CGI2012 manuscript No.
(will be inserted by the editor)

More Realistic, Flexible, and Expressive Social Crowds using Transactional Analysis

Abstract Recent algorithms have been able to simulate "social crowds" that allow agents to interact socially as opposed to only treating other agents as obstacles. Unfortunately, past social crowd algorithms lack realism and flexibility because they do not allow agents to move in and out of different and repeated social interactions, are built around a specific obstacle avoidance algorithm, or are tuned only for a specific social setting and do not allow for artist directed changes. We propose a new, simplified social crowd algorithm that focuses on the evolving social needs of agents and allows each agent to join and leave different encounters as desired. Our algorithm is based on the psychology research area of transactional analysis, does not require a specific obstacle avoidance algorithm, and allows for easy artist direction for determining the precise social environment being simulated. Our algorithm runs in real-time with 3,000 to 4,000 agents without the restrictions of previous research.

Keywords Crowd Simulation · Social Crowd Simulation · Transactional Analysis · Pair Walking

1 Introduction

The film and game industries rely heavily on simulated crowds to populate special effect scenes and virtual worlds (see [11] for a game-focused survey). Modern algorithms produce crowds with infrequent or no collisions, but agents often look stiff and artificial since they treat other characters as obstacles and don't stop to talk or walk together as friends. Scientific observation has shown that up to 70% of pedestrians show social interaction as they move [19,15,7].

Recent research has recognized the need for social interaction in crowds, but most of the proposed solutions are very inflexible: they do not allow for changes in who is interacting with whom, they require a specific obstacle avoidance algorithm, or they are designed for specific scenarios and not for artist directed social changes. We propose addressing each of these limitations to create a highly realistic and flexible social crowd simulation algorithm.

Our contributions are:

- A realistic social crowd algorithm with agents that move in and out of conversations with different agents and have repeated interactions with the same agent. This is more expressive than many previous algorithms that keep agents in fixed groups or do not allow multiple interactions. We base our implementation on the transactional analysis area of psychology which studies these types of interactions.
- A flexible framework for social crowds that works with any obstacle avoidance algorithm and allows agents to stop and talk and pair walk.
- An expressive method for making the social nature of our crowds artist directed. Using our method it is easy to change the environment to have the different feel of a work place, a school campus, a public park, etc. This art direction can change the environment in real-time to create a panic situation or to reflect changing socializing, such as when a class break ends.

We validate the realistic, flexible, and expressive nature of our work by demonstrating the existence of key features, including agents interacting with an multiple agents using transactional analysis, a framework that does not require a specific obstacle avoidance algorithm, and artist directed socializing. Our approach runs in real-time even with crowds of 3 to 4 thousand agents.

Fig. 1 Examples of the different scenarios available in our social crowds algorithm including, from left to right, agents talking on a path in a park, a bi-modal distribution of agents at a party (see Section 7.3), a 15-story building, and a globe. The resulting crowds have agents that that stop to talk and pair walk and have believable global effects.

2 Previous Work

Crowd simulation has long looked to psychology and sociology to understand current research about human behavior and group dynamics. Durupinar et al. [9,8] use the OCEAN personality method [10] to create variation in their crowds. These variations include how the agents move (like speed) and how they interact with obstacles (like a preference to move to the right). Pelechano et al. [26] use a presence measure to judge the realism of their crowds. We join this research approach by looking to current psychological and sociological understanding when formulating algorithms for realistic crowd simulation. Unlike previous work, we draw on the psychological field of transactional analysis (see Section 3), which specifically looks at how people interact over multiple social encounters.

Looking directly at social crowds, Musse and Thalmann [20] present a sociological-based algorithm for agents changing groups while traveling between fixed goals. They give results for a museum scenario where such fixed goals would be expected. Yeh et al. [32] use proxy agents to model the inter-personal influence of authority and protection. This is achieved by the creation of proxy agents, or spaces that are considered occupied by the crowd simulation engine but which are not rendered to the screen. Pedica et al. [25,24] proposed an algorithm for agents talking in groups based on human territoriality theory. Carstendottir et al. [6] focus on where agents in a crowd will choose to sit in a cafe or restaurant.

Traum et al. have done extensive work on dialog for immobile agents. Early research [22,23] forced agents to be in constant conversation and all in the same conversation. Later work allows people to join and leave conversations [16] followed by work which allowed agents to move to engage a different person [17]. This work creates believable dialogs but is not designed for large crowds with agents moving more than small distances.

Popelová et al. [27] present research focused purely on two people meeting and then walking together to a destination. The agents involved wait for each other when needed and walk abreast by using a social forces-style approach. They show empirically that this type of grouping is more believable than the simple leader-follower social setup proposed by Reynolds [28]. Karamouzas and Overmars [18] look at how small groups (2-3 members) change their formation as they weave around other groups and obstacles. Their work is primarily based on the work of Moussaïd et al. [19] who took an empirical look at groups in crowds filmed in public places. Moussaïd et al.'s work verifies the prevalence of social interaction within crowds.

Previous work in socializing and group formation in crowds is diverse and successful; however, it lacks several key aspects for the creation of realistic social crowds. First, in real situations, such as those at work, school, or a public park, people constantly move in and out of social encounters. Many recent papers designed for large crowds are not flexible enough to handle both large crowds and fluid changes in social interactions. Second, most papers enforce a specific obstacle avoidance method, forcing users to the specific advantages and disadvantages of their choice. Third, each social scenario has its own set of specific social interactions. People interact differently at work, the bus stop, on a date, or in a panic situation. Almost all previous work has been tuned to handle a specific social situation as opposed to letting the user choose the social feel. This means that if a user wants a different social scenario, a different algorithm must be implemented.

In order to respond to these issues, we present a new algorithm which increases the realism, flexibility, and expressiveness of social crowds. Our approach is realistic since it allows for constant pairing and unpairing and uses transaction analysis to determine how these interactions evolve over time. Our approach is flexible since it does not require a specific obstacle avoidance algorithm in order to run and allows agents to both stop to talk and pair walk. Lastly, our approach is more expressive in the types of environments it can simulate and allows for easy, artist directed tuning to produce a desired social environment.

3 Theory of Improved Social Crowds

One way to differentiate between traditional, non-social crowd algorithms and social crowd algorithms is by looking at the type of reward function the algorithm tries to maximize. Traditional research often presents algorithms for crowds that maximizes a reward function that gives rewards for reaching destinations quickly and avoiding obstacles. Variations to this theme include additional rewards based on agent effort or acceleration (consider [12] and [31]).

On the other hand, the reward function for a social crowd algorithm also includes a reward for social interactions. The movement required for social interactions and the movement required to head towards a destination are often different. Thus, in order to optimize a social reward function, agents must identify when and for how long socializing is more rewarding than heading straight for the destination. One type of common social reward function is as follows:

social reward function

$$SocialReward : a, b, Rel(a, b) \rightarrow reward \quad (1)$$

where a and b are the agents in question and Rel is a function that determines the relationship between the agents (are they friends, married, office mates, enemies, etc.). Some papers can be thought of as having a social reward function that returns a binary value to the Rel function. This results in agents who are really good friends with only a specific small set of agents and who have no relationship to the remaining agents. For example, agents in Popelová et. al's work [27] and Karamouzas and Overmars's work [18] only have relationships with a specific small set of other agents. In this case $SocialReward$ could be implement as follows:

$$SocialReward(a, b) = \begin{cases} High & if\ Rel(a, b) \\ 0 & otherwise \end{cases} \quad (2)$$

This type of reward function results in agents that socialize only with a specific set of people and that never change friends. While this can create believable results for settings where people are on dates or tourists who do not know anyone else in the country, it fails to create the realism of common social situations. As studied scientifically by James and Coleman [15,7], in most cases people move in and out of different social interactions and social grouping frequently. Examples of this include office workers who stop multiple times to talk with different people as they go down a hall, students who walk together with different groups of friends as they go to class, or friends at a park who stop and talk to different people.

Allowing this type of social interaction in a crowd simulation significantly contributes to its realism but adds several new complications. Since friends are no longer rigid and exclusive groups, we need a way of choosing a new reward function that changes over time and accounts for the various relationships between people. Additionally, it means that agents need to walk together, to join and part easily, and to stop to talk if they are heading in different directions.

In this section we address the theory that allows us to address these issues. We first discuss the area of transactional analysis which gives formal guidance on how to create a more realistic reward function. We then cover our flexible architecture for moving agents into stopping to talk interactions and pair walking interactions.

3.1 Transactional Analysis

We propose that far more realistic crowds will be generated by an implementation of $SocialReward$ (see Equation 1) that gives a reward based on psychological and sociological research into interpersonal social interactions. In order to design a more flexible $SocialReward$ in this way, we turned to the psychological area of transactional analysis.

Introduced by Eric Berne in the late 1950s [2], the area of transactional analysis has grown to include its own association and scholarly journal: *The International Transactional Analysis Association* and *Transactional Analysis Journal*. Transactional analysis has also produced a large range of scholarly and popular psychology books (including [4,3,5,13]). Among other psychological issues, transactional analysis studies what the expected length of a spontaneous social activity "ritual" will be. The length of the conversation is determined by the number of social "strokes" that are expected between a pair of talkers based on their relationship and past history. In the context of transactional analysis, a social stroke occurs when one party gives social attention to another party as they interact.

The transactional analysis literature (see specifically [4]) gives two main variables that affect the expected number of strokes in a conversation: the relationship of the agents and the history of recent interactions. For example, agents that have known each other for a long time would feel highly rewarded by a long conversation, while such a lengthy conversation may not be culturally appropriate for people who are not as well acquainted. Additionally, according to transactional analysis, the number of strokes, or the length of the conversation, is related to past interactions. If two people have not seen each other for a long time, regardless of how strong their relationship is, then the number of strokes would be expected to increase. For example, the short greeting between coworkers who last talked a day ago would be

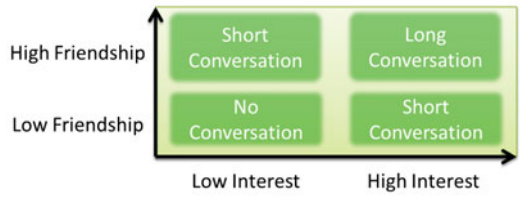

Fig. 2 The expected conversation times based on the two variables we derive from transaction analysis: friendship and conversation interest. Agents with high interest that are friends will have a long conversation. Those with lower interest (because they have talked recently) or a lesser friendship will have shorter conversations.

expected to be longer than if they had seen each other in the hall five minutes before. In our work we think about this past history as an "interest in conversation level" between two agents that rises and falls depending on whether or not they are interacting.

Using transactional analysis principles, a more realistic social reward function than that in Equation 1 would depend both on relationship and past interactions as follows:

$$SReward_{ta} : a, b, Rel(a, b), Past(a, b) \rightarrow reward \quad (3)$$

Where $Past(a, b)$ gives the interest in conversation level for a given set of agents. Using Rel and $Past$, we can determine the length of a conversation between two agents based on their friendship level and conversation interest, as shown in Figure 2. We detail our implementation of transactional analysis-based social crowds in Section 4.

3.2 Flexible Architecture

Previous research has pointed out that when agents socialize there is a mutually understood center of the conversation, called the formation nucleus (see Scheflen and Ashcraft [30] and Pedica and Vilhjálmsson [25]). When agents socialize they usually are within a small radius of this formation nucleus. Social crowd algorithms move agents toward this formation nucleus so they can interact. Note that this formation nucleus may not be stationary if the agents are walking as a group.

Once we have identified which social interactions are rewarding, we need to move agents toward these social nuclei. We focus on two main social interactions where there is a social nucleus: stopping to talk and pair walking. In stopping to talk, agents with divergent destinations stop to chat before moving on. In pair walking agents talk as they move toward destinations with similar headings. Both of these have been implemented in previous work, but we have not seen them combined in the same flexible framework.

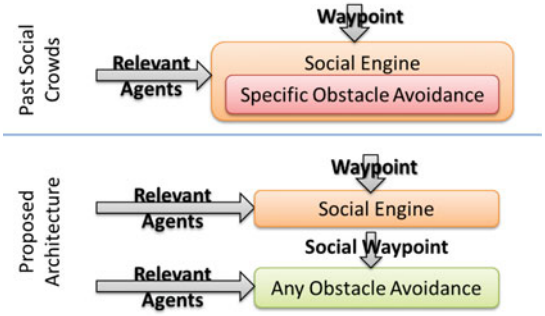

Fig. 3 Abstraction of previous social crowds architecture (top) as compared to our proposed architecture (bottom). Notice that our approach does not enforce a specific obstacle avoidance algorithm.

More importantly than combining stopping to talk and pair walking, we propose a more flexible architecture for moving agents in social situations. Previous work has moved agents so they can interact by manually altering a specific obstacle avoidance algorithm (for example, social forces [27] or a velocity approach[18]). Since each obstacle avoidance algorithm has its respective advantages and disadvantages, users of these previous crowd algorithms are not at liberty to choose the obstacle avoidance algorithm that fits their needs. Figure 3 shows the abstraction of previous approaches that wrap their implementation around a specific algorithm (top) as compared to our approach, which leaves the decision about the obstacle avoidance algorithm to the user (bottom).

To allow this flexibility, we propose an algorithm which temporarily changes an agent's waypoint. We explain how we use this to create two common effects, stopping to talk and pair walking, in the next section. We discuss the specifics of how we implement this architecture in Section 5.

social interaction analysis

4 Implementation: Transactional Analysis

As discussed in Section 3.1, transactional analysis presents a scientific method of determining the appropriate length of a rewarding conversation between two agents. Specifically, the length can be determined using two primary factors: the friendship of the agents and their past history of interaction. Using these transactional analysis principles as a base, we assign a social state to each agent that changes as time progresses. An agent's social state is composed of two parts, a friendship variable and an interest in conversation variable for each other agent with whom it has interacted recently.

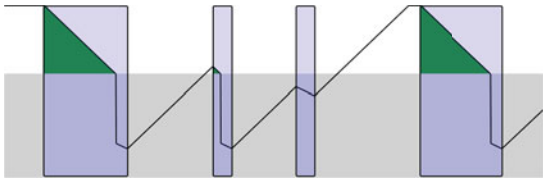

Fig. 4 Example of how an agent's (a) conversation interest with another agent (b) changes over time. The black line represents the interest level of a in socializing with b. Areas highlighted blue are times when agent a and b can see each other. Areas highlighted in green are when a and b are talking. The top part of the graph highlighted in white is where a's interest in talking is rewarding enough to make it worth socializing. The bottom part highlighted in gray represents interest levels that are too low for a to find it worthwhile to talk. Notice as a and b meet frequently, the interest level (and corresponding social reward) drops, so the conversation length shortens and eventually they stop talking. Later when they meet again, the interest level has returned. See Equation 4 and Figure 6.

In real life, friendships have the property of being reflexive and intransitive. Since relationships are built through mutual interaction, the accumulative relationship for a pair of agents should be the same, i.e. $\forall a, b \in A : Rel(a, b) = Rel(b, a)$ where A is the set of all agents. Furthermore, if a and b have a relationship, and b and c have a relationship, then a and c do not necessarily have a relationship, i.e. $\forall a, b, c \in ARel(a, b) \wedge Rel(b, c) \neg \Rightarrow Rel(a, c)$. However, the probability of $Rel(a, c)$ being higher should grow if we know a and b are friends and b and c are friends.

In our implementation, each agent's friendship is defined by an angle in $[0, 2\pi]$. The relationship between two agents, $Rel(a, b)$ in Equation 3, is determined by the angle distance between their friendship angles. If the angle distance is low, the friendship between the agents is higher. If the angle distance is high, the friendship between the agents is lower or nonexistent. We leave it as an artist directed option how large the angle distance can be for agents to still be friends. It follows that this approach has the desired characteristics of reflexivity and intransitivity, with the advantage that friends of friends are more likely to be friends than not.

The friendship angle of each agent is the first half of its social state. The second half is a list of interest values for each other agent. The more recently agents have interacted, the lower their mutual interest level will be. As time passes this interest will increase leading to longer conversations when they meet again (see Figure 4). Formally, we write the interest of agent a towards agent b as I_{ab}. If $Near$ is a function that determines if agents are close enough to notice each other and S is a function that returns true if agents are socializing, we can define the change in interest from time k to $k + 1$ as follows:

$$I_{ab,k+1} = \begin{cases} I_{ab,k} + .01 \; if \; \neg Near(a, b) \\ I_{ab,k} - .01 \; if \; Near(a, b) \wedge \neg S(a, b, k + 1) \\ I_{ab,k} - .03 \; if \; S(a, b, k + 1) \\ I_{ab,k} - .5 \quad if \; S(a, b, k) \wedge \neg S(a, b, k + 1) \end{cases} \quad (4)$$

In other words, the interest level increases when agents are not near each other, socializing agents lose interest as they talk, and when they stop talking the interest drops dramatically. This creates the effects expected in transactional analysis which say that people are less likely to socialize when they have just done so.

Notice that the second line of Equation 4 has agents lose a little interest when they are next to each other but not socializing. This accounts for the non-verbal communication that occurs when people are near each other and not talking directly. Without this slight decrease in interest, agents can get into a social jam where a and b talk while c and d talk until both sets lose interest and then a and c talk while b and d talk until they all lose interest again only to return to the initial social setup. In this case, no one moves and the behavior is unrealistic.

When agents first interact, the social interest value is at its maximum of 1. Since the list of an agent's interest relative to every other agent can grow to be n^2, we instead store interest as a sparse list. If an agent has an interest of 1 toward another agent, then we do not store that interest level in the list. Thus, if agent a comes near agent b and a does not have a stored interest value for b, a adds a new interest level and assigns it a value of 1. After a and b socialize and the interest level returns to 1, the entry for b is removed. As we discuss in our results section, Section 7, using transactional analysis in this way creates believable crowds where agents socialize with many different agents and can interact with the same agent multiple times.

Using these friendship and interest variables we can formally define the reward two agents have in talking to each other. Formally, the reward for socializing, $SReward$, for agents a and b is:

$$SReward(a, b) = Rel(a, b) \cdot min(I_{ab}, I_{ba}) \cdot GlobalI \quad (5)$$

where I is the interest of one agent to talk to another agent based on previous interactions (see Equation 4). The two interest levels should be the same, but we take the minimum in case they are not. We are interested in future work where some agents may regain interest slower than others, in which case this function will still be viable. Notice also the presence of variable $GlobalI$, which is the global interest parameter that can increase or decrease every agents' interest in talking. This allows our simulation to be artist directed (see Section 6). If

the interest level returned by *SReward* ever drops below .5, then the agents consider heading to their destinations more rewarding than talking, and the conversation ends.

5 Implementation: Flexible Architecture

As noted in our section on the theory of more flexible social crowds, Section 3.2, one of our key contributions is that our algorithm directs agents to the formation nucleus without requiring a specific obstacle avoidance function. To do this, our social algorithm directs agents by giving them temporary waypoints instead of altering any of the obstacle avoidance code. Once the interaction ceases to be rewarding, our algorithm returns the waypoint to its original location, and the agent continues on its way. During the entire algorithm the obstacle avoidance function never knows that our agents are socializing.

We can simulate two main types of social interactions: stopping to talk and pair walking. Stopping to talk occurs when two friends meet but their destinations lie in significantly different directions. Thus, the most rewarding experience is to stop and talk instead of going way off course from a destination. Pair walking occurs when two agents have destinations in roughly the same direction. In this case the agents walk together until their destinations diverge.

5.1 Stopping to Talk

Our goal with stopping to talk is to move agents using only changes in their waypoint into the formation nucleus where they can have a rewarding conversation. If two agents, a and b, can see each other and a conversation will be rewarding, our engine creates temporary waypoints for a and b. If initially the agents are not within the range of a formation nucleus (about 1.5m in our work), then the temporary waypoint is in the direction of the other agent. This leads the agent towards each other. Once they are close to the nucleus, our algorithm changes the temporary waypoint to the agent's current location pointing in the direction of the other agent. This makes the agent stop and look in the direction of the other agent. Once the social reward from talking (see Equation 5) drops too low, the conversation ceases to be rewarding, the temporary waypoint is removed, and the agent returns to its course.

Formally, if *Stopped* is the function that gives the waypoint for agent a who is going to stop to talk with b, then we have:

$$Stopped(a,b) = \begin{cases} a + ||b-a|| & if\ |a-b| > 1.5m \\ a & otherwise \end{cases}$$

5.2 Pair Walking

When agents are heading in the same direction, their formation nucleus moves with them. As mentioned earlier, previous work creates this behavior but usually assumes permanent relationships (i.e. certain agents are always walking together), which is less realistic and simpler to implement. Thus, our approach is flexible in the obstacle avoidance algorithm used and expressive in allowing more types of social interactions than previous work.

Once agents have recognized that they will be rewarded by socializing and that they have waypoints in the same direction, the pair needs a mutual destination. Instead of heading to the average of the waypoints, we chose one agent whose path both agents will use as a guide until the angle to the other agent's destination grows too high. We choose this dominant agent by picking the agent whose internal ID is lower. In the future we are interested in more psychologically-based approaches for choosing dominance. Using this approach agents hardly if ever end up in a stuck position when pair walking, something that could happen frequently if a simple average of waypoints was used.

Additionally, we move agents in pairs using only waypoints. Our algorithm must deal with two different situations when agents walk together: times when the agents are close together (within the formation nucleus) and times when an obstacle has split the agents (and they are not close together). In the former case, the agents move together towards their mutual goal. In the latter, the agent that is further ahead waits for the return of the one behind. As noted by Popelová et al. [27], stopping and waiting generally looks more realistic than heading backwards to reunite. Additionally, we have found that it looks unnatural for an agent that gets ahead to start and stop suddenly as the lagging agent's distance falls in and then out of the formation nucleus. Instead we throttle the speed of the agent in front based on angle from the agent ahead to the agent behind. If the dot product of the vector to the agent's waypoint and the friend agent is above 0, the agent will walk at full speed. If not, the agent will slow down proportional to the dot product (see Figure 5). Agents using this method speed and slow down naturally when a friend is gets waylaid as shown in our results, specifically Figure 7.

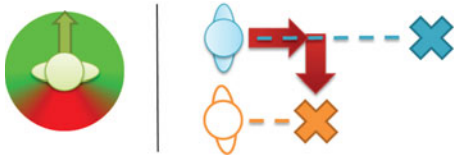

Fig. 5 Pair walking improvements for more realistic pairing. On the left we show how a pair walking agent slows down to let a friend catch up. The cosine of the angle to the other agent is used to choose a speed (green represents full speed, red represents slowing down.) On the right we show how the less dominant agent (orange, hollow agent) finds a waypoint based on the heading of the dominant agent (blue, filled agent). The less dominant agent finds a waypoint a shoulder's width away from the blue agents direction to destination.

We have also found that if agents walking together share the exact some waypoint, there can be oscillations as they get closer to the destination. This comes because as they approach their destination, they get pulled together and then repelled by the obstacle avoidance algorithm to prevent a collision with themselves. To resolve this, we assign the less dominant agent a temporary waypoint that is a shoulder's width way from the direction the dominant agent is heading. We do this by finding the line from the dominant agent to the dominant agent's waypoint. We then geometrically determine which side of that line the other agent is on. We assign the other agent's temporary waypoint to be offset from the dominant direction in that direction (see Figure 5 on the right). As we discuss in our results section, this method proves very flexible since it prevents oscillations even if the dominant agent switches from the left to right (or vice versa) of the other agent.

6 Implementation: Art Directed Socializing

Many of the papers noted in our previous work section, Section 2, focus on particular social situations. For example, Musse and Thalmann [20] model agents in a museum-like environment, Popelová et al. [27] focus on situations where people are paired off on dates, and Karamouzas and Overmars [18] look at small groups with an emphasis on shoppers at a mall. Each one of these algorithms is successful in its goals, but none of them is able to handle a large range of different social environments.

We assert that these varied social interactions do not require their own special algorithm but that with the right parameters a simulation can easily be altered to represent almost any social environment. This environment expressiveness is critical for applications of our algorithm in film, games, and planning where an artist wants to produce a specific social feeling. In order to facilitate this, we have made our social crowds algorithm artist directed using only a few parameters.

Simple

Scenario	Desired Effect	Relationships	Interest
Office/ Work place	Agents stop to talk briefly with most other agents	Lots of friends	Just above the minimum for brief conversations
School Campus	Agents talk longer, fewer friends	Fewer friends	Very high
Park	Some people are friends	Medium friends	Medium interest levels

Bi-Modal

Scenario	Desired Effect	Relationship	Interest
Party Participants	Agents stop to talk at length and pair walk, but not with waiters	Hight with all party goers.	High
Party Waiters	Agents move without little or no interaction	None with party goers.	Low

Changing

Scenario	Desired Effect	Global Interest
Panic Situation	Everyone stops talking and hurry to the exits	Drop to low
Office	People are more willing to talk as they start they day and later in the afternoon	Higher in the morning and afternoon, lower in the middle
High School Halls	People talk during breaks then rush to class	Grows to high during breaks, drops just before the bell rings

Table 1 Table of scenarios that can easily be generated using our two-variable approach to social crowds. Notice we can create simple scenarios where the distribution of variables are consistent, bi-modal scenarios where the distribution is split between two distinct groups, and changing scenarios where the user can change the social environment in real-time.

In adding this expressiveness, to our algorithm we rely on transactional analysis and the variables which define an agent's friends and interest in socializing. As shown in Table 1, a large array of social environments can be created by changing the distribution of friends and interest levels. For example, an office or workplace environment can be reproduced by giving each agent a large group of friends but a lower interest level in talking. The resulting agents will be very likely to stop to talk, but only briefly on their way to their meetings or offices. To allow for artist directed content, we have buttons which allow the user to easily move the dis-

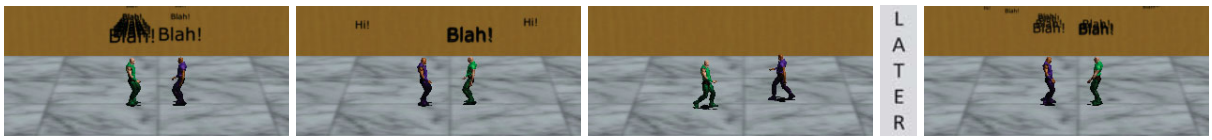

Fig. 6 Four images showing four encounters between two agents (one in green and one in purple). In the first encounter they have a lengthy conversation (conversation length is indicated by the number of word bubbles above each agent). The second image shows the next encounter where they have a shorter conversation (notice that the green and purple agents have switched positions). The next image shows a later encounter where they do not have a conversation, reflecting the reduced reward from socializing due to their previous encounters. The last image shows a much later time when their interest in socializing has returned and they engage in a longer conversation. Figure 4 shows the change in interest level graphically from a similar scenario.

tribution of friends and interest from high to low (see Equation 3).

Our algorithm can be used to create three different categories of social environments: simple, bi-modal, and real-time changing. Simple distributions give the same distribution of friend and interest values to all the agents and they do not change with time. The resulting environments include offices or workplaces, school campuses, or parks. We also allow for bi-modal distributions where there are two types of agents. For example, agents attending a party will have a high interest in talking to other party goers while the waiter agents have a much lower interest as they focus on their job. Lastly, we allow for real-time changes in the interest levels as time passes to reflect changes in the environment. We do this simply by giving the user control over the global interest level. For example, we have a panic button which immediately drops the global interest to 0 (see Figure 8). We can also reproduce smaller changes in the environment, like students who talk until just before class break ends or the fluctuations in socializing at work based on the time of day.

Combined, this straightforward ability to create simple, bi-modal, and real-time changing social environments makes our algorithm expressive in the set of social environments it can simulate.

7 Results

Our contributions are three fold: a more realistic social crowd algorithm that allows agents to interact with multiple agents over time using transactional analysis, a more flexible implementation that allows for almost any obstacle avoidance algorithm, and a more expressive, artist directed way of creating many different social environments. In order to determine if our algorithm indeed achieved these goals, we ran our algorithm with large crowds in a variety of different situations (see Figure 1) including 2D surfaces and 3D surfaces using an algorithm similar to that of Ricks and Egbert [29]. We discuss the results of each of these areas and then give the performance for the system in general.

7.1 Transactional Analysis Results

We verified our social crowd algorithm by looking for agents who move naturally in and out of conversations and who have multiple social interactions with the same agent using transactional analysis principles. In each of our scenarios our algorithm naturally moved agents in and out of multiple conversations, with each having a duration proportional to the social reward of the interaction.

An example of our transactional analysis-based principles is shown in Figure 6. On the left are three images showing successive encounters of two agents. With each encounter the interest level drops and the conversation length shortens (conversation length is noted by the number of words appearing above each agent's head). On the right is a later image showing a longer conversation since the interest has grown with the passing of time. Notice that this simple example follows the principles outlined in our theory and implementation sections about transactional analysis (see Sections 3.1 and 4, respectively).

7.2 Flexible Implementation Results

Our contribution of flexibility in the underlying obstacle avoidance algorithm (see Sections 3.2 and 5) is the most straightforward to validate. We ran our simulation using three different obstacle avoidance algorithms, each of which is distinctly different in its implementation. Specifically, we used social forces based on Helbing and Molnar [14], reciprocal velocity obstacles based on van Den Berg et al. [1], and an anticipation model based on the work of Ondřej et al. [21]. Our crowds were able to move smoothly and exhibited the same key features with each underlying obstacle avoidance algorithm. We

believe that other obstacle avoidance algorithms would produce equally valid results with our approach.

A clear example of our stopping to talk algorithm is shown in Figure 6 where two agents stop three times to hold a conversation. Our pair walking approach is shown in Figure 7. Notice in Figure 1 that our results have the global dynamics we expect from social crowds, including clear clumping patterns as agents socialize.

7.3 Generating Different Social Situations

We can alter our distribution of relationships and conversation interest to model a huge range of social environments or real-time changes in the social environment. Figure 1 gives an example of several different social environments created using the artist directed features of our algorithm.

For a more specific example, consider Figure 8 which shows how we can alter the nature of the social environment in real-time. This figure shows agents in an office talking and the changes that happen when the fire alarm goes off. We simulate this by reducing the global interest to 0, which stops the conversations, and giving each agent a destination at the nearest exit. Later the alarm stops, and we get the agents talking again by raising the global interest level back to 1.

7.4 Performance

We ran our social crowds algorithm doing performance analysis as shown it Table 2. Notice that even with thousands of agents, our algorithm still runs at real-time speeds. All tests were done on an Intel i-7 2600 chip at 3.4Ghz with our program consuming only 10-15 percent of the CPU time. We believe improved parallelization could increase the CPU usage to 100 percent and our results would run even faster. Profiling our program showed that our additional social algorithm used only 2.28% of the computation time for our run with 1000 agents. Notice that our flexible architecture makes this value very easy to compute since our algorithm is modular. We believe both of these results verify our approach as creating a method for realistic results while consuming minimal computational resources.

Agents	FPS	Agents	FPS	Agents	FPS	Agents	FPS
1000	57.2	2000	53.7	3000	30	4000	23.5

Table 2 Performance results without render time. The social crowd part of our implementation consumed 2.28% of the computation time of our program.

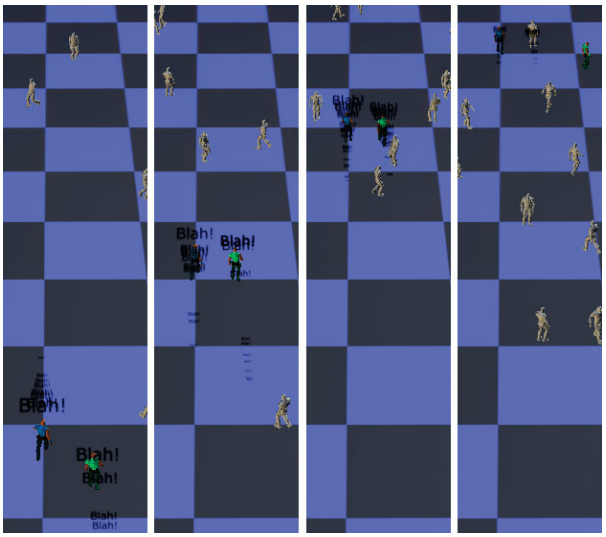

Fig. 7 Two agents pair walking through a crowd using our flexible method for pair walking. The agents wait for each other (left), walk together (middle two images), and part when their destinations diverge (right).

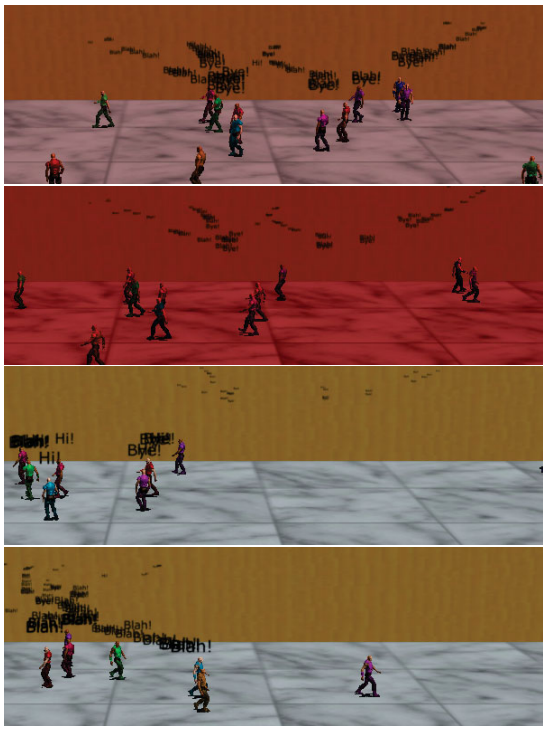

Fig. 8 Example of the artist directed nature of our algorithm. We simulate a fire alarm in an office hall by dropping the global interest to 0 (top image) and giving everyone a destination at the nearest exit. The conversations end (second image) and the agents move to the exits. We change the global interest value to 1 to give the all clear (bottom two images), and the employees return to socializing.

8 Future Work and Conclusion

In terms of future work, our algorithm is limited by group size since we currently can only generate groups of up to two people. Sociological work consistently has found that the equilibrium nature of crowds tends toward smaller groups of two or one, with less than 12% of observations showing groups of size three or more (see James [15] and Coleman [7]), Further, in larger groups people tend to break into smaller subgroups [7]. We believe this means our current results are still relevant and generally applicable, but we are still looking at ways of generalizing our work to larger groups.

We have presented our algorithm that addresses three consistent issues with previous crowd simulation work: unnatural socializing, inflexible implementations, and limited social environments. To address these, our algorithm uses the area of transactional analysis to create believable socializing between agents. Additionally, our simulation is built on a very flexible architecture that allows for almost any obstacle avoidance algorithm. Lastly, we allow for easy, real-time artist direction, allowing the user to choose the desired social environment and even change this mid-simulation.

References

1. Van den Berg, J., Lin, M., Manocha, D.: Reciprocal velocity obstacles for real-time multi-agent navigation. Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on pp. 1928–1935 (2008)
2. Berne, E.: Transactional analysis: a new and effective method of group therapy. American Journal of Psychotherapy **12**(4), 735 (1958)
3. Berne, E.: What do you say after you say hello?: The psychology of human destiny. Bantam (1984)
4. Berne, E.: Games people play: The psychology of human relationships. Penguin (2010)
5. Berne, E., Steiner, C., Kerr, C.: Beyond games and scripts. Ballantine Books (1981)
6. Carstensdottir, E., Gudmundsdottir, K., Valgardsson, G., Vilhjalmsson, H.: Where to sit? the study and implementation of seat selection in public places. Intelligent Virtual Agents pp. 48–54 (2011)
7. Coleman, J., James, J.: The equilibrium size distribution of freely-forming groups. Sociometry **24**(1), 36–45 (1961)
8. Durupinar, F., Allbeck, J., Pelechano, N., Badler, N.: Creating crowd variation with the ocean personality model. Autonomous Agents and Multiagent Systems **3**, 1217–1220 (2008)
9. Durupınar, F., Pelechano, N., Allbeck, J., Güdükbay, U., Badler, N.: How the ocean personality model affects the perception of crowds. IEEE Computer Graphics and Applications **31**(3) (2011)
10. Goldberg, L.: An alternative" description of personality": The big-five factor structure. Journal of personality and Social Psychology **59**(6), 1216 (1990)
11. Gröschel, A.: Towards believable crowd simulation for interactive real-time applications. Thesis, Hochshule fur Technik und Wirtshaft Berlin (2011)
12. Guy, S., Chhugani, J., Curtis, S., Dubey, P., Lin, M., Manocha, D.: Pledestrians: A least-effort approach to crowd simulation. ACM SIGGRAPH/Eurographics Symposium on Computer Animation pp. 119–128 (2010)
13. Harris, T.: I'm okay, you're okay:a practical guide to transactional analysis (2004)
14. Helbing, D., Molnar, P.: Social force model for pedestrian dynamics. Physical review **51**(5) (1995)
15. James, J.: The distribution of free-forming small group size. American Sociological Review (1953)
16. Jan, D., Traum, D.: Dialog simulation for background characters. Intelligent Virtual Agents pp. 65–74 (2005)
17. Jan, D., Traum, D.: Dynamic movement and positioning of embodied agents in multiparty conversations. Proceedings of the Workshop on Embodied Language Processing pp. 59–66 (2007)
18. Karamouzas, I., Overmars, M.: Simulating the local behaviour of small pedestrian groups. Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology pp. 183–190 (2010)
19. Moussaïd, M., Perozo, N., Garnier, S., Helbing, D., Theraulaz, G.: The walking behaviour of pedestrian social groups and its impact on crowd dynamics. PLoS One **5**(4), e10,047 (2010)
20. Musse, S., Thalmann, D.: A model of human crowd behavior: Group inter-relationship and collision detection analysis. Computer Animation and Simulation **97**, 39–51 (1997)
21. Ondřej, J., Pettré, J., Olivier, A., Donikian, S.: A synthetic-vision based steering approach for crowd simulation. ACM Transactions on Graphics (TOG) **29**(4) (2010)
22. Padilha, E., Carletta, J.: A simulation of small group discussion. Proceedings of EDILOG pp. 117–124 (2002)
23. Patel, J.: Simulation of small group discussions for middle level of detail crowds. DTIC Document (2004)
24. Pedica, C., Högni Vilhjálmsson, H., Lárusdóttir, M.: Avatars in conversation: the importance of simulating territorial behavior. Intelligent Virtual Agents pp. 336–342 (2010)
25. Pedica, C., Vilhjálmsson, H.: Spontaneous avatar behavior for human territoriality. Intelligent Virtual Agents pp. 344–357 (2009)
26. Pelechano, N., Stocker, C., Allbeck, J., Badler, N.: Being a part of the crowd: towards validating vr crowds using presence. Autonomous Agents and Multiagent Systems pp. 136–142 (2008)
27. Popelová, M., Bída, M., Brom, C., Gemrot, J., Tomek, J.: When a couple goes together: Walk along steering. Motion in Games pp. 278–289 (2011)
28. Reynolds, C.: Steering behaviors for autonomous characters. Game Developers Conference (1999)
29. Ricks, B., Parris, E.: Improved obstacle relevancy, distance, and angle for crowds constrained to arbitrary manifolds in 3d space. Eurographics (2012)
30. Scheflen, A., Ashcraft, N.: Human territories: How we behave in space-time. Prentice-Hall (1976)
31. Singh, S., Kapadia, M., Faloutsos, P., Reinman, G.: Steerbench: a benchmark suite for evaluating steering behaviors. Computer Animation and Virtual Worlds **20**(5-6), 533–548 (2009)
32. Yeh, H., Curtis, S., Patil, S., van den Berg, J., Manocha, D., Lin, M.: Composite agents. Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation pp. 39–47 (2008)