15-464 Final Project
An Extensible, Modular Crowd Simulation Model

Zachary Kieda (zkieda@andrew.cmu.edu)

May 25, 2015

1 Background

Crowd simulation can animate a large group of bodies, and cause the bodies to interact in different ways. Using a simulation, we can cause a crowd to appear aggressive, tentative, and even scared. The objective of crowd simulation is to give the agents in the group a realistic interaction which invokes an authentic emotion. Actually achieving this goal, however, is very difficult. This form of simulation rides the line of artificial intelligence and animation, and there are many different models proposed to achieve the objective.

Some example models are proposed by Guy, Kim, Lin, and Manocha (1). They discuss *trait theory* to arrive at a crowd simulation model. They use to models - Psychoticism, Extraversion, Neuroticism (PEN) and OCEAN, a 5-factor model. These models provide several different linear, continuous factors that describe the personality of an agent. They use these traits to linearly map to a series of target behaviors an agent will have in the group. The traits they use are the preferred distance to neighbors, preferred maximum number of neighbors, preferred speed, planning horizon, and agent radius. This study used a mass survey to determine a good linear correlation from the traits to behaviors.

Other existing models, like Menge (2), also provide a modular crowd simulation model. Merge splits the simulation into four extensible subcomponents – spacial queries, path computation, goal selection, and path adaptation. These subcomponents are based mainly on the actual animation features and movements of agents in a crowd.

The solution we propose, keeps a different form of modularity to achieve crowd simulation. We attempt to model a better AI to achieve a more advanced interaction among individuals to achieve more advanced interactions among agents. Here, we keep a complete separation among the perception of an agent and the actions they perform. In (1), an agent's personality directly influences his walking behavior. This view is simplistic, and does not offer a concrete model for the current emotions of an individual. In our model, we use the agent's perception to assign a heuristic for their comfort, and for their ability to complete their goal. From the current heuristic, we attempt to maximize our future heuristic value. A more aggressive person would not find proximity to others as important as completing the objective. A more passive person would attempt to maximize their comfort by minimizing their interactions with others. Most importantly, agents will be able to estimate how aggressive the others are in proximity, which will change how two passive people interact, as opposed to two aggressive. (1) does not give a method of determining an agent's comfort, and instead goes directly from the agent's behavior to the actions they take.

2 Overview

For this project, we preset a method for crowd simulation that attempts to balance multiple factors and goals simultaneously for each agent. Each agent acts independently on her own free will, however the actions each agent chooses are a result of the agent's personality, her environment, and her understanding and interaction with her environment. Ideally, we will strive for a model that allows a certainty of information, and allows agents to even have a misconception about their own surroundings which could cause them to make an irrational decision. We use the concepts of *trait theory* to describe the underlying behavior of each individual.

For this experiment, each agent has a series of objectives they will attempt to optimize. Depending on the agent and the situation, an agent may choose to abandon an objective if it appears too costly. To demonstrate this concept, and to explain our goals, we create a game called 'Extreme Capture the Flag.' The simulation was written in Unity and Java, and can be seen in the deliverables. The back-end is written in java, which creates an XML file which describes the positions and emotions of agents. The front-end, written in Unity, takes this XML file and displays the result. This division

was made since the back-end takes considerable time to compute.

3 Extreme Capture The Flag

3.1 Rules

1. There are two teams, Red and Blue. The field is a fixed size rectangle, split in half such that Red is on the left hand side and Blue on the right. There is a Blue flag on the blue side, and a Red flag on the red side.

2. For Red to win, a agent from the Red team must take the Blue flag over to the Red side, where the Red flag was initially located. Vice versa for Blue.

3. Each agent has an initial amount of hit points (HP), and can attack an agent from the other team. While attacking, the attacker must stay in the same location until the attack has been completed.

4. When an agent's HP goes to zero, the agent dies (is permanently removed from the battlefield).

5. Each agent has a color mood and a facial expression, each agent has some level of control over their face. Mood color is directly determined by an agent's internal state.

3.2 Game Setup

There are a few important things that each agent must balance in this game. Foremost, an agent's actions will vary greatly depending on how much they value staying alive. Second, an agent would want for their team to win the match. A third objective is gaining an advantage on the field – for example, removing an enemy could be considered advantageous.

What we stress most for this model is the interaction of the internal and external setting for each agent. Most importantly, for every external indicator, the agent must perform an analysis to translate the external details to meaning that will affect (1) the emotions of the agents (2) the rationale of the agent. These internal models define how the agent will act next. Through this division into internal and external, our system is nicely modular. Adding a feature will fit into one of these three categories: perception,

internal reflection, and action.

We give some examples of features built into the AI that effect the game. An agent has the ability to read the facial expression and mood of other agents. For example, an agent with sufficient perception could be able to detect that the other team is fearful on average, the perceiving agent could become confident in victory (assign a higher heuristic value to their current setup). We built the system such that facial expressions are easier to read than mood, however, facial expressions are easier to feign.

In the next section, we will go over the general model that we are using, and its different components. Then, we will describe how three different components work together. Finally, we introduce the CCCIP model, which is used to tweak the high level behavior of each agent. These high level behavioral attributes are fixed in an agent. These high level behavior translates to many lower level behavioral factors. These lower level behavioral attributes may fluctuate based on the agent's current emotional state. We can think of each of the lower level behavior as a part of a neural network. A behavior has a current uncertainty and a current influence. Some behavioral factors may have a large amount of influence on the character's actions, and others might not. We consider the system of lower level factors as "the internal state."

This internal state is the interface between perception and action. Perception attempts to gather more information, while action attempts to maximize the agent's heuristic objective. This heuristic is affected by the current internal state, along with the agent's ability and certainty in providing a useful heuristic.

When we want to add a feature to our AI, it will fall into one of three categories: internal, perception, and action. An addition to the internal model allows agents to build a more advanced emotional heuristic. Additions to the perception model allow agents to gain a more complex notion of their current and possible future environments. An addition to the action model gives agents a more complex system of interactions possible to reach their goal.

3.3 Game Representation

For this game in particular, we have an objective representation per group, and a representation per individual. The group objective is a Finite State Machine that omnisciently determines the state of the game. Each individual also has a FSM, but this FSM is informed by its perception.

A Few Notes on FSM Notation:
We represent $(A|B)$ as a state machine that can either be A or B, and can move bidirectionally to each state. The superstate $A \times B$ represents the cross product of the possible state transformations. For example, suppose that A, B, C, D are simple states. The FSM $(A|B) \times (C|D)$ has states $A : C$, $A : D$, $B : C$, $B : D$. The following transitions are bidirectionally linked in this FSM, $(A : C, B : C), (A : D, B : D)$, $(A : C, A : D)$, $(B : C, B : D)$.

This allows us to construct a FSM with multiple boolean properties, and traverse between them.

The Group FSM:
We represent a group's FSM while playing Capture the Flag as

$$\begin{aligned} A &= \texttt{Our Flag not Taken} \\ A' &= \texttt{Our Flag Taken} \\ B &= \texttt{Enemy Flag not Taken} \\ B' &= \texttt{Enemy Flag Taken} \\ C &= \texttt{No Flag Captured} \\ D &= \texttt{Our Flag Captured} \\ D' &= \texttt{Enemy Flag Captured} \end{aligned}$$

$$\begin{aligned} C \times ((A|A') \times (B|B')) &\in FSM \\ (C : A' : (B|B'), D) &\in FSM \\ (C : (A|A') : B', D') &\in FSM \end{aligned}$$

Note that this FSM is *omniscient*, implying that the state will change to the correct state, no matter what the individual agents know. The agent FSM, on the other hand, changes according to the agent's current knowledge.

The Agent FSM:
The only transitions that are omniscient in this interaction are when a flag

is captured. The agent immediately jumps to the `Done`, regardless if the agent has knowledge if a flag is captured or not.

$$
\begin{aligned}
A &= \texttt{Holding Flag} \\
A' &= \texttt{Not Holding Flag} \\
B &= \texttt{Being Attacked} \\
B' &= \texttt{Not Being Attacked} \\
C &= \texttt{Attacking} \\
C' &= \texttt{Not Attacking} \\
D &= \texttt{Our Flag Taken} \\
D' &= \texttt{Our Flag Not Taken} \\
E &= \texttt{Enemy Flag Taken} \\
E' &= \texttt{Enemy Flag Not Taken} \\
F &= \texttt{Done}
\end{aligned}
$$

$$G = (B|B') \times (D|D')$$

$$((A \times G) \mid (A' \times (E|E') \times (C|C') \times G)) \in FSM$$
$$((A \times G), F) \in FSM$$

$$
\begin{aligned}
\{(state, F) | state \in (A' : (E|E') : (C|C') : (B|B') : (D|D')), \\
state \notin A' : E' : (C|C') : B' : D')\} \\
\subseteq FSM
\end{aligned}
$$

Notice that from all states, except from the one where we aren't being attacked and no flags are taken, the agent can complete. An agent can reach the `Done` state iff it has either been killed, or a flag has been captured.

Using the current state in the FSM, we modify the agent's internal state appropriately based on the properties of the new state. For example, `Attacking` and `Not Attacking` are properties an agent could have. Once we enter or exit this property, we modify the agent's state appropriately. This allows us to manage the emotions with respect to each property, rather than all possible states.

4 The CCCIP Model

The CCCIP model has five different indicators – communication, confidence, courage, intelligence, and perception. These four indicators determine the

underlying behavior of an individual and how they interact in the game, and range from real values from 1.0 to -1.0. These four indicators change an individual as follows:

- Communication increases the accuracy and precision of an agent's ability to understand how their actions and emotions will influence others. Agents with a higher communication level have an easier ability to change their face and emotions to increase their comfort. Higher communication also gives an individual higher influence on others.

- Confidence decreases uncertainty in actions. This causes an agent to be less prone to changing their emotions or internal state. Having low confidence allows an individual to be more pliable to their environment.

- Courage decreases minimization of risk, making them believe that their actions are less risky than they actually are. Negative courage increases value of minimizing risk, causing them to back out of a situation.

- Intelligence increases the ability to translate perception into a heuristic given non-emotional indicators. Intelligence also gives a more accurate analysis of the risk and payoff of certain actions. This will increase its ability to find a good translation from the individual's perception and internal state to an action.

- Perception increases ability and accuracy of reading the current environment. This includes emotional indicators of other agents on the field. It's important to note the key difference between intelligence and perception – more perception gives an agent more access to more information at a time. On the other hand, intelligence increases the translation of that information to more meaningful data.

We keep two systems to manage each indicator – an ability system and a token system. The ability system changes the influence of an indicator on the agent's internal system. The token system allots a certain number of tokens an individual can spend during a single time-step.

For example, having low confidence will increase the influence of their environment on their internal state – this affects the ability system. On the other hand an individual can only process a certain amount of information at a single time. An agent with low perception and high intelligence will

only be able to see a finite number of things at once. However, this individual will read further into the actions from the information received, and will have a stronger memory of interactions with other agents. An agent with high intelligence and perception will not be able to read in depth into all the information received. So, we use tokens to represent the amount of information processed, and stop when we run out of tokens.

5 Internal State Model

We give the concrete heuristics for the internal state model. The current emotional state of an agent is really a probability space that is a subset of the possible emotions possible by the agent. Suppose we have a fixed n different emotional types that form a basis for an agent's n-dimensional emotional space. We represent the current state as a fixed graph with m edges and some n vertices. For each vertex, there's a corresponding weight and uncertainty. If we just used weights for each vertex, this would define an n dimensional point (and not an n dimensional probability space). The uncertainty over each vertex gives us a distribution that prevents an agent from being too in touch with their own emotions. Adjacent vertices have a correlation factor between the edge connecting them. Two emotions can be positively or negatively correlated. So, we can formally define $S = (V, E, W, U, C)$ where V is our set of vertices, E our edges, W our current weights, U our current uncertainties, and C our correlation factors. V, E, and C are all fixed for an individual, while W, U are modified as the simulation runs. Because of this behavior, we define $S_{base} = (V, E, C)$ as our basic behavior of an agent.

Of course, the initial configuration of S depends only on the CCCIP traits of an individual. In fact, CCCIP changes the initial values of W, U, and determines the values of C for an individual. This is done by a linear translation.

We define the characteristic emotional graph (V, E). This graph, along with the space that it defines, is based on Robert Plutchik's categorization of emotions (3). We also use the characteristic effects and exhibited behaviors from Plutchik's theory to construct a mapping from emotional context to the actions taken.

Vertex No.	Emotion	Adjacent Vertices
0	Happiness	1, 7, 4, 6
1	Trust	0, 2, 4, 5
2	Fear	1, 3, 5, 6
3	Surprise	2, 7
4	Sadness	0, 1, 7
5	Disgust	1, 2, 6
6	Anger	0, 1, 2, 5
7	Anticipation	0, 3, 4

We can define the internal emotional space of an agent as a linear program. We constrain the program such that all incoming emotional values must sum to a value less than or equal to some c_j. We define the constants c, λ, and α. We store emotions in a variable vector v. λ changes the effectiveness of the variable v, c affects the total allowed emotions at a vertex, and α is an offset. When λ is positive, v has a negative correlation with respect to other variables. When λ is negative, v has a positive correlation. For row j,

$$\sum_{i,(i,j)\in E} (\alpha_i + \lambda_i v_i) \leq c_j$$

With additional constraints

$$\sum_{i,(i,j)\in E} (\alpha_i + \lambda_i v_i) \geq 0$$

This translates into the LP

$$M_\lambda \vec{v} \leq \vec{c} - M_\alpha \vec{1}$$

$$M_\lambda \vec{v} \geq -M_\alpha \vec{1}$$

Where M_λ, M_α are both matrices, and $\vec{1}$ is a vector of 1s. We define the matrices

$$M_\lambda(j,i) = \begin{cases} \lambda_{j,i} & \text{if } (j,i) \in E \\ 0 & \text{otherwise} \end{cases}$$

and

$$M_\alpha(j,i) = \begin{cases} \alpha_{j,i} & \text{if } (j,i) \in E \\ 0 & \text{otherwise} \end{cases}$$

Note that

$$(M_\lambda \cdot v)(j) = \sum_i (M_\lambda(j,i) v_i)$$

and

$$(\vec{c} - M_\alpha \cdot \vec{1})(j) = c_j - \sum_i (M_\alpha(j,i) \cdot 1)$$

We use this linear program to bound the feasible emotional state of an agent. We can determine the feasibility of an agent's emotional state using Seidel's algorithm. When we go to a different emotional state, however, we do not find an optimum solution. When our emotions change as a result of our perceptions, this creates a vector δ that will change the current state. We step our current solution by δ to find our new emotional state. When this occurs, we check if the segment intersects with the convex region. If the segment does intersect, our new emotional state is on the nearest plane of the convex hull we intersected with. To detect and find this intersection, we use the Liang-Barsky algorithm for clipping.

Mood color is designated as a characteristic of the internal state model. This is because the mood color is not a decision problem (an action), but rather a direct reading of the current internal state. So, we designate a function

$$transMoodColor : (W, U) \to Color$$

Where $Color = \{a \in \mathbb{R} | 0 \leq a \leq 1\}^3$. The weights W and uncertainties U just define a probabilistic space of emotions. We just provide a linear correspondence that projects this space onto $\mathbb{R}^3$ and select a color that matches the mood. Once the current color is unlikely, we shift it to match the current space. Our translation is the most similar to Plutchik's color encoding to his wheel of emotions.

6 Perception

We survey several different perception indicators. This list is extensible given our model.

- Perception of nearby individuals – their team color and an estimate for their health. This just covers the individuals nearby and their approximate current position.
- Perception of the movement of other individuals. An agent with good perception will be able to predict the movements of more individuals.
- Perception of nearby individual mood, emotion, and action. Based on the color and facial expression of another individual, an agent with

good perception can figure out the mood of another individual with better accuracy and precision. Based on the current position and direction of another individual, one can find out if their action is aggressive, passive, or fearful.

Each perception is a module in the perception category. These are translated to internal state model with a corresponding accuracy of the information. This translation changes the emotions of the internal state model and provides information about the surrounding environment used to make decisions. All extensions to perceptions fall into two categories: the *hlpm* or the *Interest* model.

The high level perception model (*hlpm*) is given by a field of ((*fear*, *comfort*), (*risk*, *payoff*)). This field spans over the two dimensional plane, and has the values $((f, c), (r, p))$ for all points in $\mathbb{R}^2$. These values also have uncertainties. For example, a character might have very little certainty of what's behind another enemy. This field is defined by the function

$$P_{field} : (\mathbb{R} \times \mathbb{R}) \times dt \to ((f \times c) \times (r \times p)) \times \delta$$

We call P_{field} on values $((x, y), dt)$ where x, y are local coordinates, and we estimate the f, c, r, p parameters at local position x, y in future time dt. δ is our uncertainty at our point (x, y). For every plugin we add in to the *hlpm* will have a corresponding influence based on the current internal state model. We aggregate the influence and f, c, r, p parameters over all plugins.

We sample this perception model based on the level of interest. Interest is defined as a function that maps from our radial field of view to our 'interest level'.

$$Interest : \Theta \to \mathbb{R}$$

Such that

$$\int Interest \, d\Theta = 1.0, \ \forall \Theta.\ Interest(\Theta) \geq 0$$

This determines the interest spots in our field of vision that we will likely evaluate. We determine the set of interest points $V_{Interest}$ as the set of ranges

$$|V_{interest}| = k, V_{interest} = \max_{\tau} \{(\Theta_1, \Theta_2) | \forall \Theta.\ \Theta_1 \leq \Theta \leq \Theta_2, Interest(\Theta) \geq \tau\}$$

Where each Θ_1, Θ_2 is a disjoint range in $V_{interest}$, and k is proportional to our perception ability. This implies $V_{interest}$ is just a set of k ranges that are currently interesting to the agent. We use sampling to implement $V_{interest}$,

so something very interesting over a very small range is less likely to be noticed. Again each interest module has an influence over the cumulative interest. This influence depends on the current internal state model.

Before, we discussed the influence of an individual with respect to the communication trait of the CCCIP model. In reality, this influence is just effect the level of interest that this individual will have over the observers. More influential individuals are more likely to be payed attention to than less individuals or events. This implies that the emotions and actions of the more individuals will also carry more weight than less interesting events. We also weight interest based on continuity of consciousness, implying that an object that we were focusing on previously is weighted a bit more than a completely new object. This prevents an agent from having no attention span.

In the realm of perception, we also give the concept of memory. This gives agent the ability to estimate the danger of enemies and allies based on their previous actions. For example, an ally that has destroyed several other enemies can begin to seem something similar to a "badass" compared to other agents. This memory ability primarily keeps an estimate on the level of health on other individuals, along with an estimated level of danger associated with an enemy, or safety associated with an ally. Memory is part of intelligence, and intelligence points are spent on retaining memory or replacing an old memory. Recent memories categorized as highly interesting are retained with very high accuracy until they are placed into "longer term storage", where they are more volatile. Memory discarded goes into the set of "back burner" memories, which implies there is a higher uncertainty in the reliability of the memory as time goes on. For example, we consider the payoff of retaining memory and losing memory at every stage. When a new memory exceeds others in its influence, we evict the least influential memory. When we want to recall information about an enemy or friend, we recall the older memories, but there is a higher uncertainty proportional to the time that has passed since the memory has occurred and the significance of the memory. For example, an insignificant long term memory about another agent will be recalled with high uncertainty, while a significant memory accessed from long term memory will be accessed with higher certainty.

Notice that we have a dt parameter for the P_{field} function, which allows us to predict future *fcrp* levels. These levels are based on movements of enemies and allies and predicted actions. These are influenced by the the

moods and faces of other agents – for example, we can consider an agent's actions as aggressive, passive, etc. Because of future perception, we categorize the perception into two categories – immediate perception and future perception. Immediate perception determines the interest function, determines our current memory, and affects our current internal state. Having more perception increases the amount of information processed during the immediate perception stage. However, more intelligence allows more information to be processed during the future perception stage (which makes predictions). Future perception is determined by the concept of continuity and memory; individuals with more memory have greater memory, and can make more accurate predictions of the future. Future perception can not take in new information, meaning it can only make predictions on the current information available in memory and in the interest points.

Initial Perception is always the first stage of any frame update. First, we determine the initial *hlpm* field and its respective *Interest* field. We spend a proportional number of perception tokens based on the interest field. More perception tokens allows more emotional information and factual information to be gleaned from our external interest point. Next, we translate the immediate perceptual information gained into an emotional reaction of the information. Finally, we perform actions. We use our future perception to decide a goal to accomplish in the future that will maximize our future $fcrp$ levels. From this high level goal, we decide a lower level action to take. This interaction is outlined in greater detail in the next section.

7 Translation from Internal to Action

Classical games (like chess) have a fixed number of possible moves a player can make. An AI can be created that explores these possible scenarios, and assigns a heuristic to each scenario. A classic AI, *minimax*, maximizes the agent's heuristic, and minimizes the opponent's heuristic. In this game, however, is less transparent. The opponent's heuristic is uncertain, and there are many moves possible in a single timestep among all players. For this game, we do not select the correct action that will optimize our heuristic. Instead, we translate our current environment into an immediate heuristic field. This field tells us what positions are more or less 'safe', and attempts to estimate the opponent's moves some time in the future. Only then do we find moves that will probably increase our heuristic while mitigating our risks. An agent with higher intelligence will be able to look over multiple

time steps.

We perform this translation in two steps. First, we take our current internal and our predicted future state. Given these, we determine what kind of high level action we should take – evasive, confrontational, etc. Each high level action maps to a probabilistic space in $fcrp$, some overlapping. Given the high level action we wish to complete in the future, we determine the set of actions that should be taken immediately that will likely make us reach that goal. For each module, all lower level actions are disjoint from one another.

All actions follow this layout. For this project, we define only three actions - changing faces, movement, and attacking. Changing faces determines which face will put on – this face can be strategic or just represent their own emotions. On the other hand, movement entails both the target position and direction for the agent. So, given our current perception and knowledge, we determine the *frcp* field. From this field, we find out which higher level action is best to take (that maximizes our *frcp* value). This high level action can be something like "get back" or become aggressive. This action is determined both by the current field, and by the predicted future situation. From this, we determine the current low level actions that are possible (like movement position and direction), and determine what to do immediately.

Using this generalization of actions allows a client to provide additional behavior that fits into this advanced emotional system with less effort than hardcoding actions directly. We find that this system allows for more complex behavior, since we make actions a result of the emotional system, instead of a direct mapping from traits to behavior.

7.1 Case study : Faces

An interesting feature of this program is the ability of individuals to change faces based on their emotions. Since the ability to change facial expressions is determined by the agent's communication skills, we use communication tokens when we decide to change faces. Since the change of faces can be decided by an agent, it is classified as an action.

The client defines the respective costs associated with each high level action. The apparent cost of tokens is calculated in the back-end. This is part of a general cost/risk associated with spending tokens that an agent takes into

consideration. If the expected comfort of changing faces passes a certain threshold, we take the action. We tweak the comfort heuristic to make the actions more interesting. For example, an agent that's more emotionally unstable will have greater comfort in changing faces to let their face reflect their internal emotions, and the cost associated with changing faces will have less influence.

To prevent erratic face changes, we associate a higher cost to changing faces in succession. After an agent changes faces at time t, the cost jumps to some value c_{max}. After some time dt, the new cost of changing faces will be at least $c_{max} \cdot gaussian(dt)$. Our *hlpm* system will then associate less payoff to changing faces when there is a higher associated cost.

The higher level actions we define for changing faces are *no change*, *comfort self*, *comfort peer*, *confront enemy*, and *yield to enemy*. The face associated with each higher level action can change depending on the situation. In many cases, the spaces associated with these higher level actions overlap, and the agent just needs to search for a good fit. For example, in many situations *comfort self* and *comfort peer* can both be satisfied. The resulting action, however, is disjoint from other final actions. We can only put on exactly one face at a time, and each face is distinct.

8 References

(1) Guy, Stephen, Kim, Sujeong, Lin, Ming, Manocha, Dinesh (2011), *Simulating Heterogeneous Crowd Behaviors Using Personality Trait Theory*. 2011 ACM SIGGRAPH Symposium on Computer Animation

(2) CURTIS S., BEST A., MANOCHA D.: *Menge: A Modular Framework for Simulating Crowd Movement*. Tech. rep., Department of Computer Science, University of North Carolina at Chapel Hill, http://gamma.cs.unc.edu/menge/, 2014.

(3) Plutchik, Robert (1980), *Emotion: Theory, research, and experience: Vol. 1*. Theories of emotion 1, New York: Academic