The Illuminati Tension Manager

Larry LeBron
University of California, Santa Cruz
1156 High St, Santa
Santa Cruz, CA 95064
llebron@soe.ucsc.edu

Paul Maddaloni
University of California, Santa Cruz
1156 High St, Santa
Santa Cruz, CA 95064
pmaddalo@soe.ucsc.edu

Abstract-Tension is a major aspect of involving players in a game experience. Dynamic tension management allows authors to provide a desired tension arc to their game without having to confine players to a strict, pre-scripted path. While this has been previously attempted by a small faction of computer scientists, there remains much research to be done in this area. The *Illuminati* tension manager is our prototype solution to the notion of dynamic game tension management. In this paper, we describe the system underlying *Illuminati*, and our experience testing it with *Jumper,* a simple text-based client game. Our eventual hope is to help such technology be available and oft used in game creation.

Index Terms
gameplay, playthrough, playability, replayability, tension arc, *Illuminati* drama manager, tension manager, prescripted, mechanical affordance, formal affordance, play-testing, goal axis, <failure, success> impact.

1. Introduction

Video games can allow for complex interactions with their environments. These interactions are typically static and inflexible, and rely on pre-authored stories and content to deliver narrative layered over gameplay. Many games follow a strict story path, and have corresponding play elements that halt story progression until completed. A game's sense of tension is thus pre-scripted by the game's author(s), which may or may not be able to achieve the desired arc sought by the creators. Creators can give players a limited number of ways to interact with their game, forcing players to experience things in a certain order that authors depend on to maintain their desired narrative tension. Attempting to provide freedom can give players more latitude in how they approach the game story and its world, but may miss the mark in terms of a desired tension because of the unpredictability of players' actions.

A tension arc can be seen as a desired progression of player emotion and tension as a game is played. A typical arc found in many games, movies, and stories in general, starts with low tension that initially increases slowly, but ascends more rapidly as the user progresses toward the end of the experience, eventually reaching a climax. Different variations of this tension arc can be achieved with fluctuations of hills and valleys, or even a reverse tension arc where the beginning of an experience is most tense for subjects.

This raises the question of why should game creators and players care about users experiencing any type of tension arc? Authors can depend on their gameplay providing some type of tension, such as progressive difficulty increases as players get farther in a game, culminating in boss battles and/or some ending sequence. Alternately, the story can provide some or all of the tension, following a typical Aristotelian story arc, with the tensest moments coming at the end of a narrative. In many games these two ideas are presented simultaneously, and when done in this matter can be seen as different experiences, causing the two to have no real bearing on each other.

While some games have attempted to have the two work in tandem, this is a much less common approach. Games such as *Heavy Rain* [1] attempt to have all player action affect the story, but player interaction boils down to quick-time events, providing players with little leeway in play style or action. *The Walking Dead* [2] presents players with a similar experience, offering limited gameplay complexity, and relying on a seemingly deep choice system for character interaction. Unfortunately, due to its use of static, pre-authored events, story arcs can never diverge far from a main arc that always leads to a similar conclusion.

The majority of games take the more divorced approach, where story is an overarching structure that is told statically to the player, and pieces of gameplay are interspersed throughout. In this method, not much attention is paid to the two locking solidly together, or informing each other. *Gears of War* [3], *Mass Effect* [4], and the *Grand Theft Auto* [5] series are all examples of gameplay not allowing for much, if any, flexibility in main stories (the Mass Effect series being the most highly criticized for its faltering in this aspect). Players are free to play missions in any way that they see fit, but the story is not significantly influenced by the resulting gameplay.

A small collection of mainstream games has attempted to make some actions in gameplay affect storytelling in games. For instance, *Call of Duty: Black Ops 2* [6] determines which parts of its main story to tell players depending on how fast they are able to achieve some mission objectives, such as stopping the enemy from destroying intelligence, or saving VIPs. Unfortunately, even this modest approach is not in wide use. This example, as well as other games which attempt to give players more influence on the story through mechanical affordances and player actions, is more of a novelty rather than a staple of large commercial games at this time.

We intentionally developed our tension management system, *Illuminati*, to allow for games that bridge gameplay and dynamic story. When queried by a client game, the manager suggests player choice options for maintaining a desired game tension arc. Depending on player choices and their outcomes, the story is changed, as well as what choices are presented to the player, all while maintaining the desired tension arc provided by the author.

This is important because it allows a game to adapt to a player's actions in a particular playthrough of a game. Likewise, the tension manager ensures that player experience will change relative to the specific choices they make. This allows for variable replayability, because the probability of being presented with duplicate playthroughs is extremely low. The system allows for a directed arc that the designer determines, without having to worry about scripting a specific sequence to ensure that arc. Instead, developers using our system can craft procedural game elements (missions, NPCs, etc.), which will be instantiated based on the tension manager's suggestions.

There are some notable entries into this area of research that have influenced and inspired our approach. From the academic community, the *EmPath* and *Facade* systems attempt to control story plot by manipulating the game world, as well as the mechanical affordances given to the player. On the commercial side, some large companies such as Valve, Epic, and Ubisoft, have attempted to use dynamic systems in their games to control game tension and story by observing player behavior. For instance, Epic's *Gears of War: Judgment* [7] keeps track of player performance, and dynamically changes enemy spawns and tactics based on player actions and strategy.

In designing this system, we sought to create a general, engine-like tool, which can be utilized by a client game to maintain a specified tension arc, while fostering adaptability and gameplay variability. In the coming sections we will discuss related research in this area, and then move into an examination of the underlying tension model upon which *Illuminati* is based. Finally, we will discuss our experience testing *Illuminati* with a simple client game, along with the successes and limitations of this prototype system. In the future, our hope is to expand this work into a functional, general tension engine for any designer to employ.

2. Related Work

One attempt at dramatic management through modification of game elements is Lewis, Sullivan, and Chen's engine *EmPath* [8]. The system's drama manager manipulates the world based on author-specified evaluation mechanisms that enforce "story goodness." The game evaluates what has occurred in the game world thus far, and determines how and which plot point to present next by searching through possible future game states. It offers these plot points by adding or removing things from the game world. To receive plot points, players can find hint notes, garner items dropped by enemies, or gain information given by NPCs. The system only attempts to affect drama or tension in the game through gameplay elements, instead utilizing to story components to deliver narrative points, omitting direct control over tension by the system.

Another related work is *Suspenser, a Computational Model of Narrative Generation* developed by Cheong and Young [9]. This system takes a story world comprised of possible events, actions and repercussions as input. It then outputs a sequenced story, with the intention of matching author-specified suspense levels for story points. The system models an event's suspense based on a model of the reader's perceptions. The reader model contains the hypothetical reader's notions of possible plans for the protagonists to accomplish their goals. Overall suspense is then defined as the inverse of the number of planned goal solutions. *Suspenser* measures the potential suspense of actions based on summations of action-effects that might threaten or support those goals. It then picks the closest fitting matches to add to the output story.

Cheong and Young present a well-defined model with some supporting sample output, but admit their implementation does not match the complexity of the underlying theory. In its current implementation, *Suspenser* supports a more simplistic reader model than they desired, and only allows for output stories to achieve overall levels of high or low suspense. Additionally, the system does not allow for real-time user input, limiting its ability to be used in highly interactive games.

In a successful attempt at marrying story manipulation with player interactions, we look to Stern and Mataes's *Facade* [10]. Facade uses two explicitly tracked values of tension and affinity. In this system, tension is a controlled story value appearing in the effect slots of story beats. Affinity is the degree to which the two non-playable characters (NPCs), consider the player to be siding with them. It is not possible to have positive affinity with both NPCs at the same time, making any attempt to do so futile.

In its story-beat selection, Facade's drama management system attempts to conform to a desired Aristotelian drama arc. The system scores beats by taking into account the player's action history, and selecting the valid beat that best matches the desired tension curve [11]. These beats have predetermined effects, and are not dynamically generated, so the system must pick the beat that will result in the closest tension outcome. This system's approach allows for replayability, as players can have many different experiences based on their approach to the story.

A surprisingly fresh and forward thinking approach in the commercial sector was taken with Ubisoft's AAA first-person shooter, *Far Cry 2* [12]. Designers for this game were interested in gamers being able to shape the story through their actions, as well as having actions carry meaning and weight. This end was achieved by having generalized pieces of pre-authored story and behavior dynamically sewn together depending on how the player interacts with the game. Content presentation was determined by the player's "infamy," a metric that is calculated based on the gamer's actions [13]. The player's main method of interaction is through their weapon, with the use (or lack of use) determining their "infamy." This correspondence neatly ties the formal affordances of the game with the mechanical affordances. In their open world shooter, the developers strove for gameplay shaping the story, rather than the two being incidental.

Far Cry 2's system is described as having a "dynamic story architecture" that takes large banks of content and divides them up into small pieces, called "micronarratives." The system then delivers those pieces in a way that reflects the current state of the game world as well as by "infamy" [14]. This process was designed to be invisible to the player, which unfortunately made its implementation mostly overlooked by critics and the masses. Since missions were 40-60 minute chunks of gameplay, the resulting dynamic decisions made by the system about the story and world seemed like happy coincidences to players, rather than results of their actions. Had gameplay sections been smaller, it is possible that this novel system may have evinced more impact, and/or been more apparent to those playing.

Another piece of related work that takes an inventive approach to dynamic tension is Valve's *Left 4 Dead 1* and *2* [15]. These two games utilize a system called the *Director*, which determines gameplay elements based on player actions and experience. To achieve this, there are multiple artificial intelligence systems at work within this game [16]. The protagonists and enemies have "reactive path following," "intentional actions," and an awareness of all events that have occurred in the world, all in order to provide appropriate feedback and responses to player progression through a level. There is an "adaptive dramatic pacing" element which is designed to give peaks and valleys of tension in gameplay by altering enemy population size and placement. The *Director* uses procedural placing of enemies and loot by analyzing metrics, and reasoning based on player performance and position. This system generates a dramatic game pacing that ostensibly provides a different gameplay experience each and every time a level is played, promoting replayability on a large scale.

The *Director* mostly succeeds in providing a dynamic experience that changes based on player or team performance, rooted in mechanical affordances. The problem with it is that the same story is always told, and the player has no agency in how, when, and where the events of the game will take place. In a missed opportunity, the *Left 4 Dead* series always presents the same sections of the environment and story every playthrough, leaving the *Director* only able to manipulate gameplay aspects.

3. The Illuminati Tension Manager

Our system, which we've named *Illuminati*, serves as an engine-like tool for dynamically managing the tension of a client game. It functions by providing the game with suggestions for player choices and game state modifications that conform to a specified arc of tension over time. In order to utilize the system, the client game must comply with *Illuminati's* supported game model. For this prototypical version, *Illuminati* can only provide suggestions for a game based on a single success-failure continuum, along which the player making choices to proceed toward success. We will call this the *goal axis*, and the player's current position on this axis, the *goal state.*

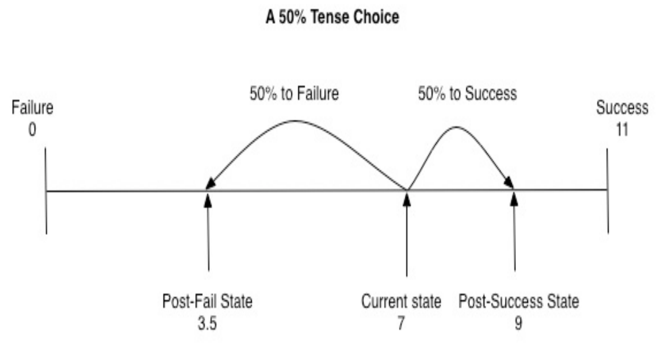
At the start of the game, *Illuminati* must be initialized with the desired tension arc, the arc duration, and the goal axis values for success and failure. The tension arc provides the manager with a series of percentage-based tension parameters that can apply to any specified game duration. At run-time, the client-game can then query *Illuminati* to receive choice and state-modification suggestions. It is the client's responsibility to update *Illuminati* with the current goal state and time. The system will then interpolate to find the desired tension level, making it the client's responsibility to implement *Illuminati's* determinations.

The Tension Model

In order to make choice and modification determinations, *Illuminati* employs a mathematical model of moment-to-moment tension. Its model ties together notions of goal-based choice importance and outcome probability. The system maps the notion of importance directly to tension. In this model, a choice's importance is dictated by how close it can bring the player to an end-state on the goal axis. A choice that would result in no movement on this axis would be considered to have no importance, whereas a choice that could result in reaching an end-state would be maximally important.

According to this model, a choice's tension is equal to its importance. Therefore, a set of player choices that would potentially move the state 50% towards an end-state would present 50% tension. In designing our system, we dubbed this goal-state movement percentage the *impact* of the choice.

It is important to mention that this impact percent on the goal axis will be represented differently depending on the actual game state. For example, consider a scenario in which the current state on the axis is a 7, where 0 is marked as failure, and 11 is marked as success. A 50% tense choice might therefore move the state 3.5 units towards failure, while it would only move the state 2 units towards success. In this case, both of these outcomes move the state the same impact percentage towards an end, but are represented differently in the game-space. In this scenario, the choice presented would have an *failure impact* of 3.5, whereas the *success impact* would be 2.



4. PRESERVING PLAYER AGENCY

A. Choice Variation

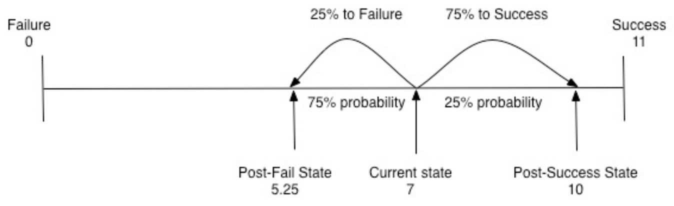
If the impact percentage was the sole factor in importance determination, a given tension level determination would allow only choice options with an impact percent matching the desired tension level. Although this one option might match the desired tension level, limiting player choice in this way would drastically restrict a player's sense of agency. We felt strongly that client games should be able to offer multiple choices at any given tension level, so we incorporated the notions of modified impact percentages and probabilistic tension evaluation.

With these additions to the model, *Illuminati* can suggest choice options with variable impact percentages. The model compensates for the delta between these modified impacts and the desired tension percent by altering the outcome probabilities. The perceived tension of the choice will remain constant if the probability of an outcome is decreased as its impact is increased beyond the desired tension percent. In a calculating an impact-modified choice, *Illuminati* will increase either the success or failure impact percentage, and will correspondingly decrease the probability of that outcome. The probability decrease will be proportionate to the impact percent delta.

In the example above, the system would have assigned a probability of 50% to each outcome, as they both match the desired tension level. We refer to this as the base probability. A modified choice for this scenario might present an additional 25% success impact percentage, an increase of 50%. This would result in an increased success impact of 3. In response, the system would decrease the probability of this outcome by 50% of the base probability, for a final probability of 25%.

To maintain balanced tension, the same 25% would increase the failure probability for this option, and the failure impact would be decreased by 50% proportionately. The client game can then implement this probability space through pure computation, or by giving the player an execution task with a difficulty level that matches the specification. By necessity, this balance limits the system from modifying impact percentages by more than 100%. It also limits impact percentages to a range of 0 to 100%.

A 50% Tense Choice with 50% Modification (pM)



The Balanced Choice Tension Formula

$$(.5 + .5 * pM) * (tension - pM * tension) == (.5 - .5 * pM) * (tension + pM * tension)$$

fail probability * fail impact % == success probability * success impact %

In these instances, the extra impact percent could allow players to reach an end state before experiencing the full tension arc duration. This inspired us to introduce the concept of author-allowed progress throttling, which we will discuss in the next section.

C. Access to End-States

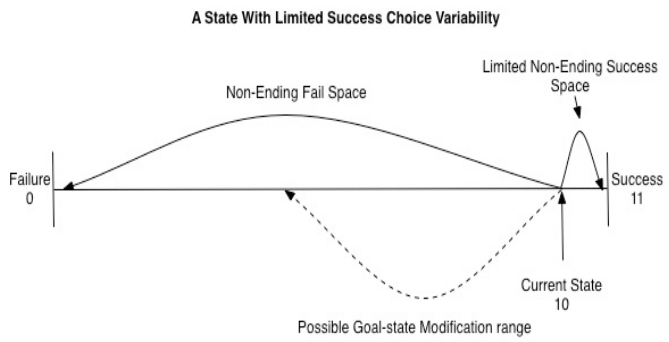
For players to have a sense of agency as they try to accomplish their goals, it is necessary for them to feel that their choices truly impact their progress towards success or failure. Unfortunately, it is challenging to guarantee a desired tension arc experience while maximizing player agency. In the initial, unmodified tension model, players would be unable to reach an end-state until the desired tension arc reached 100%. In the modified probability model, it is possible for an end-state to be reached prematurely.

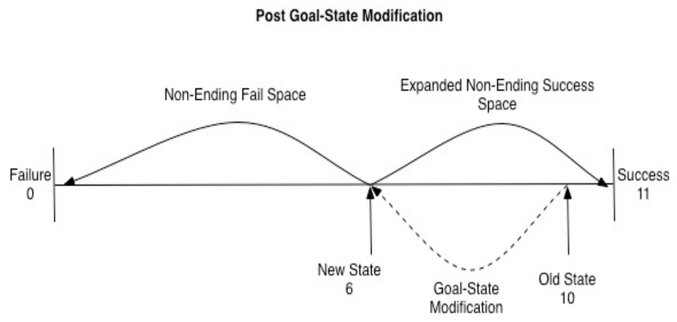
To address this, *Illuminati* allows client games to specify a minimum percent time before end-states will be accessible.

For example, a client can specify that the player must experience 65% of the desired tension arc before being able to succeed or fail in their goal. Once this point in the arc duration is reached, the system will allow extra impact percentages that cover the entire goal axis. The probability of these modified choices will still be balanced to reflect the current desired tension, resulting in significantly amplified choices occupying correspondingly less of the probability space. As mentioned above, extra impact percentages cannot exceed the desired tension percent, so end-states can only be reached at tension levels of 50% and above.

As choice variability is random and dynamic, the system will not enforce game-ending choices until 100% tension has been reached. *Illuminati* therefore provides an override system which allows the client game to request a set of climactic choices. These choices will push the game state to failure or success, ending the experience.

If the minimum percent end-state time has not yet been reached, *Illuminati* will limit the overall impact of choices to prevent end-states. If the player has advanced too far to one end of the goal axis, this type of impact limiting will restrict the system's ability to provide choice variation. In these cases, the system will suggest a modification to the goal-state, which will push the player away from the edge of the goal axis. At most, this goal-state modification will push the state halfway to the other side of the axis. *Illuminati* will then present choices that reflect the impacted goal state. Again, it will be the client game's responsibility to enforce the state change.


A State With Limited Success Choice Variability
Non-Ending Fail Space
Limited Non-Ending Success Space
Failure 0
Success 11
Current State 10
Possible Goal-state Modification range



D. Client game: *Jumper*

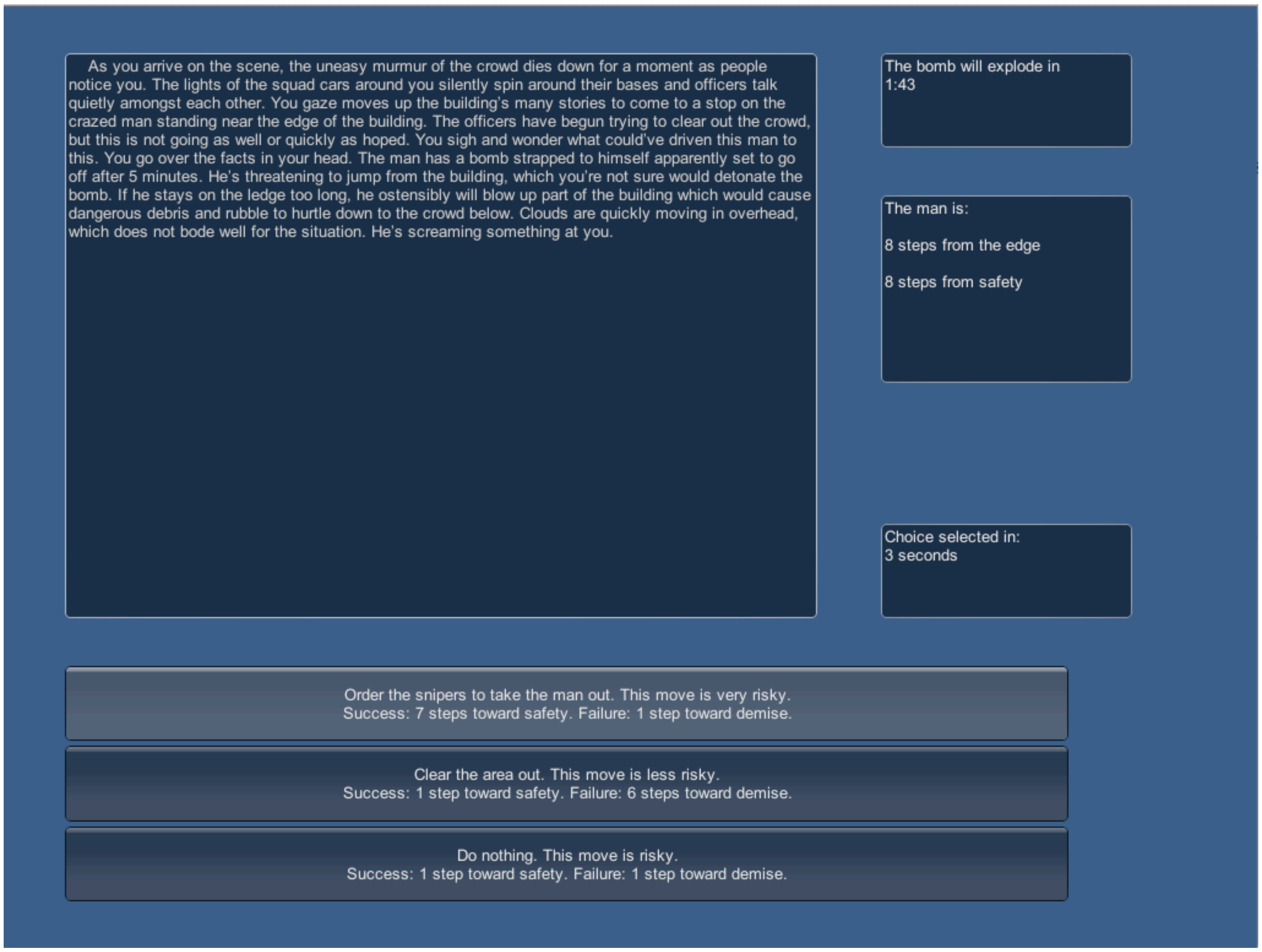
The underlying *Illuminati* tension manager was tested with *Jumper,* a simple game which was designed to highlight the engine's current ability and potential. The game is a simple text-based interactive story, where the player interacts by deciding between three choices within a twenty second time frame per turn. The entire game is relegated to 2 minute playthroughs to keep any single game session from becoming stale, and also allow for replayability. For this game, we chose a highly tense situation of a man threatening to jump off of a building, with the caveat that he also has a bomb strapped to his chest.

Players are presented with scene text, which describes what is happening in the game. They are told how much time the bomb has left on its timer, how many steps the man is from demise as wells as safety, and also how much time left is left in a turn. The game presents three buttons, each with a description of what the protagonist (the player) will attempt, along with how many steps the jumper will move from the edge of the building if successful, how many steps toward the edge if unsuccessful, and how risky the move is. The user selects a button, and once the choice timer runs out, they are presented with the results of their action via scene description and the jumper's new location. The resulting scene text is tied to a player's success or failure, though the player choices are procedurally determined based on how "tense" the current game is. Random events can also be inserted into the scene text if the player begins to succeed or failure too early in gameplay.

Three outcomes are possible in *Jumper*'s current iteration: players can save the jumper before the bomb goes off, the jumper can leap from the building before the bomb's timer has elapsed, or a stalemate where the jumper neither jumps nor comes off the ledge, and instead the bomb's timer expires ending the game. Players are unable to win the game earlier than 50 seconds having passed, but beyond that threshold, users have the ability to win.

Although *Jumper* demonstrates a good deal of *Illuminati's* capabilities, it also falls short in a few significant areas. Because of the text-based nature of the game, and limited development time, it relies heavily on pre-scripted text assets. This results in gameplay experiences that aren't as representationally varied as the underlying dynamic impact and challenge structure of the choice options.

Additionally, *Jumper's* goal axis is tied directly to the man's number of steps from the edge (failure) and



As you arrive on the scene, the uneasy murmur of the crowd dies down for a moment as people notice you. The lights of the squad cars around you silently spin around their bases and officers talk quietly amongst each other. You gaze moves up the building's many stories to come to a stop on the crazed man standing near the edge of the building. The officers have begun trying to clear out the crowd, but this is not going as well or quickly as hoped. You sigh and wonder what could've driven this man to this. You go over the facts in your head. The man has a bomb strapped to himself apparently set to go off after 5 minutes. He's threatening to jump from the building, which you're not sure would detonate the bomb. If he stays on the ledge too long, he ostensibly will blow up part of the building which would cause dangerous debris and rubble to hurtle down to the crowd below. Clouds are quickly moving in overhead, which does not bode well for the situation. He's screaming something at you.

The bomb will explode in
1:43

The man is:

8 steps from the edge

8 steps from safety

Choice selected in:
3 seconds

Order the snipers to take the man out. This move is very risky.
Success: 7 steps toward safety. Failure: 1 step toward demise.

Clear the area out. This move is less risky.
Success: 1 step toward safety. Failure: 6 steps toward demise.

Do nothing. This move is risky.
Success: 1 step toward safety. Failure: 1 step toward demise.

safety (success). In order to succinctly represent the goal state, the game employs an integer-based representation of the goal axis, instead of a floating-point continuum. This limits the choice options from complying fully with *Illuminati's* suggestions, as all values are rounded to conform to the game's discrete state representation.

The system's depth would be better represented by a client game with a continuous, procedural representation of the goal axis and goal state, which could be represented easily with a graphical implementation. The current version of *Jumper* does not allow players to actually execute any of their choices beyond deciding which button to press. The game then rolls a dice (bounded random number generator) to see if the choice succeeded. Unfortunately, this misses the opportunity to allow the procedurally determined challenge to impact users' play outside of renegotiating which choices to pick. By allowing players to execute on this probability, they would better be able to internalize the probability space, instead of attempting to reason as to how the computer will evaluate success and failure. A simple idea that could be implemented for a near-future iteration of *Jumper,* would be some sort of meter that grows larger or smaller as determined by the tension manager. This would make the changes that Illuminati is making more apparent, as well as aptly giving the player the opportunity to interact with this newly determined challenge.

4. Successes of Illuminati

As evidenced by play-tests with *Jumper*, Illuminati succeeds at providing choice options which adapt to the specified goal state. This allows the client game to present options that change relative to prior player actions, promoting greatly mixed mechanical variability. As stated above, it is the responsibility of the client game to represent this mechanical variability, which is an area where *Jumper* is unable to demonstrate the full potential of the system.

Additionally, *Illuminati* succeeds at presenting choices which reflect the desired tension specified. As a play session of *Jumper* progresses, the player will be presented with choices that impact the goal state directly corresponding to the specified tension arc. In this regard, *Illuminati* functions as a general tension management engine in the way we intended.

5. Future Iterations

This prototype of the *Illuminati* system is limited in several ways, which we hope to address in future iterations. In its current form, the system is only able to reason over a single goal axis. The choice options are all directly tied to this axis, which limits the overall expressiveness of the system. A deeper game experience would likely offer multiple simultaneous goals, each with multiple goal-state axes.

For example, Bethesda Softworks' *Dishonored* [17], allows players to pursue multiple objectives simultaneously, each allowing multiple avenues for success or failure. At any time, a given player might be attempting to pursue assassination, collection, and navigation goals via varied means that include stealth and combat. The intersection of these goal axes all combine to dictate the player's tension level, based on the player's valuation of these goals. In order for *Illuminati* to support this level of client game complexity, we will need to expand its tension model accordingly.

Another significant limitation of *Illuminati* is its restriction to suggesting choice options and goal-state modifications for tension modeling. In our future work, we plan to expand the system's tension model to include other forms of tension, such as representational and temporal tensions. Representational tension modeling will include modifications to the sound and appearance of game content, while temporal tension modeling will affect the flow of game-time related aspects. Allowing *Illuminati* to regulate tension through these alternate avenues would provide greater gameplay variety and more significant variation between replays. Additionally, this would provide the system with alternate means for suggesting high-tension scenarios without allowing player access to end states.

6. Conclusion

This paper has presented the first iteration of the *Illuminati* tension manager and its application to a real-time game that runs in a discrete, text based world. Other modern takes on procedural modification of game story and environment that informed our system have been examined in our discussion. The client game *Jumper* was built to test the initial functionality of the system and has successfully showcased some of *Illuminati's* abilities, as well as some future places for improvement. While the discrete presentation of *Jumper's* world shows the changes that *Illuminati* makes, it falls somewhat short of allowing the player to internalize and interact with the constantly fluctuating challenge and impact. Our tension manager is still in its nascent stages, and many improvements are forthcoming.

As a first foray into dynamic tension management, we believe that our system shows exciting promise. Our hope is to create a general tool that will be usable by the majority of games in the future. We aspire to promote games having more dynamic, procedural content that allows authors to provide a desired story, while at the same time gameplay that allows for maximum replayability and flexibility. This application can apply to the multitude of game genres, ranging form action games, to puzzlers, to adventure games. By providing a desired tension arc for a game, the engine can provide a corresponding experience: from a puzzle game changing its difficulty on the fly, to an adventure game changing the location and order of events based on player history.

While we have lofty goals for this engine, our next step is to apply *Illuminati* to a real-time game that offers a continuous interaction with the game world. We imagine that this would be done with a graphically-based game, as opposed to *Jumper's* text-based gameplay. This seminal iteration provides a good base to build upon, and we believe our desires can be achieved. As we are large proponents of the power of the video game medium, the inclusion of dynamic tension management seems like an important element for the advancement of this art form.

7. Acknowledgements

The authors of this paper would like to thank Michael Mateas and Noah Wardrip-Fruin for their continuous support and encouragement in our endeavors, as well as Sejin Park and Amanda Triplett for their patience and playtesting.

8. References

[1] Quantic Dream. *Heavy Rain*. Sony Computer Entertainment, 2010. Playstation 3.

[2] Telltale Games. *The Walking Dead*. Telltale Games, 2012. Xbox 360, Playstation 3, PC, iOS.

[3] Epic Games. *Gears of War 1, 2, 3*. Microsoft Studios, 2006, 2008, 2011. Xbox 360.

[4] Bioware. *Mass Effect 1, 2, 3*. Electronic Arts, 2007, 2010, 2012. Xbox 360, Playstation 3, PC.

[5] Rockstar Games. *Grand Theft Auto IV*. Take-Two Interactive,. 2008. Xbox 360, Playstation 3, PC.

[6] Treyarch. *Call of Duty: Black Ops 2*. Acitivsion, 2012. Xbox 360, Playstation 3, Wii U, PC.

[7] People Can Fly. *Gears of War: Judgement*. Epic Games, 2013. Xbox 360.

[8] M. Mateas, C. Lewis, S. Chen, and A. Sulliven. "EMPath: Applying Drama Management to a Quest-Based Game," 2009.

[9] Y. Cheong, M. Young. "A Computational Model of Narrative Generation for Suspense," in AAAI 2006 Computational Aesthetic Workshop. July 16, 2006.

[10] M. Mateas "Interactive Drama, Art and Artificial Intelligence." Ph.D. dissertation, December 2002.

[11] M. Mateas, A. Stern. Facade: An Experiment in Building a Fully-Realized Interactive Drama," 2003.

[12] Ubisoft Montreal. *Far Cry 2*. Ubisoft, 2010. Xbox 360, Playstation 3, PC.

[13] C. Remo, B. Sheffield (July 18, 2008). Gamasutra.com: Redefining Game Narrative: Ubisoft's Patrick Redding On Far Cry 2 [Online]. Available: http://www.gamasutra.com/view/feature/3727/redefining_game_narrative_.php

[14] C. Onyett. (February 22, 2008) GDC 2008: Far Cry 2's Dynamic Story [Online]. Available: http://www.ign.com/articles/2008/02/22/gdc-2008-far-cry-2s-dynamic-story

[15] Valve Corporation, Turtle Rock Studioes. *Left 4 Dead 1, 2.* Valve Corporation, 2008, 2009. Xbox 360, PC.

[16] M. Booth, Valve. "The AI Systems of Left 4 Dead," in Artificial Intelligence and Interactive Digital Entertainment Conference at Stanford. 2009. [Online] Available: http://www.valvesoftware.com/publications/2009/

[17] Arkane Studios. *Dishonored.* Bethesda Softworks, 2012. Xbox 360, Playstation 3, PC.