# Contents

# 1. 2016

## 1.1 March

### Preface (2016-03-04 15:54)

My name is Neil Donald and I am a student of Computer Games Technology at Abertay University Dundee. This assignment will follow on from my previous "Technical Art" project and continue to use the assets of Autodesk Maya 2015 to look more in depth at the dynamics used in films and video game applications. Dynamics such as accurate hair and cloth simulations, convincing particle effect for explosions and liquid simulation and close simulations of crowd behaviour.

For the coursework of this module the students were asked to find a problem area within the film and game industries and execute a project that addresses this issue.
I was originally researching methods of using maya nParticle systems and different applied forces to create more visually interesting and immersive elemental magic representation in video games. Many highly acclaimed video games that use magic as a main mechanic such as Bathesda's Skyrim have the problem where magic ends up not being as spectacular as some would hope. Shamus Young for example writes about how the fighter class has interesting finishing moves where as mages "stand perfectly still

while the enemy stands perfectly still and it holds that pose for a few boring seconds until it ends. They don't explode... or anything cool." Dull actions and animations can reduce the effectiveness of this aspect of gameplay and even put off some from engaging in the magic of the game.

However, even though the representation of magical combat is a part of the field that could be graphically improved, games that use these are mostly still functional when run and the overall game play is not necessarily hindered.

Crowd control in battle simulating games such as the Creative Assembly's Total War franchise and Relic Entertainment's Company of Heroes 2 however is a far more important mechanic as the actions and behaviours of an artificial enemy could make or break a scenario in terms of immersion and challenge for the player. Having an enemy assault your well prepared army in a smart and tactical way with flanking and assaulting weak points can create a believable, difficult enemy to maneuver against and make the player truly invest their efforts in the battle. If the enemy however just charged headlong at the player's most elite units time and time again ignoring other more sensible options in a futile and suicidal fashion the player would not need to think or even invest any effort in the battle and just wait for victory. This sort of situation has been experienced by many reviewers and casual players and can annoy or even ruin a players experience. This sort of situation is stated perfectly in a review by Steve Butts of IGN. He says in a short paragraph how " poor strategic and tactical AI, particularly with smaller forces and sieges, are more annoying", how funny it is when "armies batter themselves against my walls as individual units rather than wait for nearby reinforcements." how annoying it is when "spearmen ignore a wide-open gate right in front of them and raced instead for a different gate on the other side of the enemy fort." "These types of things don't happen every time, but they do happen, and can ruin an otherwise enjoyable turn."

Sometimes though this might be the situation but your most elite troops, as experienced by Rich Stanton when reviewing Rome II Total War, would unexpectedly turn and run with minimum casualties. " and watched with incredulity as they were routed with ease in every battle."

For reasons such as this the aim of my submission for this module is to create a crowd simulation using Maya 2015, to simulate accurate battle tactics and actions taken by a game's artificial intelligence. Primarily concentrating on a attacker/defender scenario such as one side attacking a formation or a siege of sorts and making use of algorithms based on statistics such as a squads strength and moral.

Shamus Young [2014] [review] Availiable at http://www.escapistmagazine.com/articles/view/video-games/columns/experien ced-points/11041-Bethesda-Hates-Mages-12-Reasons-Magic-in-Skyrim-Sucks [Accessed from 26/02/2016]

Rich Stanton [2013] [review]http://www.theguardian.com/techn ology/2013/sep/06/total-war-rome-ii-review [Accessed from 26/02/2016]

Steve Butts [2013][review] Avaliable at http://uk.ign.com/articles/2013/09/06/total-war-rome-2-review [Accessed from 04/03/2016]

## Initial Research and Development (2016-03-14 21:36)

In order to hopefully create a near realistic simulation of a battle for computer games, I felt the most significant aspect to focus on would be the implementation of different statistics and attributes to each of my agents that would affect their own and their comrades actions during the course of the battle.
Looking for resources online I found several articles on the theory of aspects that affect battle, such as morale.
One thought provoking article by Jesse B Fletcher (et al) on simulating wars has a section on "The Spirit of War" which looks a lot into morale. It describes what aspects affect morale such as training, social morphology and recent wins or loses, which I wont look into just now as it would be more suited to long term simulations such as Sid Meiers Civilization or the large scale aspect of Total War. The social aspect would be useful for example if an agent runs or dies another would be affected. Then what morale effects is looked at. Morale compliments both an armies maneuverability and assault effectiveness, as the lower the morale of an agent or army is the less an agent is willing to do and the damage dealt decreases. These two ideas are very interesting and I aim to implement them both into my simulation, but hopefully in a way that is realistic but doesn't take away from the enjoyment of a game in the same way that "top tier" units would flee after a few casualties, an extreme situation experienced by Rich Stanton.
The article then mentions how conscious efforts would be made to increase the levels of these "emotional resource". This is an interesting concept, to think that if a squad had moral or even health lower that a certain threshold then they would hang back or progress slower so that they can recover unless they were specifically ordered otherwise. This would be a good feature to simulate in my project, especially since I hope to have a smart "AI" that would aim to assault a position to the best of it's abilities.
This particular article as well as some aspects of others have been useful in gaining an idea of the theory behind what I aim to achieve, but I shall continue to research this and practical techniques to help me with this task.

As of this week I have implemented a base script for the agents of my scene to work from, which applies the flocking behaviour of a squad in my crowd simulation. The three parts of this being separation, cohesion and alignment, which causes the agents to avoid each other while still maintaining movement within a group using average distances and movement headings from all relevant agents.

My aim for the next stage is to implement separate squads that are affected by disconnected behaviours from each other, as well as apply the statistics to each agent and squad such as health, strength and morale. I will also aim to find more specialized resources on implementing these features.

Fletcher et al [2011][professional article] Avaliable at http://escholarship.org/uc/item/7hk4279k [14 march 2016]

(As described in previous post)

Rich Stanton [2013] [review]http://www.theguardian.com/techn ology/2013/sep/06/total-war-rome-ii-review [Accessed from 26/02/2016]

---

### Single Agent behaviours and movement (2016-03-28 19:45)

The foundation of crowd simulation, in which ever scenario it may be used in, is the behaviours of each individual agent and their interactions with every other object and agent in the simulation. With this in mind and with the aim of realizing an appropriate morale algorithm for use in my battle simulation, I searched and found articles discussing "Steering Behaviours" by Craig Reynolds. In his articles he talks about "Seek and Flee" and then the related "Pursuit and Evade". The foremost is an interesting concept as it talks about a target acting almost like gravity to "boids", as he refers to them, that seek and a repellent to those that flee and that no matter what is affecting the boids movement they will always verge towards or away from the target. Pursuit and evade link with these behaviours but the boids anticipate where the target may go. These ideas clarifies what I was already musing over in terms of the behavours that will be applied to my attacking and defending agents. I am now thinking that my attackers would always seek a target but maybe avoid defenders that are stronger than they are or routes that lead to more defenders than they could handle. The defenders could pursue attackers that come too close or are maybe bellow a certain health thresh hold so as to finish them off. Then of course if a squad or agent has no morale they would flee all dangers to them. Even though these articles of Reynolds did not discuss methods of implementation they do provide many references to other theoretical articles, and all of these have helped me simplify my pseudo code and my future planning for my squads actions.

The goal this time was to apply battle behaviours and attributes to the different agents and their squads, however I had not full developed the base movement towards a set target. So I felt that I would do this first. I designated the area that I wanted the agents to head towards by creating a cube object to act as a "flag" so that I could have the agents move towards it's position. Several attempts to add the direction of the flag in relation to each agent has not gone as simply as I originally thought, meaning that I have spent a lot of my allocated time getting this to work. Though now I have what I consider a main step done I'll be able to progress hopefully quicker.

With this flag system I thought maybe in the future if not in the next step I could have a way point system as seen in games such as Hellbent's Supreme Commander or Ensemble's Age of Empire. I could have a pop up window like a command console that allows the user to put down more flags that the agents will head towards. I have already spent a few moments implementing a basic pop up menu that creates the "flag", so the ground work is there for use in the future.

Craig Reynalds [2004] *Steering Behaviours for Autonomous characters* [professional article] Available at http://www.red3d.com/cwr/steer/ [28/03/2016]

---

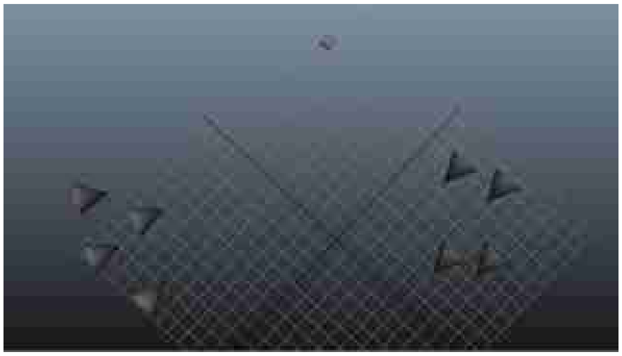
## 1.2 April

### Team assignment and Agent Ranking (2016-04-14 19:19)

Recently I have been prioritizing on my other course works but have been able to move forward and apply teams to the agents of my project, as well as apply the foundations for Leaders of my squads which I hope will solve problems I have been having with the agents reaching my "flags" targets.

Not much to see as this project is focusing on the functionality of crowd control making use of very simple graphical representation, but here is the general idea of what I have so far:

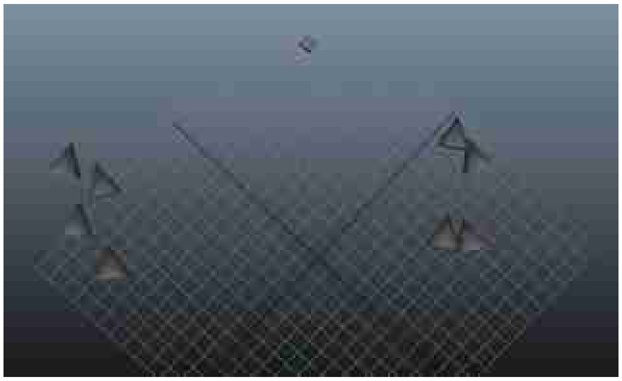
[1]



From the above image you can see the initial positions of the two groups of agents represented by simple cones so as to clarify which direction they are heading, and the cubic "flag" at the top corner of the grid.

Below you can see group movement in affect as the agents clump together and head in the general direction of the flag. At the moment the agents clump together and don’t head directly to the flag, instead heading past it.

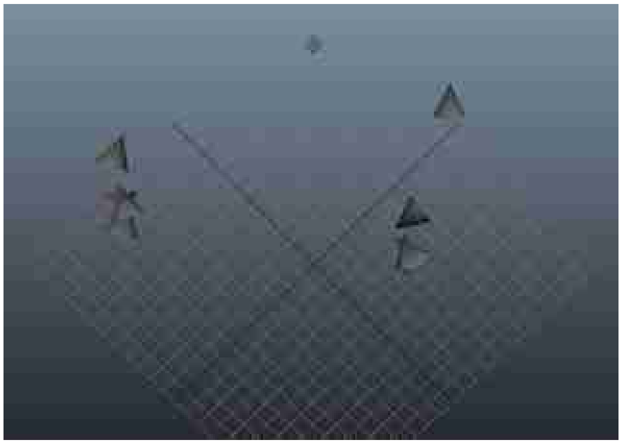
[2]



I was also working on applying team movement, with each agent acknowledging which team their neighbours are in. I will have each agent attack or flee from their enemy later on in development, but for now I will just have the agent simply move away from their foes by not applying cohesion and alignment behaviours between them.
When working on getting my agents to move in terms of their teams, I came across problems when trying to affect the squad movement towards targets flags. Looping through each agent and then looping through each of their neighbours to see if they are on the same team meant for some reason movement wasn’t being applied. After trying out different methods to get this to work I asked my lecturer who pointed out my rendering in Maya was set to 24 frames-per-second and not real-time, which meant as my code was taking longer than 24 frames to run the movement wasn’t being shown.
Fixing this I can now see my agents moving as they were before but now a bit slower as the script is being processed. For now the lagging movement is still workable but perhaps once I have all I want to include in this project I may aim to make the script more efficient.

After reading a thesis by Edgar Rodriguez and considering my problem with moving towards targets, I felt that i would implement "leaders" for my squads that would go towards the flag with my other agents being "soldiers" and following behind. Using the theory from page 59 I modified the alignment feature for the specified leaders so they simple move towards the flag.

[3]



As show above this does not necessarily work well. If you look on the right side you can see that my leader for that squad is merrily moving along, not even in the correct direction. The rest of the squad is huddled together now moving aimlessly as the leader is out with the cohesion range. This is not so bad on the left side squad as the agents started closer together but this clarifies many now obvious problems.
Firstly the alignment directional code does not provide the correct direction of the flag from the agent.
Secondly, my separation scripting is not keeping the agents from merging in with each other which would definitely make any accurate collision detection impossible.
Finally having the soldier agents follow the leader through their cohesion code would not work if later on the soldier fled from battle but wanted to return. This could be fixed by using the same technique described in the thesis paper which would be to loop through the agents in the squad and follow the next one down the hierarchy.
These three problems will be my main aim for my next post, and with these completed I will have the base of my project done and will be able to move onto the actual battle simulation.

Edgar Rodriguez [2010] *Behavioral Animation for Maya Particles Using Steering Forces*
http://tigerprints.clemson.edu/cgi/viewcontent.cgi?article=1956 &context=all _theses [Accessed 14/04/2016]

1. `https://4.bp.blogspot.com/-iinijMtOsWY/Vwag1WpPXYI/AAAAAAAAAKY/mvdMAPtgIWcUjgAxAO-MooZFrmSkq92HQ/s1600/StartOff.png`
2. `https://1.bp.blogspot.com/-Bbaj4YdL9yg/Vwag1cd2-dI/AAAAAAAAAKc/SsiIxngAnwIiEmZlRYu7ohV3PHcJPdfKQ/s1600/moving.png`
3. `https://3.bp.blogspot.com/-bUmH7AWUMPk/Vw_a8mWaIFI/AAAAAAAAAKs/PKHFbw_UO6I75nZFjAiEck1GN3ZUuCGvwCLcB/s1600/leaders.png`

## Base Team Strategies **(2016-04-21 22:01)**

In my last update I had some problems that I postponed fixing to do with the base behaviours of my agents. This time I have sorted these problems and made further progress through the stages of my project.
I fixed the first problem where my agents would not move towards their target correctly. It turns out that my trigonometry for working and moving between the agents own position and the targets position was implemented incorrectly and with the wrong position vectors, but this was easily corrected. I had the target code in a separate function from my alignment behaviour, and for the mean will am using target finding over alignment, as I shall explain beside my leader/soldier layout.

I had a separation issue where my agents would faze into each other rather than stay at a distance. I've reduced weighting just now, but this is not a permanent solution nor does it exactly fix the problem. I an implementing a separate range check from the alignment and cohesion behaviour, and this range check will most likely coincide with my attack range of my agents. Thinking in terms of swordsmen, they would want to be close enough to hit an enemy with their sword but not close enough for their friend to accidentally hit then with one, so they would move away.

I have decided to leave the implementation of my agents soldier/leader ranking for the time being, I'm aiming for all the agents to behave in the same anyway. Implementing individual behaviours for this base stage will yield the same results as if I have "follow the leader" functionality when I have my agent moving towards different flags. I am also having the agents simply detach from the group and engage the enemy if there are any within a certain distance, including the leader. The leader will become more influential in the scenario later on when I implement "commands" that the leader will issue when evaluating an enemy team's strengths, commands such as "hold fast" or "charge".

Aside from these I have change the layout of my script, as well as set down simple "states" for my agent to show what they are currently doing.
The way my script was previously set out it was difficult to designate an enemy I wanted my agents to attack if they were within range. Instead with the way I was originally looping through neighbours of my agents the target would be over-written. As well as that the weightings I would have of the influence of my separation, cohesion and targeting functions were having underwhelming or overbearing results compared to what was needed. With help from fellow students the layout will become clearer and more predictable for my next update.

## Submission Extension and Attack Behaviour (2016-04-30 18:49)

Recently has been the time period of my second semester final submissions. Unfortunately due to several setbacks for this and for other modules I have been unable to invest the necessary amount of time to each of my submissions, forcing me to miss the original deadline for Dynamics. The university has granted me an extension till the 3rd of May for this and another module, though with this time I feel that I will not be able to create a polished or fully developed battle simulator that I had originally planned for. However with the time I have I may still be able to downgrade this to a bar brawl simulator, where two teams of sports fans act and attack without displaying such structure or strategies a trained military squad would display.

Since my last update I have amended some of my code to make it more effitient and functional as well as put in health, strength and morale variables as well as effecting agent attacking behaviour using the states i implemented previously. Each state value is the acknowledgement of a behaviour. State 0 is Defeated meaning the agent will not be part of any calculations; 1 is general group flocking towards a target; 2 is for when an agent is rushing an enemy and 3 is if an agent is fighting. I would like to implement more behaviours such as fleeing or defending but for now these shall do to display base behaviours.
Initially all agents are set to state 1 and move towards the target, and as the agents of both teams enter a specific range from each other they get set to state 2 and charge their closest enemy. If the enemy is close enough to hit the switch to stance 3 to fight and deplete the enemy's health, and once an agents health is brought to zero they are set to state 0 and no longer affect the scene.

I have had some trouble with changing the agent states at times. Sometimes simple syntax errors went unnoticed for a while while other times conditions weren't being identified as being set due to overwriting from for loops and if statements involving each neighbour close to the specific agent.

Several issues were seen when enemies were set to attack each other. There is also a major problem with the agent _fight() function that is currently breaking the attacking behaviour. When an agent is set to rushed and then again to fight it should search for the closest enemy near to it to be attacked. However the agents are only recognizing themselves and so attack themselves. I was unable to find the problem to this just yet, though I think the problem maybe situated around the search for enemies or the distinction of the closest agent.
Another issue is that one agent will always turn away from the enemy and move in the opposite direction from their opponent, which they should not do. Assessing the targeting function showed that this may be due to me having used the function previously used for a stationary target. Having a moving target may affect where the agent would think it is which it should but in a wrong manner. This may also be because the agents see themselves as their target,so solving this issue first may deal with the two problems.
From this issue of a chase it is also shown that the agents will keep chasing an original enemy target even when there is a closer target. This again could be solved or at least a solution may present itself when the agent targeting is corrected.
This I will aim to solve for the start of the next session. With this solved the base behaviours of the agents will be ready to be worked upon for any additional features I may have the time to achieve, such as more states and leader orders.

## 1.3 May

### Solving Problematic Behaviours and Final Amendments (2016-05-03 18:26)

Since my last update I have managed to solve the problem with my agents turning away from enemies instead of moving towards them. Looking back at internet resources I found towards the beginning, such as Shiffman's (2012) book and Reynold's (1999) original descriptive thesis, I looked agian at the steering forces that could affect an agent. Originally I was using the same targeting function to find heading vectors for stationary targets as well as when I was looking for moving enemy agents. I had thought previously that this may have been an issue and noticing inconsistencies between my code and the theory I made a copy of this function and set the position vectors of the agents to subtract rather than add to a agent's heading vector. The agents now move together as required, meaning that adding the positions together to get the heading value was actually forcing them away from each other, making them flee almost.
With this in mind I solved my nearest enemy designation, which required an amended array of the agent's neighbours, and moved onto implementing a simple team "Aura" and fleeing functionality.

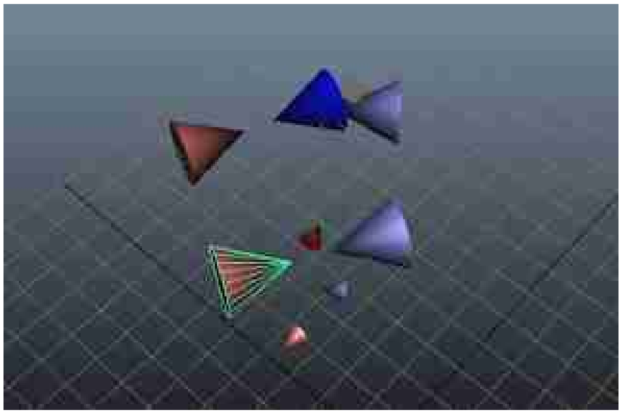
With the time I had left before submission I was able to amend the targeting code so that a agent would not travel backwards on occasion after fighting and adjusted several functions to remove unnecessary code for effitiency. I also set up the base for morale affects on the agents, however I was not able to have events, such as an ally's death or if their health was too low, affect the morale before I had to submit this. This was due to the way I searched for an agent's neighbours and set states as well as the speed at which an agents health would be depleted, meaning they would not have time to run. If I had managed to possibly implement transition states showing that an ally is "dieing" rather than defeated, this would allow me to catch this transition and only affect the morale of the agent that one frame rather than constantly after the agent died. A time delay would also be beneficial for the fighting stage so the enemy would not be defeated more or less instantly after three frames, and this would allow possible animations to be implemented in the future.

If time had allowed for this submission I feel that I would have been able to have made the simulator or a

close example that I had originally intended. I would have liked to have utilized a team "aura" that would affect a "leader's" decisions during any battle, allowing for extra agent states such as "hold fast" where the agents would stop and form together in a formation with an additional increase to morale.
I had also hoped to implement obstacle such as city walls or natural choke points that would emit their own aura that the leaders would evaluate and set orders to approach this.

Here is a image of my final submission just as an idea of what I was able to accomplish:

[1]



Shown here is an image of about halfway through the simulation, after the agents have already traveled from their starting positions to the the grey cube just visible in the center top behind two blue team agents and are now rushing and fighting an enemy. I have coloured the agents so as to distinguish which team they are on, and have coloured the agents, who currently have the leader attribute, darker shades although sadly this trait has not been utilized as of the submission. You can see at the bottom where most of the fighting has already taken place, with the agents that are smaller than the others being defeated and no longer having a role in the fight. Each agent is allocated random values for they're strength and health meaning that every run through has a different result. In this run through the blue have the upper hand at this moment, however the remaining health of the blues might be low enough that the strength of the reds may over come them.

I am fairly happy with what I have manged to achieve with this simulator and what I have learned from it. Researching into the different aspects of autonomous agents in simulators has given me an appreciation of the work involved to implement a realistic representation of a battle scene or even pedestrian movement. With time to come I feel that I could employ the techniques that I have learned and create the final basic functioning battle simulator that I had originally aimed for.

Craig W.Reynolds [1999][online thesis] *Steering Behaviours for Autonomous Characters* [Available from] http://www.red3d.com/cwr/steer/gdc99/ [Accessed 03/05/16]

Daniel Shiffman [2012][online book] *The Nature of Code: Chapter 6: Autonomous Agents* [Available from] http://natureofcode.com/book/chapter-6-autonomous-agents/ [Accessed 03/05/16]

[1] https://4.bp.blogspot.com/-o6zS1ZAd5ak/VyjUQppFjeI/AAAAAAAAALA/xtlSskCkpaAiBRJ5FwPmYj5aNOSFugXzQCLcB/s1600/War.png

BLOGBOOKER

BlogBook v0.9,
$\LaTeX\,2_\varepsilon$ & GNU/Linux.
`https://www.blogbooker.com`

Edited: May 3, 2016