as a baseline for gaze behavior.

We aim to increase the realism of crowd simulations by proposing a gaze behavior model for the virtual characters in the scene. Especially, aiming to address the mentioned problems, the proposed approach focuses on predicting interest points and gaze duration for characters during run-time.

1.2 Research Questions

By taking into consideration the mentioned problems in the game and animation industries, we aim to address the research questions that are outlined as follows:

- How does the characters in virtual crowds decide their interest points and be aware of their environment?
- By using a computational behavior model, can we generate realistic scenes, in which characters shift their gazes similar to real-life examples obtained from videos?

1.3 Contributions

The contributions of the thesis are as follows:

- We develop a computational model that simulates interest point detection in real-time crowd simulations with the state-of-the-art psychology, cognitive-science and computer graphics studies.
- We introduce a group-based approach in interest point detection methods

in which all characters belong to a group even if they walk alone. This approach provides us performance improvement and realistic implementation on perception of humans in groups [1].

- We incorporate the effects of characters' personality to their gaze behavior. Users can alter the gaze behavior of a character by adjusting the shyness and curiosity parameters.
- We examine the human visual system and causes of attention shifts in crowds and we shape our model accordingly. We apply ensemble encoding mechanism [1, 2] and incorporate the impact of walking speed on gaze behavior [3].
- We propose a model with adjustable parameters that allows designing scenes with different crowd norms where each norm represent a distinct society. A different crowd norm means that gaze behaviors and reactions of characters change from crowds to crowds such that a salient entity in a scene may not be that salient in another scene.
- We develop a gaze behavior framework that can be added as a plug-in to game engines (e.g., Unity [4]).
- We design virtual environments that simulate real-life examples to evaluate the proposed model.
- We show that by using our framework with just altering the parameters of characters and crowd norms, human gaze behavior in real-life scenarios can be simulated.

Chapter 2

Background and Related Works

2.1 Saliency in Computer Graphics

Human visual attention can simply be considered as the cognitive process to concentrate on a visual object. It has been studied for a long time by psychologists and cognitive scientists. The results obtained have been put to use in computer graphics studies and applications to improve the quality of rendered images [5, 6] and to simulate virtual human behaviors more realistically [7, 8].

When a person look at a scene, the brain cannot perceive the whole scene and analyze it instantly, instead only the area within two degrees range from the center point can be focused [9]. Next, the object identification stage starts and attention is directed serially to each object in a display [10]. It is indicated that the combination of two processes, which are bottom-up and top-down, forms human visual attention mechanism [11]. The bottom-up process is a stimulus-driven attention and depends on the properties of the objects. Human gaze is involuntarily directed towards the salient area, which possibly have a higher importance for that character. To give an example, the movement of a running man in a pedestrian group with slow pace can be a sign of danger. It is simply a difference of an area from its surroundings. The top-down process is a goal-driven

process and can be thought as dependent on given tasks or the prior knowledge of the viewer. It also takes role in determining longer-term cognitive strategies [12].

As one of the earliest methods, Itti et al. [13] apply bottom-up visual attention model on 2D images to compute saliency. The result is called *saliency map*, in which each pixel reflects a saliency value, i.e., how much the area possibly grabs attention. Saliency maps can successfully predict attention scores of a scene at a time from the viewer’s perspective. On the later studies, different features such as flickering [11], motion [11] or depth [14] are included to saliency map computations. On the other side, visual attention is not determined by just incoming inattentive reflexes resulted from bottom-up process. The studies of Longhurst et al. [15] and Cater et al. [9] are examples of different models that simulate top-down processes.

Sundstedt et al. [16] point out that the pure bottom-up process is not sufficient to compute a person’s gaze. They apply psychophysical experiments using a computer game to show that saliency maps are weak in task-oriented scenarios. Itti argues that the bottom-up and top-down processes are not mutually exclusive and the features of one can be integrated to those of the other [17]. Judd et al. [18] use face detection algorithms along with saliency maps. They use the idea that the human visual system searches for faces initially when they look at images. Another study [14] that combines these two processes, in which bottom-up factors computed with saliency maps and top-down factors are computed by spatial context of objects. When the object is close to the screen, it has a high saliency score. They compute saliency scores of objects instead of regions, which is what we use in our model.

2.2 Crowd Simulations and Crowd Models

Computer simulations study the masses and try to predict their behavior because the computer graphics technologies make it possible to generate multiple virtual characters. Reynold’s model with boids [19] is one of the earliest successful works

in this area. Moreover, modeling of crowds improves progressively and different aspects are studied in detail such as collision avoidance [20], autonomous agents [21], behaviors [22], or personalities [23].

Our work is especially related to the behavior of crowds, so we need to mention crowd models briefly. Alongside with physical crowds-people gathered in the same place- our main concern is primarily related to psychological crowds, in which individuals share a common social identity and their behavior can change with the social norms [24].

According to Templeton et al. [24], state-of-the-art psychological crowd models can be divided into two broad categories: *mass* approaches and *small group* approaches. In mass crowd approaches, individuals are considered as part of a large gathering. One subtype of this approach, “homogeneous mass”, assumes that all entities in the crowd have the same properties and share the same goal. The work of Fang et al. [25] is an example of homogeneous mass approach, which simulates an egress situation and analyzes crowd densities. The other subtype is “mass of individuals” where individuals carry different properties. For example, Shie et al. [26] assign different attributes to individual agents to generate more realistic simulations. The second category, the small group approach, basically depends on the idea that crowds are formed by small groups and the model represents inter-group relations. This approach has a significant impact on the development of our study. The methods in this category can be divided into three subtypes: *non-perceptual groups*, *perceptual groups* and *cognitive groups*.

To begin with, “non-perceptual groups” approach consider groups as homogeneous ensemble of individuals with identical properties. Entities that constitute the group act together as they are a big single entity. Groups preexist and members do not have intra-group interactions. In their work, Dogbe et al. [27] assume that groups are formed with a small numbers of members who already know each other and tend to move together. In “perceptual groups” approach, the main difference is that group members are aware of other members of their own group. Social norms are involved in determining group behavior and individuals may change the group they belong, e.g., [28]. The last approach, “cognitive groups”,

is similar to “perceptual groups”, with one difference: individuals that share common properties are gathered in the same group. Moreover, these properties of individuals can change during the simulation and this may result in new groups, e.g., [29].

In addition to these grouping types, personalities of characters play an important role in structuring the behavior of the crowd. By using Eysenck’s three-factor personality model, Guy et al. [30] develop a mapping between personality descriptors and simulation parameters to generate heterogeneous crowd behaviors. Durupinar et al. [23] assign personalities to virtual characters with respect to the OCEAN personality model [31]. Then, they examine how small groups and global crowd behavior are influenced by changing the personality parameters. Both of these works demonstrate the use of personality in crowd behavior. According to Big Five personality traits, personality dimensions are defined as ‘openness to experience’, ‘conscientiousness’, ‘extraversion’, ‘agreeableness’, and ‘neuroticism’. In the proposed approach, two parameters-shyness and curiosity- are used to reflect personality, which can be related to ‘extraversion’ and ‘neuroticism’ for shyness and ‘openness to experience’ for curiosity.

2.3 Attention Models and Applications on Crowds

Simulating attention in virtual characters is a highly challenging task. There are varying studies that aim to model attention behavior realistically. Chopra-Kullar and Badler do one of the earliest works on this area (where to look), which uses bottom-up features and computes gaze shifts with respect to them [32]. Later, Kim et al. [33] propose a model that uses both bottom-up and top-down attention mechanisms. They use features of objects such as position, orientation, and velocity to generate attention scores for the objects in the visual field of the virtual agent. As a different approach, Itti et al. [34] and Sullivan et al. [35] use saliency maps to reach the same goal.

In a recent study, Oyekoya et al. [8] use four intrinsic saliency attributes, proximity, eccentricity, orientation and velocity, to determine interests of characters in scene objects. They compute and combine these four components in a summation function to obtain a final saliency score, which is used to direct attention. Simply, the object with the highest score grabs attention. They compare their work with random gaze model and gaze tracker data and observe that their method generates results that are highly correlated to gaze-tracking outputs. However, this model is only applicable for predicting the gaze behavior of users and avatars they control, not for crowd simulations. It does not generate a realistic gaze behavior for a virtual character in a virtual crowd, since the user of the gaze tracker is aware of that s/he is controlling an avatar in a simulation and s/he may not act like a person in real-life. Also, being in a crowd or belonging to a group has some psychological effects that can alter behaviors of characters, which is missing in this model. Even though simulating crowds is not the goal of this study, it has a considerable impact on our work, especially with its saliency scoring method.

Another study, and may be the one mostly inspires us in motivation and development stages, belongs to Grillon and Thalmann [7]. They simulate gaze behavior in crowd characters. Their approach runs as an extra layer to an existing crowd simulation where the interest points of characters can be directed to static objects or other characters in the scene. For static objects in the scene, they assign meta-information that defines interestingness, and depending upon this property, the objects attract attention. Along with it, they assign different levels of interest to characters so that some of them attract more attention than others. The field of view feature is also implemented, so only the entities that stay in this field can be viewed by the agent. Although we use this study as a baseline, we include additional features: (i) we assign personality traits to characters and (ii) implement adjustable crowd norms. In this way, we can easily alter the criteria for gaze shifts and obtain heterogeneous gaze behaviors.

2.4 Human Visual System and Gaze Behavior

We briefly analyze some of the related studies about human visual system and perception in human psychology. Ensemble coding concept suggests that brain uses summary information to perceive the viewed scene. According to psychological studies on perception [1, 2], humans perceive a group of people as a whole single entity by inspecting the abstract features of group’s appearance, which grants a faster and more precise inspection than inspecting each character one by one. With ensemble encoding, to obtain perception of a group, detailed information of individuals is sacrificed [36, 37]. Moreover, ensemble encoding-getting visual information of a group by averaging the features of members- provides surprisingly high sensitivity [38].

2.5 Discussion

We examine related works in four aspects: *saliency computation methods*, *behavioral crowd simulations*, *gaze point detection in virtual environments*, and *human visual system*.

Both of the approaches proposed by Grillon and Thallmann [7] and Oyekoya et al. [8] do not include personalities to their computational models, which can cause significant differences on the behavior of characters such that a shy person tries to avoid looking at other pedestrians while a not-shy person moves his/her gaze through other characters more frequently. We admit that there are discussions on how shyness is related to personality traits; a shy person is not necessarily introvert. It is either related to low extraversion [39] or neuroticism [40]. We choose shyness instead of introversion or neuroticism and apply it to change gaze behavior. When the proposed framework is used with a crowd model with personality traits, the shyness parameter can be associated with extraversion or neuroticism levels of the character. Next, when simulating the behavior of crowds, including societal effects has a positive impact for more realistic results [24]. As

an example, crowd norms can alter the decisions of characters, thus different norms cause different gaze behavior.

In our work, although we simulate gaze behavior of crowd, the proposed model is inspired by both non-perceptual groups and perceptual groups approaches [24], therefore it can be considered a hybrid model. In real life examples, it is uncommon to observe crowd characteristics where all the agents walk as a single entity in the environment [41, 42, 43]. Observations on crowds suggest that up to 70% of people in the street prefer to walk in groups [44]. Hence, we assume that groups in the crowd are preexisted, i.e., they are friends or family members, and attracted to move together unless the group size is 1. In addition to this, social norms of the scene, which are pre-adjusted, have an impact on interest point decisions of characters.

Chapter 3

Simulating Interest Point Detection and Gaze Behavior

In this chapter, we explain the details of the proposed approach. The motivation of our proposed approach is to increase the realism of crowd simulations by implementing a gaze behavior model.

From the industrial perspective, even the latest video-games do not use a proper gaze behavior model for the virtual characters and the movie industry needs observations of artists for this purpose. When we look from the academic perspective, there is still not much work on this field of computer graphics [8, 7]. The existing approaches are either not applicable to crowds or omit social aspects of crowds. We believe that creating a framework and an example pipeline for gaze behavior of virtual characters will have a great impact on behavioral crowd realism and increase the interest in this area.

The proposed approach, which runs as an extra layer on Unity game-engine, aims to serve this purpose. It consists of two main steps: *interest point detection* and *applying gaze shifts to characters*. In Step 1, the process is completed in three stages. First, the user needs to adjust the parameters of entities in the scene. Second, we compute the properties that will be used in saliency scoring.

In the third stage, we compute the saliency scores and predict interest points for each character. Step 2 has two stages. The first stage calculates the gaze states of characters using the predicted interest points and the second stage generates head animations depending on the gaze duration factor. We provide the details of these two steps and describe the algorithm for computing saliency score in the sequel.

3.1 Interest Point Detection

This part of the thesis deals with the properties that are used in saliency computation and real time prediction of interest points. Before going into details, we need to clarify the crowd organization and group behavior that we used.

Group organization of crowd: Observations on crowds suggest that up to 70% of people in the street prefer to walk in groups [44]. Based on this, our simulated crowds consist of individual pedestrian agents each of which belongs to a group. Since our target is simulating the crowd in street on a normal day, we use the word *group* for characters that have sociological relations between them, i.e., they know each other and do not stay together incidentally. The group size ranges from one to five in the current implementation of the framework. Even when an agent is walking alone, we consider it as a member of a group, whose member count is one.

Perception of group by virtual characters: Along with the small groups approach for crowd organization, we adopt the ensemble encoding concept [1, 2]. This concept claims that humans perceive a group of people as a whole single entity by inspecting the abstract features of group’s appearance. By implementing this feature, virtual characters perceive only groups, not the individuals who belong to them. Moreover, properties of groups are determined by averaging the values of members’ properties.

Determining field of view for characters: Because the individuals in the same

group stay together willingly, we assume that characters in the same group have the same speed and direction vectors; hence their view angles, view directions and distances to other objects are similar. With this assumption, viewed entities in the environment will be same for every member of a group. However, each character determines its own interest point independently because personalities of characters can be different from each other.

In addition to realism concerns, “small groups” approach brings computational performance improvement to the framework. The number of entities to look at decreases from number of individual characters to number of groups. In our framework, we know that each group contains at least one character in it, so we can assume that the number of comparisons for an individual character to detect interest point will decrease in general.

3.1.1 Attributes of Characters and Objects

To determine interest points, we need to assign certain attributes to entities in the scene so that we can make comparisons to find the most salient point. These assigned attributes are displayed in a parametric form and need to be adjusted by the user before running the program. Also, this process facilitates the generation of heterogeneous crowd behavior (e.g., some characters walk faster than others). The attributes assigned to characters and non-human objects differ by their definition or usage. However, as a common trait they all range between 0 and 1.

Characters in the model have both physical and psychological attributes. Physical attributes, which are height, speed and attractiveness, result in behavioral and physical variety in the crowd. Moreover, their other purpose is to allow the model to compute relative saliency parameters such as proximity and velocity. On the other hand, psychological attributes; shyness, curiosity have a direct effect on characters’ gaze decisions. We use them as threshold values for characters to determine whether it should look at a salient point or not. Adjustable attributes of non-human objects are speed, attractiveness and type. From these

parameters, we compute a single parameter; distinctiveness, and use it in saliency comparisons.

3.1.2 Saliency Score Computation Parameters

Because characters and non-human objects have different properties and roles in the scene, we compute their saliency scores in distinct processes. In these processes, we compute individual scores that are necessary for computing the total score that represents saliency of an entity.

3.1.2.1 Agent-Group Saliency Parameters

Characters perceive groups as single entities; hence we use the term agent-group instead of agent-agent. The parameters used are proximity, periphery, velocity, orientation and attractiveness. The studies of Grillon et al. [7] and Oyekoya et al. [8] and our observations on real-world crowd video examples motivated our choice of these five criteria as parameters to the scoring function.

Proximity: For an agent, as the distance to an object or another agent decreases, the tendency for the agent to look at them increases. Due to the functionality of human visual system, closer objects seem larger than the ones that are far away. Also, objects behind the closer one are occluded and are hard to see.

Periphery: Human visual system is more active for the motions occurring in peripheral vision area than the ones occurring out of this area. The attention is maximum when the object is entering periphery.

Orientation: An agent will be more attentive to objects that come through to it because in subsequent time steps the object will be closer than the ones that are far away.

Velocity: Difference in velocity values grabs attention of an agent. Generally speaking, objects with high velocity are salient, however, if the speed of the crowd norm is high then a slowly moving object may be salient.

Attractiveness: Different than the above four criteria, attractiveness is a top-down saliency feature. As attractiveness of a character increases, the chances that it grabs attention gets higher. Again, the crowd norm is an important factor in the resulting score.

Height: We choose this criteria because height is one of the most obvious physical properties that we can detect in a crowd. We consider an agent with a height value different than the norms of the crowd as salient.

3.1.2.2 Agent-Object Saliency Parameters

We use the parameters *proximity*, *orientation*, *velocity* and *distinctiveness* in agent-object saliency computation. As being the only parameter different than the agent-group saliency parameters, *distinctiveness* reflects the center-surround mechanism used in saliency map computations. As the object becomes different than the neighboring objects, it grabs more attention.

3.1.3 Saliency Scoring

This module uses character groups and non-human objects in the scene as input and returns the most salient entity in the scene. The steps of this process is as follows: determining entities that stay in the agent’s field of view, computing individual saliency scores (e.g., proximity, orientation), and computing the total saliency score. The pseudo-code for computing the total saliency scores is given in Algorithm 1.

Algorithm 1 Saliency Model Interest Point Computation

Require: $viewedGroups \leftarrow$ groups agent can see

Require: $viewedObjects \leftarrow$ objects agent can see

```
1: for each frame do
2:    $p_a \leftarrow$  compute agent's position in world coordinates
3:    $d_a \leftarrow$  compute agent's direction in world coordinates
4:    $v_a \leftarrow$  compute agent's velocity
5:    $\Delta TGaze \leftarrow$  elapsed time since the beginning of current gaze motion
6:   if  $\Delta TGaze \geq minDur$  then
7:     for each group in  $viewedGroups$  do
8:        $S_p \leftarrow$  compute proximity score ▷ Eqn. 3.1
9:        $S_o \leftarrow$  compute orientation score ▷ Eqn. 3.3
10:       $S_{per} \leftarrow$  compute periphery score ▷ Eqn. 3.4
11:       $S_v \leftarrow$  compute velocity score ▷ Eqn. 3.2
12:       $S_a \leftarrow$  compute attractiveness score ▷ Eqn. 3.5
13:       $S_h \leftarrow$  compute height score ▷ Eqn. 3.6
14:       $S \leftarrow$  compute total score ▷ Eqn. 3.7
15:    end for
16:    for each object in  $viewedObjects$  do
17:       $S_p \leftarrow$  compute proximity score ▷ Eqn. 3.1
18:       $S_o \leftarrow$  compute orientation score ▷ Eqn. 3.8
19:       $S_v \leftarrow$  compute velocity score ▷ Eqn. 3.2
20:       $S_d \leftarrow$  compute distinctiveness score ▷ Eqn. 3.10
21:       $S \leftarrow$  compute total score ▷ Eqn. 3.11
22:    end for
23:     $S_{max} \leftarrow$  highest saliency score
24:     $DecideGazeState(S_{max})$  ▷ Algorithm 3
25:    if can copy gaze then
26:       $GazeCopy()$  ▷ Algorithm 4
27:    end if
28:     $p_t \leftarrow$  compute gaze target position
29:     $ApplyGazeAnimation(p_t)$ 
30:    Update attention coefficients of entities ▷ Algorithm 5
31:  end if
32: end for
```

d_m	Maximum distance that an agent can see
d_{ag}	Euclidean distance between agent a and group g
S_{max}	Maximum pedestrian speed in crowd
S_{min}	Minimum pedestrian speed in crowd
S_n	Speed value of crowd norm
p_a	3D position of agent a
p_g	Average 3D position of group g
p_e	3D position of entity e (can be group or object)
p_t	Target gaze position
fd_a	Forward direction of agent a
fd_g	Average forward direction of group g
α	the angle between fd_a and the vector $p_g - p_a$
β	the angle between fd_a and fd_g
C_s	coefficient to simulate impact of speed on gaze behavior

Table 3.1: Parameters for saliency scoring.

3.1.3.1 Determining Viewed Groups and Objects

Because of the nature of human visual system, only the entities that are located inside the field of view can visually attract the viewer. We have two lists for each character group, ‘*viewedGroups*’ and ‘*viewedObjects*’, that store groups and objects, respectively, in its field of view at time stamp t . The viewed groups and objects are determined by doing three culling processes, which are *proximity*, *orientation* and *occlusion* culling. Because comparing all of them in scoring process would cause unnecessary computations, they are eliminated beforehand. The following culling processes are applied as described in Algorithm 2:

Proximity culling: For a group, only the groups that stays in a distance are accepted and others are eliminated. We determine this distance by ‘*maxViewDistance*’ parameter, which we set as 15 meters in our framework, meaning that agents cannot direct their gaze to other agents which are farther than 15 meters.

Orientation culling: Here, entities that passed proximity elimination are taken as input. We eliminate the ones that stay out of the average visual angle of a group. We choose the angle as 80° .

Occlusion culling: For the remaining viewed entities, we perform visibility tests for each agent to find out if these entities are occluded by another entity at the current time-stamp. We do this by ray casting from agents to the viewed entities. If the hit distance is shorter than the distance between agent and the tested entity then, it is occluded by another object, which means the agent cannot see the target entity. For this reason, we remove that group from the viewed groups list of the agent.

By doing these culling processes, we define a realistic field of view area for agents and improve computational performance with reduced number of comparisons. Even though there are more than one hundred groups in the scene, an agent usually compare four or five groups at maximum in a single step.

Algorithm 2 Culling

```

1: function CullEntities(allEntitiesList)
2:   viewedEntities  $\leftarrow$  allEntitiesList
3:   for all entity E in viewedEntities do
4:     if distance( $p_g, p_e$ )  $\geq$  maxViewDistance then
5:       viewedEntities.remove(e)
6:     else if Angle(fdg, ( $p_e - p_g$ ))  $\geq$  80 then
7:       viewedEntities.remove(e)
8:     else if RayCast( $p_g, p_e$ ).hit  $\neq$  e then
9:       viewedEntities.remove(e)
10:    end if
11:  end for
12: end function

```

3.1.3.2 Agent-Group Saliency Scoring

In this step, we compute the total saliency scores of the viewed groups of agent *a* and choose the most salient one as ‘*salientGroup*’. To obtain the total saliency score, we first need to compute the individual scores of all saliency criteria, which are proximity, orientation, periphery, velocity, attractiveness and height.

1. Given the agent’s position p_a , and the group’s position p_g , we compute the proximity score, S_p , as

$$S_p(t) = \frac{d_m - d_{ag}}{d_m}, \quad (3.1)$$

where d_{ag} is Euclidean distance between two position vectors p_a and p_g , and d_m represents the maximum distance that agent a will stop looking after. Subtracting d_{ag} from d_m ensures that the closer agents to have higher proximity scores. We divide the score by d_m to normalize the value to the range $[0, 1]$.

2. The velocity score, S_v , is based on the relative velocity of the group's walking speed with respect to the crowd norm's speed, and we compute it as

$$S_v(t) = \frac{\|\mathbf{v}_g - \mathbf{v}_a\| - s_n}{\max((S_{max} - s_n), (s_n - S_{min}))}. \quad (3.2)$$

The difference between the velocity of the viewed group (v_g) and the velocity of the agent (v_a) gives the relative velocity. We compute the velocity score by subtracting the speed of the crowd norm from the magnitude of the relative velocity. We normalize the velocity score by dividing it to the maximum of $(S_{max} - s_n)$ and $(s_n - S_{min})$ (cf. Table 3.1).

3. The orientation score, S_o , is the relative direction of the group with respect to the agent, and we compute it as

$$S_o(t) = \left(\frac{\pi}{2} - \alpha\right) \beta. \quad (3.3)$$

For an agent, groups that have opposite directions will produce higher scores than those having similar directions. As the angle α increases, the orientation score decreases, since the target is at the edge of periphery. To normalize $S_o(t)$ to the range $[0, 1]$, we divide the score by $(\pi \times \frac{\pi}{2})$ and take its square root. π and $\frac{\pi}{2}$ are the maximum values β and α can take because the orientation culling process eliminates groups that are located behind the agent (cf. Table 3.1).

4. We compute the periphery score, S_{pe} , for evaluating groups that enter peripheral view cone as

$$S_{pe}(t) = \left(\frac{\pi}{2} - \beta\right) \alpha. \quad (3.4)$$

The minimum value of β together with the maximum value of α reflect the case where group g enters the periphery of agent a . The normalization of $S_{pe}(t)$ is the same as the normalization of S_o .

5. We compute the attractiveness score, S_a , as

$$S_a(t) = \begin{cases} \frac{a_g - a_n}{1 - a_n}, & \text{if } a_g \geq a_n \\ 0, & \text{otherwise} \end{cases} \quad (3.5)$$

where a_g is the average attractiveness of group g and a_n is the attractiveness value of the crowd norm. S_a represents the relative attractiveness of the target group with respect to the crowd norm. The attractiveness property is in the range of $[0,1]$.

6. We compute the height score, S_h , as

$$S_h(t) = \frac{h_g - h_n}{4\sigma}, \quad (3.6)$$

where h_g is the average height of the group, h_n is the height of the crowd norm, and σ is the standard deviation of the height in the crowd. We assume that the height value is normally distributed. S_h is z-score of h_g divided by 4σ , which indicates how much h_g is different than h_n [45]. The resulting value is in the range $[0,1]$.

Total Saliency Score: To be able to compare different entities, we combine the individual scores as a single scalar value for each entity. The weighted sum function takes the individual saliency scores and generates a total saliency score, which is in the range $[0, 1]$:

$$S(t) = C_a(W_p S_p + W_v S_v + W_o S_o + W_{pe} S_{pe} + W_a S_a + W_h S_h), \quad (3.7)$$

where $W_p, W_v, W_o, W_{pe}, W_a, W_h$ represent the pre-defined weights of six saliency criteria, which are proximity, velocity, orientation, periphery, attractiveness and height, respectively. The weights of each sub-score can be set by the user or set randomly as long as the sum of them is equal to 1. C_a is a coefficient that correspond to factors that affect the gaze behavior of agent a , which are speed and losing interest.

3.1.3.3 Agent-Object Saliency Scoring

Since non-human objects have different properties than agent groups, computing saliency scores of objects requires scoring of distinct criteria. However, the process is similar to agent-group saliency scoring. The criteria are proximity, velocity, orientation, and distinctiveness. The computations of proximity and velocity scores are the same as the ones for the agent-group saliency scoring process (cf. Eqns. 3.1, and 3.2, respectively).

1. The orientation score S_o is computed as:

$$S_o(t) = \frac{\pi}{2} - \alpha. \quad (3.8)$$

The reason for omitting β angle in Eqn. 3.8 is that the direction vector of an object is not meaningful when the object is static in the scene. Thus, the relative position of object o with respect to agent a determines the orientation score. We divide the score by the maximum α angle, $\frac{\pi}{2}$, to normalize it.

2. To compute the distinctiveness score S_d , we compute the unnormalized distinctiveness score, S' , first.

$$S' = \frac{\sum_{i=1}^n x_i}{x_j}, \quad (3.9)$$

where n is the number of different object types that exist in the decided radius of object o , x_i are the numbers of objects of each type and x_j is the

number of objects of object o 's type. This distinctiveness score is inversely proportional to the frequency of the object type, i.e., the less frequent an object, the higher score it gets. We normalize the distinctiveness score dividing this value to the sum of the scores:

$$S_d = \frac{S'}{\sum_{i=1}^n S'_i} \quad (3.10)$$

Total Saliency Score: We compute the total agent-object saliency score as

$$S(t) = C_a(W_p S_p + W_v S_v + W_o S_o + W_d S_d), \quad (3.11)$$

where W_p, W_v, W_o, W_d represent pre-defined weights of the four saliency criteria, which are proximity, velocity, orientation and distinctiveness, respectively. Similar to agent-group process, the sum of weights needs to be equal to one and C_a is a coefficient that correspond to factors that affect the gaze behavior of agent a .

3.1.3.4 Applying the Behavioral Effects of Agent's Properties

In this section, we explain the coefficients that are involved in C_a ($C_a = C_s \cdot C_i$). The first coefficient, C_s , represents the impact of walking speed on gaze behavior. We use the second one, C_i , to determine how fast an agent loses her/his interest on the salient entity.

According to the study [3] and our observations on videos of real crowds, the walking speed of an agent is inversely proportional with her/his motivation to look at. As an example, running pedestrians look at the environment much less than slowly walking pedestrians. We compute this effect as

$$C_s(t) = \begin{cases} 1 - 0.4 \sqrt{\frac{v_a}{v_m}}, & \text{if } v_a \geq 1 \\ 1, & \text{if } v_a = 1 \\ 1.25, & \text{otherwise} \end{cases} \quad (3.12)$$

where v_a is the velocity of agent a , v_m is the maximum velocity value defined in crowd norm. When the agent is walking with maximum velocity, his/her gaze decreases to 60%. Also, as an agent walks with $1m/s$, C_s is 0, so it has no effect on scores. When an agent stands (i.e., *velocity* = 0), C_s becomes 1.25, which increases her/his attention duration and frequency. Fontana et al. [3] suggest that the attention frequency and duration decreases with increasing walking speed. The minimum and maximum coefficient values are determined with the observations on real world crowd videos, which are used in evaluation. C_i is used as a coefficient to determine the total score by including the agent's interest. While the agent a is actively looking at an entity e , we decrease the score of e relative to a as the agent loses her/his interest (cf. Algorithm 5).

3.2 Adjusting Gaze

We repeat saliency score computation process for all groups and objects viewed by the agent. After computing the total saliency scores, we determine the entity with the highest combined saliency score as the interest point. The entity can be a group or a non-human object. When there exists only one entity e in the visual area of agent a , entity e gets the maximum saliency score. However, if e is not salient enough, agent a should not direct her/his gaze through e . To determine if an agent would look at an entity, we use two personality traits *shyness* and *curiosity* for the agent. The user needs to specify these values for each agent within the range $[0, 1]$. Shyness is simply the threshold value for an agent to look at other agents. An agent with high shyness value (introvert) looks at other groups much less than the agent with low shyness value (extrovert). The use of the curiosity parameter is similar; it determines whether an agent finds the

salient object interesting or not (cf. Algorithm 3):

Algorithm 3 Decide Gaze State and Target Position

```

1: function DecideGaze( $S_{max}$ )
2:    $p_e \leftarrow$  position of  $S_{max}$  entity
3:   if  $S_{max} \geq$  threshold then
4:      $p_t \leftarrow p_e$ 
5:   else
6:      $p_t \leftarrow$  ComputeDefaultGaze()
7:   end if
8: end function

```

We execute gaze shifting with respect to the decided gaze state. The decision mechanism have four different states:

State 0: Look at front: In this state, the agent looks forward. We compute the target position as the summation of two vectors:

$$p_t = p_a + f d_a.$$

State 1: Look through direction of group: In this state, the agent looks in front of the group's center of mass.

$$p_t = p_g + f d_g.$$

State 2: Look at a salient group: The target position is the average position of the salient agent group.

State 3: Look at a salient object: The target position is the position of the salient object.

We consider *State 0* and *State 1* as default (idle) gaze behavior because the agents are not interested in the environment. When we decide the default gaze behavior, we select one of these two states randomly with the probabilities of *State 0* and *State 1* are 0.7 and 0.3, respectively. *State 2* and *State 3* are active gaze behaviors, in which some entities in the environment grab attention. After

deciding the gaze state, we apply gaze animation to the agent. Table 3.2 describes the parameters for gaze adjustment.

C_i	coefficient to simulate the agent to lose her/his interest
$minDur$	the minimum duration that an agent has to look after starting a gaze process
$maxDur$	the maximum duration that an agent can hold its gaze on the same target
$\Delta Time$	the duration between current frame and the previous frame
S_e	Salient entity that the agent will direct her/his gaze through
p_{at}	the vector from p_a to p_t
$rotAngle$	the angle to rotate head of agent in y axis

Table 3.2: Parameters for gaze adjustment.

3.2.1 Gaze Copy

When deciding the gaze of agents, another important process, *gaze copying*, takes place. Gaze copying is result of the group behavior of crowds. When an agent is staring at an object in the scene, the other agents that see her/him have an intention to look at that object. According to Gallup et al. [46], 26.9% of passersby adopts the gaze direction of the pedestrians who are already staring. We include this feature in the model as intra-group gaze copying mechanism (cf Algorithm 4). The members of the same group are affected by each others' staring behavior. We determine this case randomly with a probability of 0.3.

However, in the case where multiple agents are staring at multiple entities, the decision gets complicated. To solve this problem, we choose the entity that is stared by more agents as the *staredOne* entity. For example, in a group of four pedestrians, two of them are staring at object *o1*, one of them is staring at object *o2*. Then remaining agent, if s/he will copy the gaze, starts looking at *o1*. Besides, in the case where the staring member counts are equal, we choose the target entity randomly.

Algorithm 4 Gaze Copy

Require: *agent.group.staredOnesList* $\leftarrow$ *list of entities members are staring*

- 1: **for** *each agent a in group g* **do**
- 2: **if** *IsStaring(a)* **then** $\triangleright$ looking at the same entity more than 1 sec
- 3: *g.StareList.add(a)*
- 4: **end if**
- 5: **end for**
- 6: $\triangleright$ Gaze Decision Part
- 7: **for** *each agent a in group g* **do**
- 8: **if** *g.StareList.Length > 0 and random() < 0.3* **then**
- 9: *staredOne gets most stared entity*
- 10: *a.S_e $\leftarrow$ staredOne*
- 11: **end if**
- 12: **end for**

3.2.2 Gaze Shifting Animation

The process is simply computing the required rotation angle and rotating the head object until it faces the target position.

$$rotAngle = \angle(f\vec{d}_a, \vec{p}_{at}),$$

where $f\vec{d}_a$ is the forward direction of agent a , $\vec{p}_{at}$ is the vector from p_t (target position) and p_a is the position of agent a). After this step, agent a starts to look at tp and gaze shifting is completed. Because we compute saliency scores and perform gaze shifting comparisons at every frame, characters automatically shift their gazes without delay.

3.2.3 Gaze Duration

After an agent starts to look at another agent, if it does not stop looking after some time then the model would generate unrealistic behavior. On the contrary, if an agent changes its gaze frequently, e.g., in every 3-4 frames, again this would generate unlikely motions. We specify the minimum and maximum gaze durations

to solve this problem.

Minimum gaze duration ($minDur$): This is the amount of time that agent a must hold its gaze before shifting it to another object. We set the value of this parameter to 0.5 sec to generate realistic gaze shift behaviors.

Maximum gaze duration ($maxDur$): This is the amount of time that agent a can hold its gaze before it loses its interest for the current salient entity. We set the parameter to approximately six seconds. However, losing interest does not occur instantly, instead characters lose their interest slowly in time. So, $maxDur$ parameter does not work as $minDur$, which changes gaze state sharply. We use another parameter, C_i , to simulate losing interest, which decreases at every time step, to be multiplied with the saliency scores (cf. Eqns. 3.7 and 3.11).

Algorithm 5 Adjust Interest Coefficients

```

1: function UpdateInterestCoeffs()
2:   for all  $score$  in  $agent.scoresList$  do
3:     if  $score$  belongs to salient agent then
4:        $score.C_i \leftarrow score.C_i - \Delta Time \times 0.15$ 
5:     else
6:        $score.C_i \leftarrow score.C_i + \Delta Time \times 0.01$ 
7:     end if
8:   end for
9: end function

```

Algorithm 5 depicts the adjustment of interest coefficients for determining the gaze duration. When agent a looks at an entity e , C_i of e with respect to a decreases while those of other entities increase. After some time, the score of e will be lower than the thresholds of a (shyness/curiosity) or another entity's score, which causes a to automatically shift its gaze.

Chapter 4

Implementation of the Framework

This chapter describes the implementation of the proposed model and the construction of virtual scenes that are used in evaluation. Graphics models, gaze animation, navigation of characters and architecture of the framework are explained in detail.

4.1 Graphics Models

We use character and environment models to generate virtual scenes. We use environment models to represent the market area of Middle East Technical University (METU) and design character models that are similar to the characters in videos that we recorded. We model characters used in simulations by using Adobe Fuse toolkit [47], which enables non-artists to model human characters easily. After modeling the characters, we rig 3D models of characters and animate them with another application Mixamo [48], which can be connected to Fuse. We choose these tools due to their simplicity and compatibility with the Unity game engine. To construct a scene in which virtual crowd moves and gaze

behavior of characters can be displayed, we prepare virtual scenarios in Unity Game Engine [4] with respect to the videos that we shot in METU campus market area. Because the crowd and the area are highly suitable to be used in a gaze behavior simulation, we choose this area as the test scene. Besides, the crowd in METU campus can be considered as a closed group, compatible with the requirement for “crowd norm” property used in the proposed model.

4.2 Animation

We prepare animations using Unity and Mixamo animation packages, which store motion capture animations. We place the skeletons with Mixamo tool to make them ready for applying animations. Because the main goal of the proposed model is to predict interest points, we animate gaze by just rotating the character’s head. We compute the rotation angle and apply the rotation to the head of the character. To simulate steering behavior and path finding, we use a free unity package Locomotion System and Unity’s navigation mesh agent component. The locomotion system blends the motion-captured walking and running cycles and adjusts the skeleton to provide a plausible moving animation.

To achieve path finding, we assign NavMesh agent component to virtual characters. With the help of this component, agents can avoid each other and other moving or static obstacles while walking towards their target. As a required preprocessing for this component, we bake the scene for navigation meshes, i.e., walkable/non-walkable areas of the scene are determined and mapped.

In test scenario scenes, the number of destination points ranges from one to five. The user specify the speed and destination points of the crowd. After specifying these parameters, agents determine their paths automatically through their target.

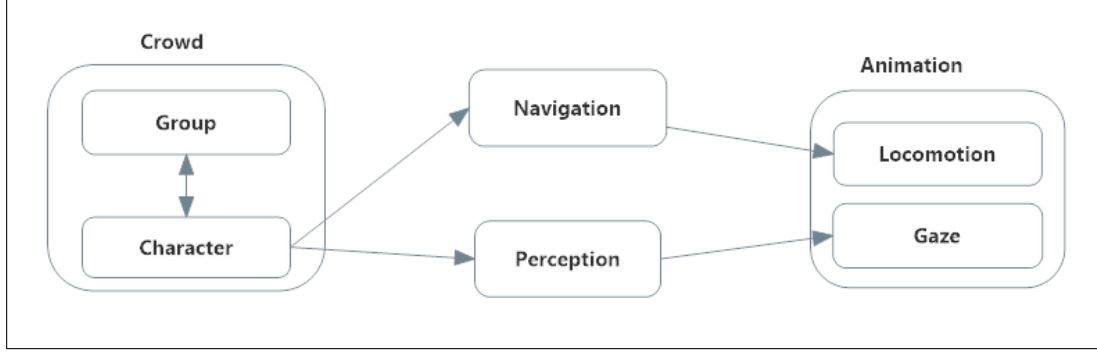


Figure 4.1: The relationships between agent components.

4.3 Architecture of the Simulation

We determine the behaviors of agents with pre-defined rules and scoring functions as a result of the interaction with the environment and other agents. Figure 4.1 shows the relationships between agent components. Figure 4.2 depicts the gaze behavior decision logic of an agent as a flowchart. In the diagram, ‘Start’ represents the idle state where the agent stands before starting to a new action or after finishing the previous action. We first check the group component that the agent belongs to find out whether there exist any entities in the field of view of the agent’s group. If there exist no entities, the agent’s gaze enters the default gaze state, in which s/he looks in front or through the forward direction of the belonging group. However, if there are scene entities in the field of view area of the agent, then we compute the saliency scores of these entities. We compare the maximum score with the threshold values, shyness for groups and curiosity for objects. If the score passes the threshold, then the agent starts to look at the salient entity. Otherwise, s/he enters the default gaze state. For both cases where the agent looks forward or looks at the salient entity during the minimum gaze duration, agent preserve her/his state. However, when the duration ends, the end state is reached.

We implement the framework on top of the Unity Game Engine and use C# as the programming language. We code the program flow by following Unity’s scripting procedures. We mainly implement seven classes, in which four of them are controller scripts:

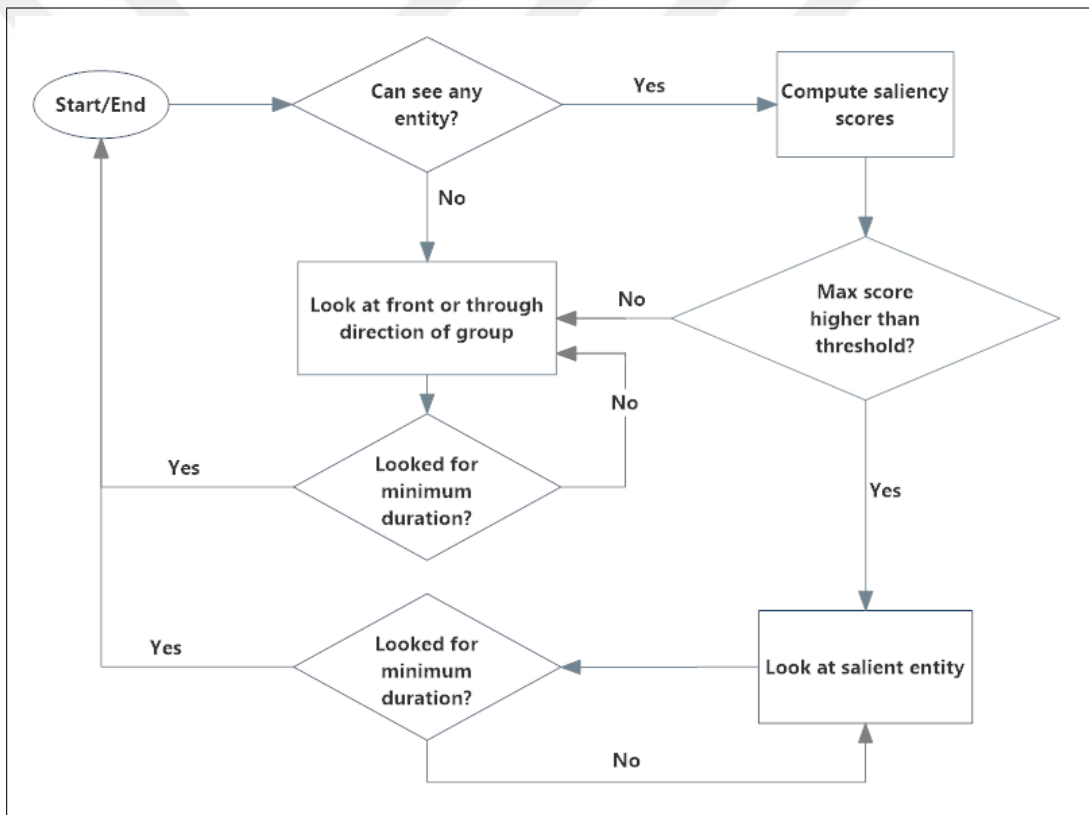


Figure 4.2: Gaze decision logic.

GroupController script is attached to group objects, which are abstract objects that hold agents as children. It updates the group properties by averaging the properties of child agents. We store group properties as a field of *GroupProperties* class. Also, we compute the field of view of the group and determine *viewedGroups* and *viewedObjects* in this script.

CharacterController is the main script that computes interest points and decides gaze states. After getting the list of viewed entities, it computes saliency scores and finds the most salient entity for the agent. It decides where the agent will look at, or which agent's gaze will be copied. Along with these, we use Unity's Time class in this script to determine the gaze duration. Each virtual agent has its own instance of this script.

ObjectController script is used for scoring non-human objects. It holds *ObjectProperties* instance as a field and keeps these properties updated.

GazeController script is used for rotating agent's head with the given target position input. *CharacterController* script calls gaze controller methods. Each agent has its own instance of this script.

GroupProperties class is inherited from *CharacterProperties* class, so it stores the same parameters. However, we update its parameters with the average of the group members' properties at every frame.

CharacterProperties class is instanced and used as a field in *CharacterController*. Its main functionality is to hold agent's properties, which are updated by *CharacterController* at every frame.

ObjectProperties class stores non-human objects' properties and computes the distinctiveness of the object with respect to neighboring objects.

Chapter 5

Evaluation and Results

Chapter 5 explains the evaluation of the model and the framework. To manage this, we record videos of people and analyze their gaze behavior. Along with the videos, we construct a virtual scene using Unity 3D game engine and simulate test scenarios in this virtual scene. We execute the navigation process of virtual agents with the nav-mesh agent component of Unity and generate animations with the Unity locomotion package.

5.1 Scenario Simulations

To verify the interest point detection and gaze behavior of virtual crowds, we perform a series of simulations for pre-determined scenarios. The purpose of these simulations is to validate the proposed gaze behaviors by comparing them with the gaze behavior of people appearing in the real videos.

In the proposed model, mainly ten different features have impact on the gaze behavior of agents. We test each of these features in different scenarios and compare with the corresponding videos to demonstrate their effects on attention. We record the videos used in the experiments in the same place in two different days. We record Scenarios 1-7 in Day-1 and Scenarios 8-9 in Day-2. In all

simulations, crowd norm settings are fixed. In this way, we demonstrate that the crowd norm feature is appropriate to use with the crowds located in similar regions.

5.1.1 Scenario 1

We use this scenario to examine the effects of proximity and periphery features. In the video, walking female agent grabs attention for two reasons: she got closer to the male agents and she enters peripheral view of them. Our implementation simulates this scenario with similar gaze behavior (cf. Figure 5.1). Male agents look at the female agent when she got closer and entered their periphery. To test the effects of these parameters, we set the weights of proximity and periphery parameters to zero in the second run of the simulation. As expected, none of the agents direct their gaze towards the female agent with this change.

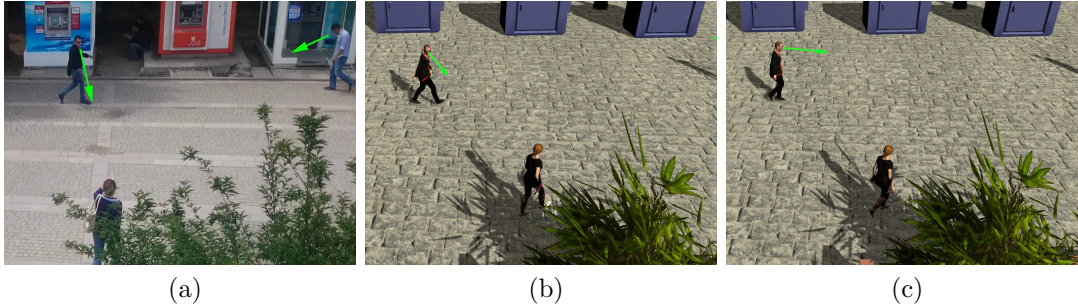


Figure 5.1: Effects of proximity and periphery: (a) video, (b) simulation, and (c) simulation (weight=0).

5.1.2 Scenario 2

We use this scenario to examine the effects of agent velocity. In the video, a female agent with high velocity runs from left side to right side of the scene. Within all the agents, the running female agent is the most salient entity. Most of the other agents look at her while she runs without performing any gaze shifts. With the implementation of the proposed model, we obtain gaze behavior similar

to the real video. In the simulation, three of the pedestrian agents look at the running female agent while she is running. Because velocity decreases agent's gaze intention, the female agent looks only at her front. To demonstrate the effect of velocity, we run the simulation with *velocityWeight* parameter set to zero. In this case, the running female agent cannot grab attention from the crowd (cf. Figure 5.2).



Figure 5.2: Effects of agent velocity: (a) video, (b) simulation, and (c) simulation (weight=0).

5.1.3 Scenario 3

This scenario aims to show the effects of attractiveness parameter. In the third video, an attractive female agent walks through the left side of the scene, and a male agent directs his gaze towards her. In the virtual scene, we set the female agent's attractiveness parameter to 0.8, which is a quite high value. In this condition, due to attractiveness score of the female agent, she grabs attention. However, when we decrease the attractiveness parameter to 0.4, which is still above crowd norms for this scenario, the female agent cannot grab attention (cf. Figure 5.3).

5.1.4 Scenario 4

This scenario consists a relatively complicated scene, in which the effects of character height and distinctiveness of non-human object can be examined. In the



Figure 5.3: Effects of agent attractiveness: (a) video, (b) simulation, and (c) simulation (agent's attractiveness=0.4).

video, the dog is the most salient entity and the small female child is the second most salient entity. Two walking agents mostly look at the dog and when the walking male agent come closer to the female child, he directs his gaze towards her for a short period of time. The other agent group (the female agent and the child) also look at the dog. We simulate this scene by means of distinctiveness, height, attractiveness and curiosity parameters. The reason that make the dog salient is that both it is a distinct object (no other object of the type animal exists) and it has a high attractiveness value. We specify high curiosity values for all agents, which increases the duration of their gazes through the dog. Besides, the height of the female child, which is way lower than the crowd norm's height value, make the group that the female child agent belong salient. The average height of the group that contains the female child become low and this affects the group's saliency score (cf. Figure 5.4).

For testing the effects of these parameters, we execute three simulations. First, we set the weight of distinctiveness parameter to zero, and as a result, no agent looks at the dog through the simulation. Second, we set the curiosity parameter of agents to zero. Because none of the agents are curious enough, the dog cannot grab attention. Third, we set the height parameter of the female child agent to 1.65, which is the average crowd norm. This time, walking male agent does not direct his gaze towards the female child agent's group and he continues to look at the dog.

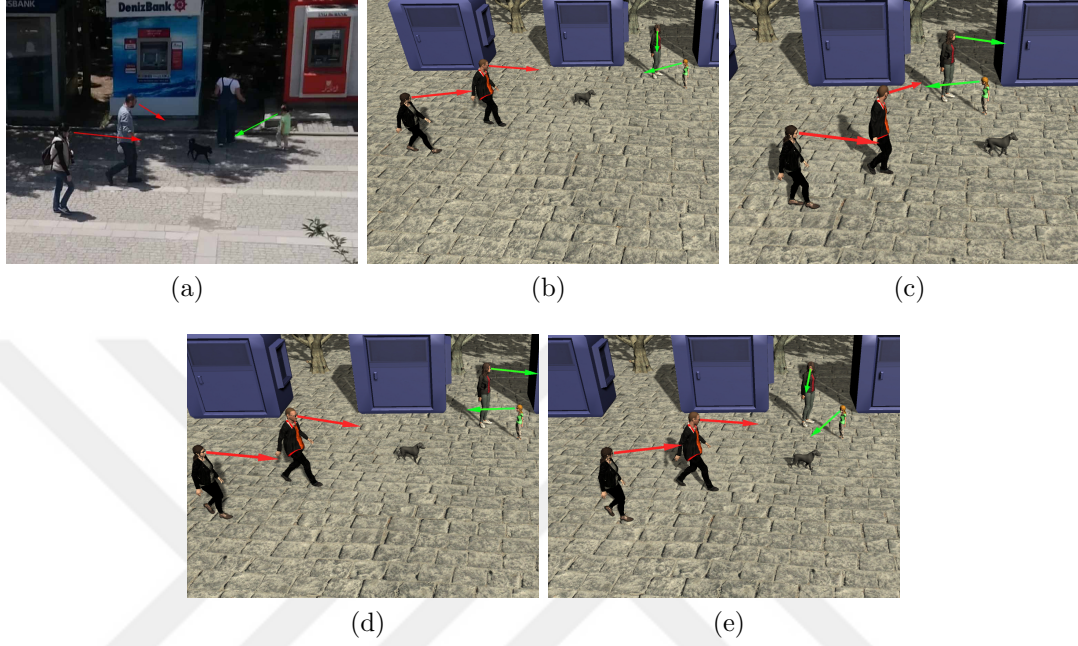
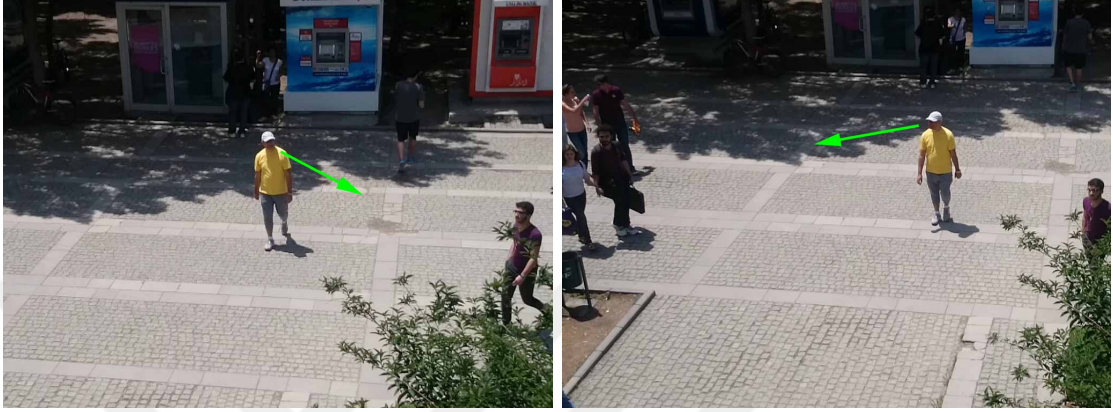


Figure 5.4: Effects of distinctiveness, curiosity and height: (a) video, (b) simulation, (c) distinctiveness = 0, (d) curiosity=0, and (e) child height = 1.65.

5.1.5 Scenario 5

In this scenario, we highlight the effects of the shyness parameter. The related video contains an extrovert male agent that looks at nearly all of the character groups around him while walking. In the simulation, we implement this condition by assigning low shyness value of 0.25 to the agent. The virtual agent acts similar to the real agent appearing in the video. To test the effect of the shyness parameter, we set it to 0.7. By doing so, the agent does not look at any other character during the simulation (cf. Figure 5.5).



(a)



(b)



(c)

Figure 5.5: Still frames showing the effect of agent shyness: (a) video frames, (b) shyness = 0.25, (c) shyness = 0.7.

5.1.6 Scenario 6

This scenario aims to indicate the impact of distinctiveness parameter of non-human objects. In the video, the bicycle is the most salient entity in the scene and so the passing-by agent directs his gaze towards the bicycle. We generate a similar gaze behavior in the simulation with the use of the distinctiveness parameter (cf. Figure 5.6). In the neighborhood of the bicycle object, it is the only object with type *vehicle*. Other than the bicycle, there are ATM machines and trees, whose types are *building* and *vegetable*, respectively. Being a rare type of object in the neighborhood results in a high distinctiveness score for the bicycle, which make it the most salient entity for the agent. In the second run, we set the weight of distinctiveness parameter to zero, and as a result, the agent does not look at the bicycle.



Figure 5.6: Effects of object distinctiveness: (a) video, (b) simulation, and (c) simulation (weight=0).

5.1.7 Scenario 7

Scenario 7 demonstrates the gaze copy phenomena between agents. In the video, we can observe the gaze copy behavior on the sitting couple. First, the male agent looks at one of the passing agents, then the female agent sitting next to him copies this behavior and starts to look at the same passing agents. In the simulation, we generate the gaze copying phenomenon by setting the male agent's shyness parameter to 0.3 and the female agent's shyness parameter to 0.7. This settings result in a situation where the male agent is more attentive than the

female agent. Because they are in the same group, the female agent copies the gaze behavior of her group member, as seen in Figure 5.7.

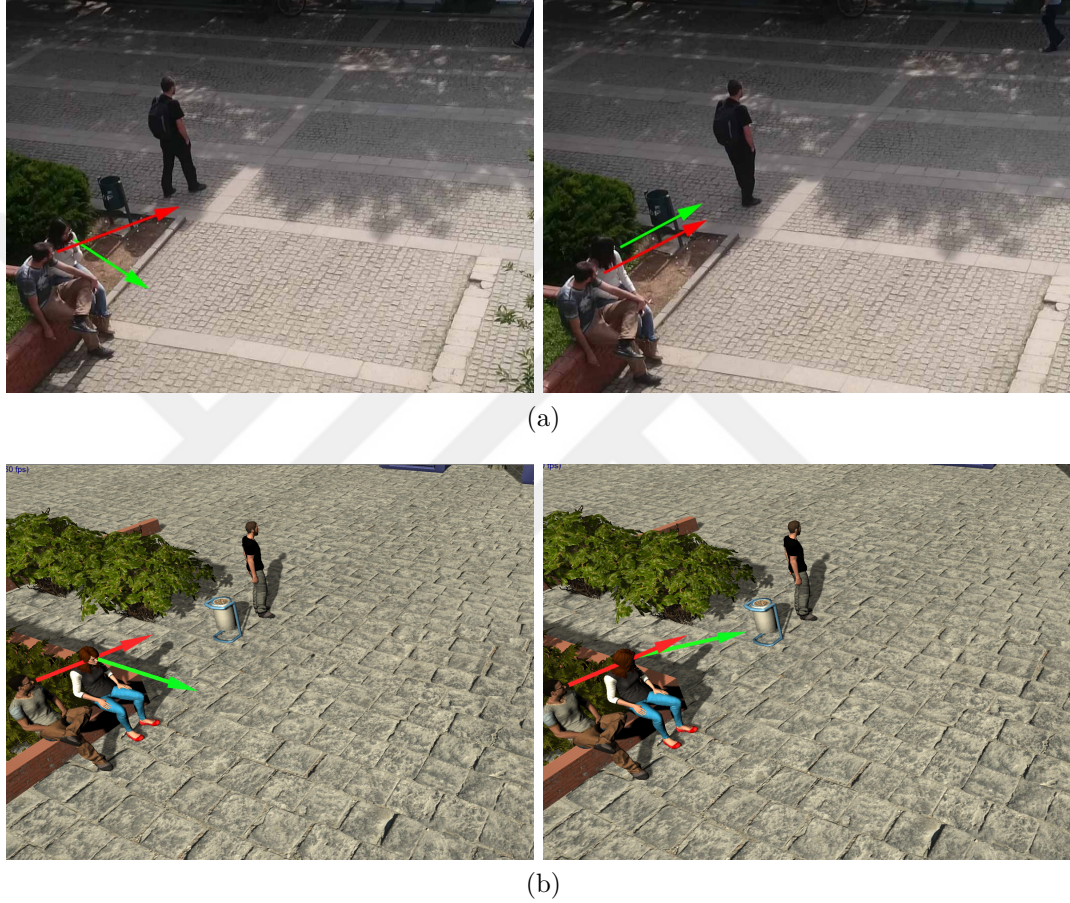


Figure 5.7: Still frames showing the effects of gaze copy: (a) video frames and (b) simulation frames.

5.1.8 Scenarios 8

This scenario shows the effect of the periphery feature. In Fig. 5.8, the first row shows frames from the real video and the second row shows frames from the simulation results.



Figure 5.8: Effects of periphery: first row: the real video, second row: the simulation result.

5.1.9 Scenarios 9

This scenario demonstrates the combined effect of proximity, periphery, attractiveness and height parameters. We also observe the effect of low speed by looking at sitting female agents; they change their gaze more frequently than the moving agents. In Fig. 5.9, the images on the left column belong to the captured video frames and the images on the right column belong to the simulation frames.

5.2 Performance

To test the performance of the framework, we run simulations containing 50, 100 and 200 agents on a PC with Intel Core(TM) i5-4590 3.30GHz CPU, 8GB RAM, and Nvidia GeForce GTX 970 graphics processor. Figure 5.11 shows a simulation with 100 agents where we disable gaze copy behavior scripts in part (a) and enable them in part (b).

Tables 5.1 and 5.2 depict the frame computation times and frame rates (frames

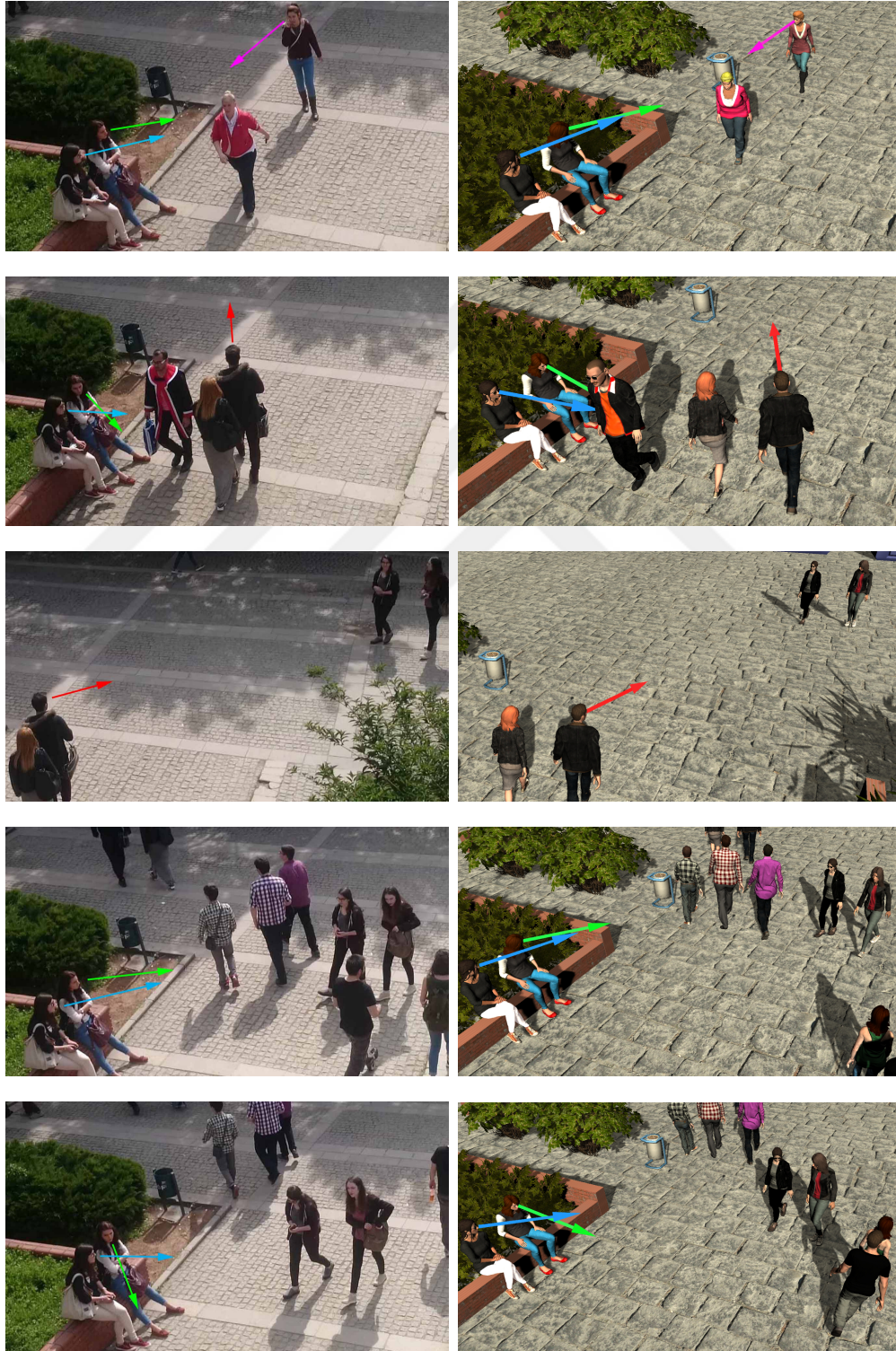


Figure 5.9: Still frames from a 40 second video: left column: the real video, right column: the simulation result.

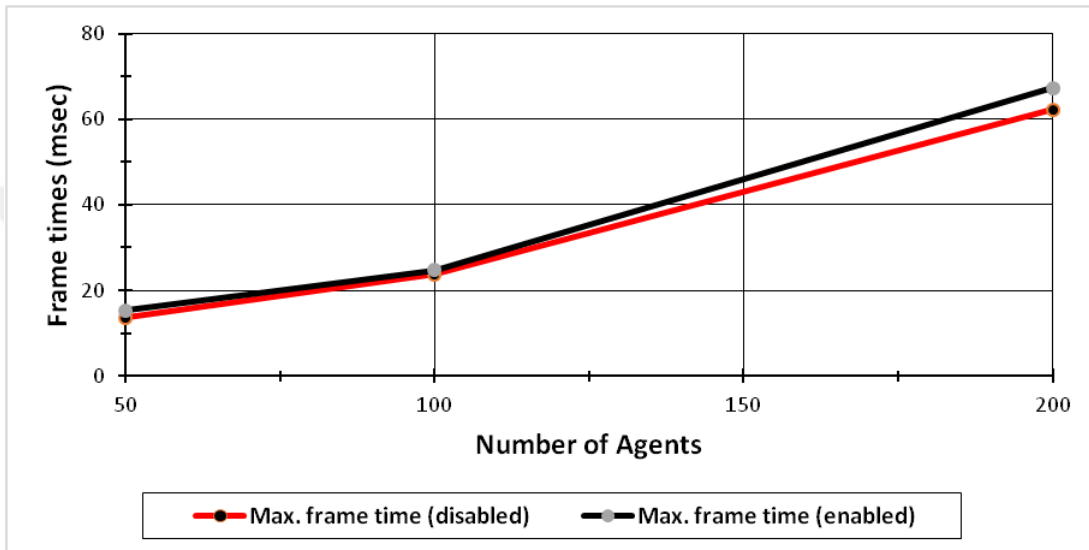
per second) with increasing number of agents in simulations where gaze behavior scripts are disabled and then enabled, respectively. Figure 5.10 shows charts for the comparison of (a) frame computation times and (b) frame rates (frames per second). These results demonstrate that the frame rates are considerably close. When the provided increase in realism is considered, the performance of the framework is promising.

Number of agents	Minimum frame time (msec)	Maximum frame time (msec)	Average frame rate (fps)
50	13.8	9.8	87
100	23.7	16.9	50.5
200	61.2	47.8	18.5

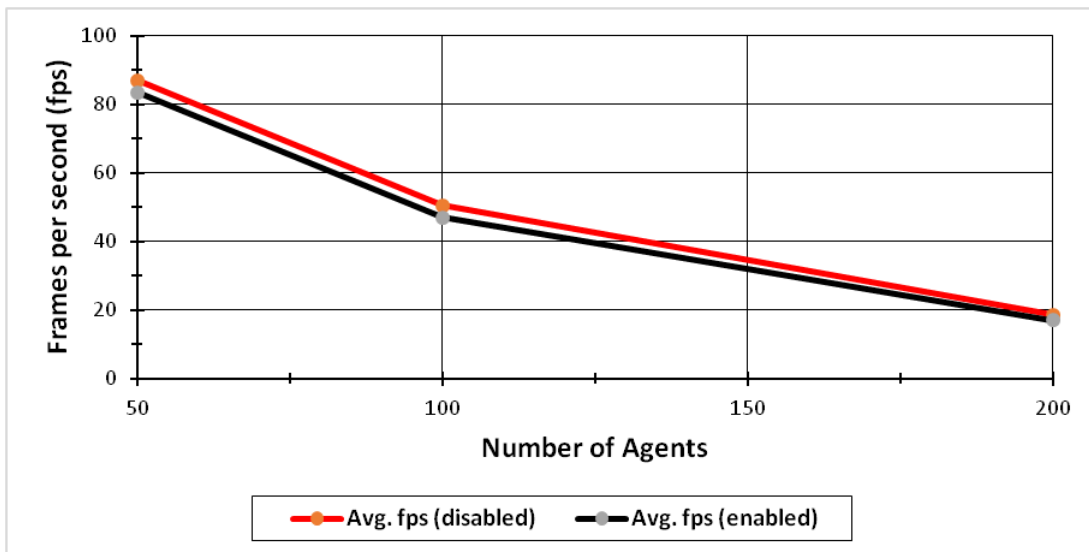
Table 5.1: Simulation performance measurements where gaze behavior scripts are disabled.

Number of agents	Minimum frame time (msec)	Maximum frame time (msec)	Average frame rate (fps)
50	15.3	9.8	83.5
100	24.8	18.5	47
200	67.3	53.4	17

Table 5.2: Simulation performance measurements where gaze behavior scripts are enabled.

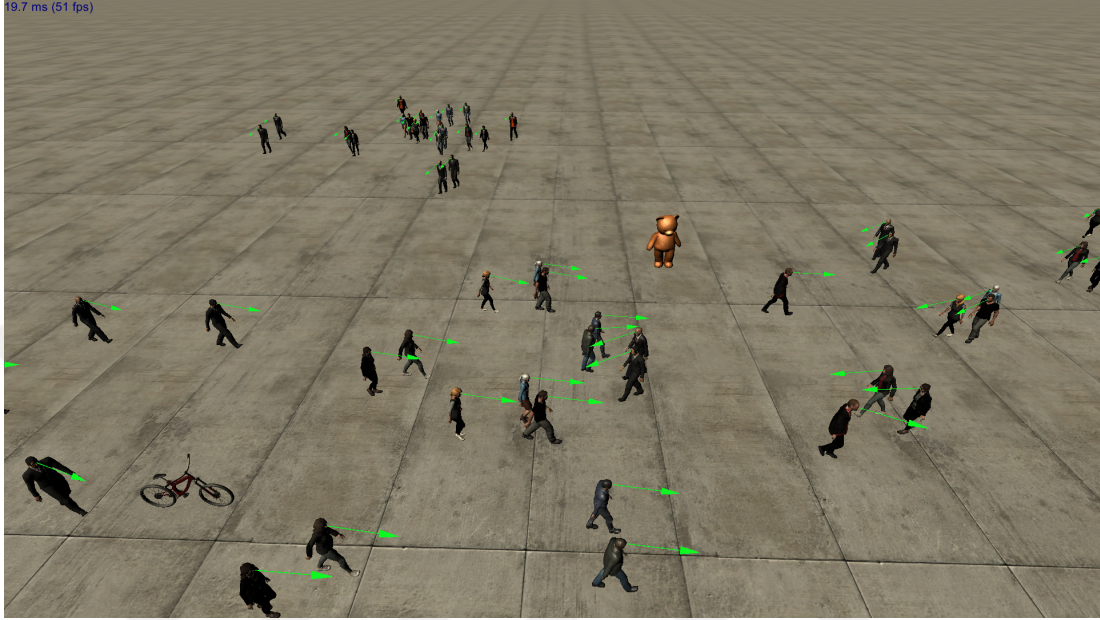


(a)

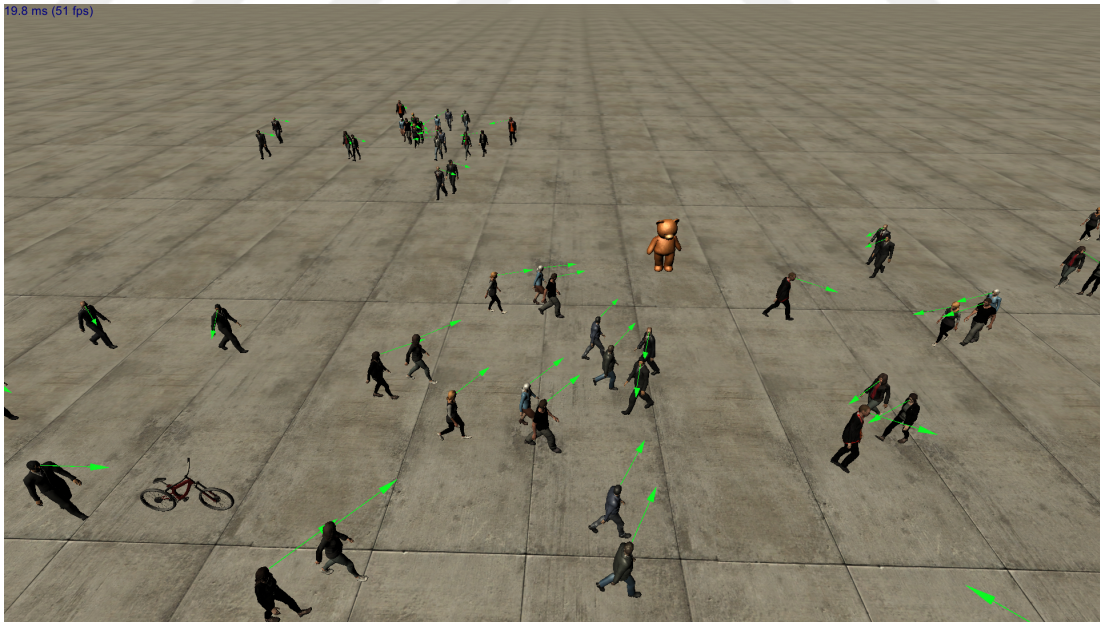


(b)

Figure 5.10: Graphs depicting (a) maximum frame computation times (msecs) and (b) frame rates (frames per second) for different number of agents.



(a)



(b)

Figure 5.11: A crowd simulation with 100 agents. Gaze behavior scripts are (a) disabled and (b) enabled. When the gaze behavior script is enabled, the agents on the bottom left look at the salient object, bicycle. The agents on the top left either look at the teddy bear or look at each other depending on the proximity and periphery scores of teddy bear. The ones on the middle mostly look at the teddy bear.

Chapter 6

Conclusion

We present a computational model for simulating gaze behavior of virtual crowds through predicting interest points at each time stamp. The proposed model aims to provide a realistic gaze simulation by utilizing state-of-the-art psychological studies about grouping of agents, perception of crowds, and personality. The resulting framework is easy to implement and can be added as a plug-in to game engines.

In the literature, there are not many studies about simulating gaze behavior of virtual crowds. And existing studies take only bottom-up approach into consideration and overlook the personalities of agents. Our approach provides a saliency scoring algorithm suitable for perceiving groups of characters, additional top-down saliency criteria, characters with personality, and the implementation of the gaze copy phenomena.

In the experiments, we simulate the gaze behavior of people in captured videos by modifying various parameters and obtain results that are similar to the real videos. Moreover, we highlight the effects of each parameter to validate their impacts on the model. Hence, the evaluation of these scenarios show that the proposed framework is appropriate for generating realistic gaze behavior in virtual crowd simulations.

Nevertheless, this study can be considered as an initial step in the research questions addressed in this study and further efforts are needed to generate more realistic gaze behavior. Some the topics for future extensions are as follows:

- Additional properties can be assigned to both characters and non-human entities (e.g., effects of gender on gaze behavior can be implemented).
- Level-of-Detail (LOD) can be used to give different scores to entities with different proximity.
- Inter-group gaze copying can be included alongside with intra-group gaze copying.
- Effects of group size on gaze behavior can be examined and included in the model.
- More realistic gaze animations can be added. Here we focused only on interest point detection.
- Gaze shift animations and timings can be improved by making characters turn their eyes, head and upper body more realistically. The studies made by Grillon et al. [7] and Oyekoya et al. [8] are good examples of gaze animation.
- Gaze behavior can be combined with steering behavior of agents and may affect their paths.
- The framework can be optimized to handle large crowd simulations.

Bibliography

- [1] G. A. Alvarez, “Representing Multiple Objects as an Ensemble Enhances Visual Cognition,” *Trends in Cognitive Sciences*, vol. 15, pp. 122–131, Mar 2011.
- [2] T. D. Sweeny and D. Whitney, “Perceiving Crowd Attention: Ensemble Perception of a Crowd’s Gaze,” *Psychological Science*, vol. 25, no. 10, pp. 1903–1913, 2014.
- [3] F. E. Fontana, A. Uding, A. Cleneden, L. Cain, L. A. Shaddox, and M. G. Mack, “A Comparison of Gaze Behavior Among Elderly and Younger Adults During Locomotor Tasks,” *Brazilian Journal of Motor Behavior*, vol. 8, no. 1, 2014.
- [4] Unity Technologies, Inc., “Unity Game Engine,” 2016. [Online; accessed 28-June-2016].
- [5] G. W. Meyer, H. E. Rushmeier, M. F. Cohen, D. P. Greenberg, and K. E. Torrance, “An Experimental Evaluation of Computer Graphics Imagery,” *ACM Transactions on Graphics*, vol. 5, pp. 30–50, Jan. 1986.
- [6] J. A. Ferwerda, S. N. Pattanaik, P. Shirley, and D. P. Greenberg, “A Model of Visual Adaptation for Realistic Image Synthesis,” in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’96, (New York, NY, USA), pp. 249–258, ACM, 1996.
- [7] H. Grillon and D. Thalmann, “Simulating Gaze Attention Behaviors for Crowds,” *Journal of Visualization and Computer Animation*, vol. 20, no. 2-3, pp. 111–119, 2009.

- [8] O. Oyekoya, W. Steptoe, and A. Steed, “A Saliency-based Method of Simulating Visual Attention in Virtual Scenes,” in *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, VRST ’09, (New York, NY, USA), pp. 199–206, ACM, 2009.
- [9] K. Cater, A. Chalmers, and G. Ward, “Detail to Attention: Exploiting Visual Tasks for Selective Rendering,” in *Proceedings of the 14th Eurographics Workshop on Rendering*, EGRW ’03, (Aire-la-Ville, Switzerland, Switzerland), pp. 270–280, Eurographics Association, 2003.
- [10] A. M. Treisman and G. Gelade, “A Feature-integration Theory of Attention,” *Cognitive Psychology*, vol. 12, no. 1, pp. 97–136, 1980.
- [11] L. Itti, “Quantifying the Contribution of Low-level Saliency to Human Eye Movements in Dynamic Scenes,” *Visual Cognition*, vol. 12, pp. 1093–1123, 2005.
- [12] C. E. Connor, H. E. Egeth, and S. Yantis, “Visual Attention: Bottom-up Versus Top-down,” *Current Biology*, vol. 14, pp. R850–R852, Oct 2004.
- [13] L. Itti, C. Koch, and E. Niebur, “A Model of Saliency-based Visual Attention for Rapid Scene Analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 1254–1259, Nov 1998.
- [14] S. Lee, G. J. Kim, and S. Choi, “Real-Time Tracking of Visually Attended Objects in Virtual Environments and Its Application to LOD,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, pp. 6–19, Jan 2009.
- [15] P. Longhurst, K. Debattista, and A. Chalmers, “A GPU Based Saliency Map for High-fidelity Selective Rendering,” in *Proceedings of the 4th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, AFRIGRAPH ’06, (New York, NY, USA), pp. 21–29, ACM, 2006.
- [16] V. Sundstedt, E. Stavrakis, M. Wimmer, and E. Reinhard, “A Psychophysical Study of Fixation Behavior in a Computer Game,” in *Proceedings of the 5th Symposium on Applied Perception in Graphics and Visualization*, APGV ’08, (New York, NY, USA), pp. 43–50, ACM, 2008.

- [17] A. McNamara, K. Mania, G. Koulouris, and L. Itti, “Attention-aware Rendering, Mobile Graphics and Games,” in *ACM SIGGRAPH 2014 Courses*, SIGGRAPH ’14, (New York, NY, USA), pp. 6:1–6:119, ACM, 2014.
- [18] T. Judd, K. Ehinger, F. Durand, and A. Torralba, “Learning to Predict Where Humans Look,” in *Proceedings of the 12th IEEE International Conference on Computer Vision*, ICCV ’09, pp. 2106–2113, Sept 2009.
- [19] C. W. Reynolds, “Flocks, Herds and Schools: A Distributed Behavioral Model,” *ACM Computer Graphics (Proceedings of SIGGRAPH)*, vol. 21, pp. 25–34, Aug. 1987.
- [20] 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 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA ’09, (New York, NY, USA), pp. 177–187, ACM, 2009.
- [21] W. Shao and D. Terzopoulos, “Autonomous Pedestrians,” *Graphical Models*, vol. 69, pp. 246–274, Sept. 2007.
- [22] S. R. Musse and D. Thalmann, *A Model of Human Crowd Behavior: Group Inter-Relationship and Collision Detection Analysis*, pp. 39–51. Vienna: Springer Vienna, 1997.
- [23] F. Durupinar, N. Pelechano, J. Allbeck, U. Gudukbay, and N. I. Badler, “How the Ocean Personality Model Affects the Perception of Crowds,” *IEEE Computer Graphics and Applications*, vol. 31, pp. 22–31, May 2011.
- [24] A. Templeton, J. Drury, and A. Philippides, “From Mindless Masses to Small Groups: Conceptualizing Collective Behavior in Crowd Modeling,” *Review of General Psychology*, vol. 19, pp. 215–229, Sept. 2015.
- [25] Z. Fang, S. Lo, and J. Lu, “On the Relationship Between Crowd Density and Movement Velocity,” *Fire Safety Journal*, vol. 38, no. 3, pp. 271–283, 2003.

- [26] C. Shi, M. Zhong, X. Nong, L. He, J. Shi, and G. Feng, “Modeling and Safety Strategy of Passenger Evacuation in a Metro Station in China,” *Safety Science*, vol. 50, no. 5, pp. 1319–1332, 2012.
- [27] C. Dogbe, “On the Modelling of Crowd Dynamics by Generalized Kinetic Models,” *Journal of Mathematical Analysis and Applications*, vol. 387, no. 2, pp. 512–532, 2012.
- [28] F. Qiu and X. Hu, “Modeling Group Structures in Pedestrian Crowd Simulation,” *Simulation Modelling Practice and Theory*, vol. 18, no. 2, pp. 190–205, 2010.
- [29] S. I. Park, F. Quek, and Y. Cao, “Simulating and Animating Social Dynamics: Embedding Small Pedestrian Groups in Crowds,” *Computer Animation and Virtual Worlds*, vol. 24, pp. 155–164, May 2013.
- [30] S. J. Guy, S. Kim, M. C. Lin, and D. Manocha, “Simulating Heterogeneous Crowd Behaviors Using Personality Trait Theory,” in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA ’11, (New York, NY, USA), pp. 43–52, ACM, 2011.
- [31] J. S. Wiggins, *The Five-factor Model of Personality: Theoretical Perspectives*. Guilford Press, 1996.
- [32] S. Chopra-Khullar and N. I. Badler, “Where to Look? Automating Attending Behaviors of Virtual Human Characters,” *Autonomous Agents and Multi-Agent Systems*, vol. 4, pp. 9–23, Mar. 2001.
- [33] Y. Kim, R. W. Hill, and D. R. Traum, “A Computational Model of Dynamic Perceptual Attention for Virtual Humans,” in *Proceedings of the Conference on Behavior Representation in Modeling and Simulation*, BRIMS ’05, 2005.
- [34] L. Itti, N. Dhavale, and F. Pighin, “Realistic Avatar Eye and Head Animation Using a Neurobiological Model of Visual Attention,” in *Proceedings of SPIE 5200, Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation VI*, 64, pp. 64–78, SPIE Press, January 2004.

- [35] C. Peters and C. O’Sullivan, “Bottom-up Visual Attention for Virtual Human Animation,” in *Proceedings of the 16th International Conference on Computer Animation and Social Agents*, CASA ’03, pp. 111–117, May 2003.
- [36] D. Ariely, “Seeing Sets: Representation by Statistical Properties,” *Psychological Science*, vol. 12, pp. 157–162, Mar 2001.
- [37] J. Haberman and D. Whitney, “Seeing the Mean: Ensemble Coding for Sets of Faces,” *Journal of Experimental Psychology: Human Perception & Performance*, vol. 35, pp. 718–734, Jun 2009.
- [38] S. C. Dakin, P. J. Bex, J. R. Cass, and R. J. Watt, “Dissociable Effects of Attention and Crowding on Orientation Averaging,” *Journal of Vision*, vol. 9, pp. 1–16, October 2009.
- [39] A. S. Rancer and T. A. Avtgis, *Argumentative and Aggressive Communication: Theory, Research, and Application*. Thousand Oaks, CA: Sage Publications, 2006.
- [40] O. P. John and S. Srivastava, “The Big Five Trait Taxonomy: History, Measurement, and Theoretical Perspectives,” in *Handbook of Personality: Theory and Research, 2nd edition* (L. A. Pervin and O. P. John, eds.), (New York, NY), pp. 102–138, Guilford, 1999.
- [41] J. S. Coleman and J. James, “The Equilibrium Size Distribution of Freely-Forming Groups,” *Sociometry*, vol. 24, no. 1, pp. 36–45, 1961.
- [42] A. F. Aveni, “The Not-So-Lonely Crowd: Friendship Groups in Collective Behavior,” *Sociometry*, vol. 40, no. 1, pp. 96–99, 1977.
- [43] J. James, “The Distribution of Free-Forming Small Group Size,” *American Sociological Review*, vol. 18, no. 5, pp. 569–570, 1953.
- [44] M. Moussaïd, N. Perozo, S. Garnier, D. Helbing, and G. Theraulaz, “The Walking Behaviour of Pedestrian Social Groups and Its Impact on Crowd Dynamics,” *PLoS ONE*, vol. 5, pp. 1–7, 04 2010.
- [45] E. Kreyszig, *Applied Mathematics*. Hoboken, NJ, USA: John Wiley & Sons, 1979.

- [46] A. C. Gallup, J. J. Hale, D. J. T. Sumpter, S. Garnier, A. Kacelnik, J. R. Krebs, and I. D. Couzin, “Visual Attention and the Acquisition of Information in Human Crowds,” *Proceedings of the National Academy of Sciences*, vol. 109, no. 19, pp. 7245–7250, 2012.
- [47] Adobe, Inc., “Adobe Fuse CC,” 2016. [Online; accessed 28-June-2016].
- [48] Adobe, Inc., “Mixamo,” 2016. [Online; accessed 28-June-2016].

