Camera Tracker: A Reactive 3D Camera System

Nicholas Poirier
Ivan Allen College, Georgia Institute of Technology
1111 Mecaslin St. NW
Atlanta, GA, 30318
1-(302)-562-7934
Nick.Poirier@gatech.edu

ABSTRACT
In this paper, I describe the research and work that went into the creation of the Camera Tracker system, with a look at previous related works in both popular interactive media and academic research.

Keywords
Platforming, camera

1. INTRODUCTION
In any virtual, three-dimensional environment, our relationship to that environment is framed significantly by our perspective into it. This perspective is mediated by an in-game "camera". How a camera goes about framing a virtual environment, the information it chooses to show or exclude, and the efficiency and clarity with which it presents that information can have a dramatic impact on the perceived difficulty of a game and on how the player understands and interacts with that environment.

Technically, a camera consists of computer code that dictates how the game world should be rendered onto the screen, how the player and world should be framed, and how this framing will change dynamically over time. In virtual game environments, there are a few standard conventions for camera systems which revolve around the player's perspective relative to an in-game avatar. Nitsche defines the four main camera positions in video games as, "the first person POV, following cameras(and related views such as over-the-shoulder cameras), overhead views (and related views such as isometric style), and predefined third person POVs (fixed or moving)."[1] The choice between having a first-person (through the avatar's eyes) or third-person (from outside the avatar) camera is often determined by which style of gameplay the camera complements. Games which revolve around shooting and/or fast reflexes often opt for a first person camera while games that need players to have a greater environmental awareness (for example, real-time strategy games) choose a third-person perspective which can offer more immediate information about a surrounding environment. There are rarely fixed camera distinctions, but camera styles tend to fall along genre differentiations based on a broad tradition.

The genre of games which this project seeks to address is that of the third-person platformer. Platforming games are play environments in which a player is required to skillfully move a character from one point in an environment to the next. The manner in which they accomplish this is dictated by the game's design but often features some common elements of a character in motion including running, jumping, climbing, and balancing. Additionally, players are often confronted with obstacles which must be overcome in order for them to move on in the environment. These obstacles can include particularly challenging jumping sections, combat areas, and complicated environmental puzzles. In general, three dimensional platforming games are notorious for having troublesome cameras at times. The move from 2D to 3D platforming environments necessitated a more dynamic camera, making the in-game camera work much more complicated. The standard approach to camera in a 3D platformer is an over-the-shoulder camera which is either rotated around the avatar by the player or automatically controlled by the game.

Figure 1: Standard 3rd Person Camera [2]

While this approach to camera does a sufficient job of allowing players to accomplish the standard tasks required of them in the 3D platforming environment, it can hinder a player's progress when it runs up against the challenges of manipulating a camera inside a 3D environment.

2. CHALLENGES OF THE 3D CAMERA
Since virtual cameras dictate the means by which environmental information is delivered to a player, significant decisions need to be made about how an in-game camera frames an environment relative to the player's avatar. Framing a 3D virtual environment with the goal of helping a player navigate through it brings a specific set of challenges to light.

2.1 Framing and Visual Information
One of the challenges of using a camera in a third person game is in positioning the camera in such a way as to provide the player with a sufficient amount of visual information. Players need this information in order to maintain good situational and environmental awareness. However, important objects can sometimes be excluded from the frame at critical moments or never enter the frame at all. In order to give the player a full picture of their surrounding environment the camera has to frame the relevant elements in the environment.

2.2 Depth Perception

Especially in the context of a third person platforming game, depth perception in virtual worlds provides another layer of challenge to the player. Accurately judging the distance of a platform from an avatar is a critical task when attempting a jump. If the player perceives the platform to be farther away than it actually is, they might overshoot the jump. Visual environmental cues, which are transmitted by the camera, often aid in depth perception in virtual environments just as they do in real life.[3] These environmental cues include perspective, relative, motion parallax, light and shading, and blurring or depth of field. However, in standard 3D environments on a two-dimensional display, standard binocular depth cues like steropsis are not present. Camera perspective plays a large role in how the player judges the distance between a current platform and the next.

2.3 Occlusion

Another challenge presented to the virtual 3D camera is inn preventing objects from getting in front of, or occluding, other important objects in the environment. Problems of occlusion can obscure the player's view of their in-game character or hide important environmental and situational information from the player. Even the model of the avatar itself can obscure important information like the avatar's distance from an edge. These aspects of framing and camera position can add challenge to the game in ways that are frustrating and sometimes unavoidable.

3. RELATED WORK

3.1 Games

A number of modern in-game camera systems have developed their own approaches to camera control and framing. Each of the techniques described below is an attempt at solving one of the aforementioned challenges of the virtual 3D camera.

Figure 2: Hold To Focus Prompt [4]

"Hold To Focus"

The first common approach to turning the camera into a tool for the player is the "Hold To Focus" cue. This approach is commonly used to highlight an important event occurring near the player, often as part of a scripted animation or sequence. In practice, when the player triggers this sequence, an on-screen prompt urges the player to hold a certain button. Once held, that button triggers either a cutaway camera sequence to show the player some nearby, but off-screen, event or it forces the camera to turn and focus on an important environmental object without any cutaway editing. Examples of this practice can be found in *L.A. Noire* [5], *Bulletstorm* [4], and many other games. The "Hold To Focus" cue is designed to provide the player with important visual information using a scripted event to temporarily shift their focus in the world.

Psychonauts "Smart Camera"

One particular attempt at using the camera to provide environmental information and to guide the player is the optional "Smart Camera" in Tim Schafer's *Psychonauts* [6]. The "Smart Camera" is designed to enable "hints that direct the camera towards specific targets" [7]. Though the in-game effects are subtle, it seems like the "Smart Camera" often points towards the player's next platform or objective. This system can be a bit difficult to work with at times as it conflicts with the standard manual camera control present throughout the rest of the game.

Context Sensitive Framing

Another common practice, found in games like *Prince of Persia: The Sands of Time* [8], *Assassin's Creed* [9], and *Batman: Arkham City* [10] is the context sensitive camera. This camera reacts to the actions of a player character to provide important, context sensitive, information and/or to create a dramatic effect. This style of camera becomes apparent in situations like when the player is hanging from, or shimmying across, a ledge. In these instances, the context sensitive camera might position itself above the player, orthogonal to the wall, and pointing downwards at the player. This camera translate left-to-right joystick input into motion along the ledge, and creates a sense of vertigo or height by showing the ground or abyss threatening the player should they fall from the ledge.

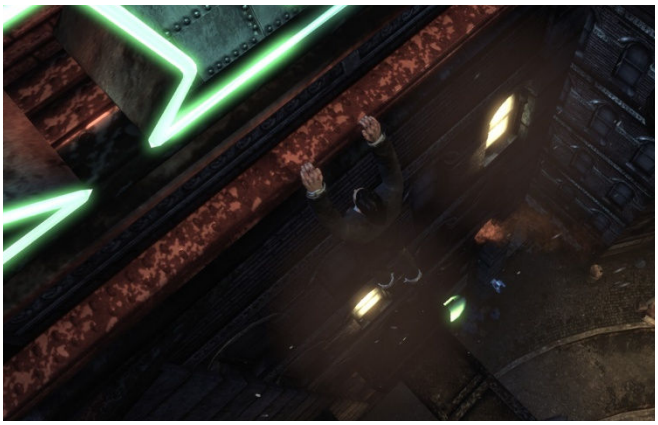

Figure 3: Context Sensitive Ledge Camera [10]

3.2 Research

Academic research in the field of virtual cameras helped inform the ways in which this project could approach a system for dynamic camera control.

One notable piece of early work in dynamic three-dimensional framing is William Bares' *ConstraintCam*. *ConstraintCam* is described by Bares as "a real-time camera visualization interface for dynamic 3D worlds"[11]. The *ConstraintCam* system tracks unpredictable AI agents as they move around in a small virtual setting. Then, based on input from a user, the system tries to frame specific agents. It accomplishes this task by determining some ideal positions for a camera based on specific constraints. These constraints are determined through the use of general film theory and some strict technical guidelines like "avoid occlusion". *ConstraintCam* also generates multi-shot viewports if all tracked actors cannot be viewed in the same frame. This system uses dynamic framing to make sure that the user can see all desired information, in this case the location and actions of selected agents, at all times.

A similar constraint-based system is found in Arnav Jhala's *Maze-Ball*. *Maze-Ball*, which was implemented in Unity 3D, explored the interplay between camera viewport and challenge. Jhala constructed a variety of profiles for an in-game camera based on factors like distance from target, height, and frame coherence. These various profiles then are chosen as viewpoints for a player attempting to navigate a ball through a maze. Jhala theorized that if more information was provided to the player, perceived difficulty would decrease. After extensive user testing, he found that, "More information about the maze of the game leads to decrease of the challenge value. Similarly more information about the enemy leads to a decrease in reported challenge."[12] These findings support the notion that camera perspective can have a large impact on perceived game difficulty.

Work by Kneafsy and McCabe[13] explores the possibilities afforded by a cinematic perspective to camera control inside a game world. They plan to implement a camera control system for 3D worlds which is informed by cinematography. One goal of this approach is to evoke an emotional response from the user through dynamic camera placement. In *Camera Control through Cinematography for Virtual Environments*, they present a number of methods for selecting cameras within a virtual world. These methods often use a selection of film idioms for framing a character in third person. An overarching intelligent system then uses a decision making process, informed by in-game events, to determine which camera profile to use at any time.

All of these virtual camera solutions experiment with novel ways of framing a virtual environment. These attempts at virtual camera are designed either for dramatic effect, or to present specific visual information to the player at the right time. The game approaches can be analyzed, re-implemented, and studied for their ability to help players interact with the virtual world. Particularly relevant here is Jhala's work because it establishes an important link between camera framing and difficulty. It becomes apparent that a system can manipulate the camera adjust the level of difficulty a player is having with the game.

4. APPROACH

4.1 Theory

As mentioned, the above camera solutions alter their framing of virtual environments to present narrative or gameplay-related information to the player or for dramatic effect. However, though they acknowledge the interplay between camera and difficulty, none of the solutions (except for Jhala) alter themselves based on how the player is performing in the game. This project proposes a dynamic camera system which reacts to player performance in real time and chooses a camera solution best suited to aid the player at their current in-game goals. For the purposes of this project, the implemented system will focus on aiding environmental awareness and navigation as well as aiding technical accomplishments like successfully completing a jump. Furthermore, this system will learn from previous player performances to choose the ideal camera implementation for the current play through.

4.2 Technical

Without the ability to sufficiently alter the camera system of an existing platforming game, I decided to implement my own short 3D platformer in which to set up the camera system. For this task, I chose to use the Unity3D game engine. The game is composed of three levels, themselves made up of a few discrete jumping sections, of increasing difficulty. Underlying game logic was programmed in C# for Unity 3D. Additionally, a series of XML files are used to store the data about player failure and success. The game is designed to be played with an Xbox 360 controller, because of the fidelity of motion allowed by a controller's joystick, but it works with keyboard too.

4.3 Level Design

Each of the levels in the game environment was designed to test a variety of scenarios or situations around jumping from one platform to another. It was important to test the camera system's usefulness in a number of different styles of challenge including tests of timing, jump accuracy, depth perception, and navigation. The elements listed below are present in one or more levels as a means for presenting one or more of these tests to the player.

Long Jumps

Long jumps require the player to jump twice (double jump) in order to successfully land on the next platform. This is a common feature in many platforming games and was a necessary challenge for testing if a player could accurately judge his or her distance from their desired point of landing. I expected depth perception to be challenged here as this type of jump requires players to try and gauge an approximate distance, into the screen, for the next platform.

Moving Platforms

Moving platforms serve to test a number of player skills. These include distance-from-player judgements, distance over time approximation, and an ability to accurately track a moving object in space. Additionally, moving platforms were introduced which move in dramatically different directions. Some feature a more vertical path of motion while others move horizontally through space.

Vertical "Steps"

In a few cases, the player is required to jump up a stack of "steps" or platforms arrange above one another. The player must be able to accurately predict a jump arc in order to know at what point in a jump their character will be able to reach the next step.

Quick Timing

One section features some sequences of jumps which fall into the abyss if the player lingers on them for too long. These segments require the player to be able to move quickly through the whole sequence in order to succeed. Additionally, some of these sequences feature ninety degree turns, requiring the player to re-orient themselves (and the camera) in order to see and move to the next platform. These segments are more a test of player speed than accuracy.

4.4 Camera Design

For this system, six different cameras were designed. Each is an attempt to solve one of the original three problems. Though only six solutions were implemented due to time constraints, the system itself can work for any number of camera solutions. New concepts can be programmed and implemented into the existing camera controller framework.

Dynamic Framing

This camera moves around the scene in order to frame all relevant objects, typically points along a path, at once. Over the shoulder cameras only offer visual information about what is in front of the player's current facing. From new perspectives, the player should be made more aware of the important elements in their environment, and thus will perform better. This camera positions

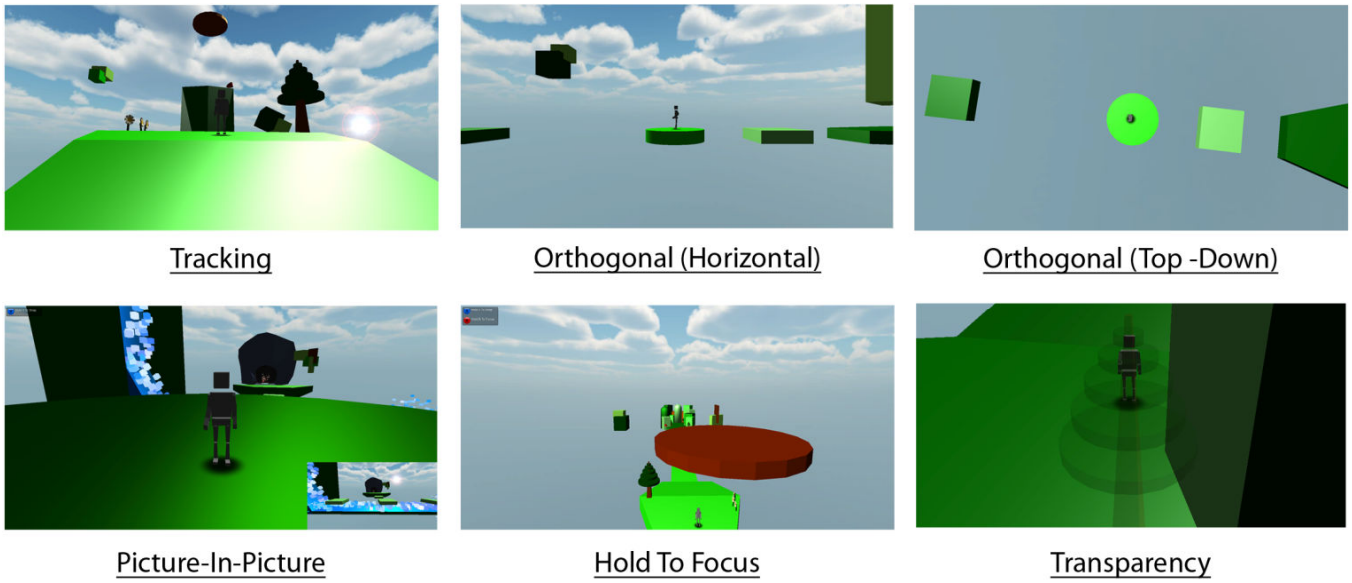

Figure 4: Camera Configurations

itself along a vector between the player's current position and the next point that they should travel to.

Planar/Orthogonal Framing

This technique attempts to reduce problems of depth perception in three directions into a problem of judging two dimensional distance. The camera positions itself directly orthogonal to the path between two jumps. The player can then judge their distance from the next platform by the X or Y on-screen distance instead of the virtual world Z distance of objects on screen. The two implemented versions of this camera position themselves either to the side of the player or above.

Hold To Focus

This is an implementation of the "Hold To Focus" system from other games, but with the ability to focus on the next objective at any time instead of only during pre-scripted events. The player can hold down a button to cause the camera to move its focus towards the next platform in a sequence. When the button is not held down, the camera reverts to a standard over-the-shoulder view.

Picture in Picture

In this mode, there are two cameras. One is the standard over-the-shoulder camera, which positions itself at a fixed viewpoint from behind the player. The other is a camera which focuses on objects in the world, but appears in a different frame on-screen. The second camera always frames the next important platform in a level. While the split attention resulting from another frame being present may be a detriment, the extra information provided can still aid in navigation and environmental/enemy awareness.

Depth of Field

This camera keeps environmental elements in focus and alters the Depth of Field (DoF) on the camera to put the most important element in focus. The potential benefits of this camera are three-fold. First, the camera continues to frame relevant information, giving the player more knowledge about their environment. Second, DoF immediately shifts the player's gaze to one focal point, directing attention. Finally, relative depth blur may provide additional depth perception cues, which, as previously stated. are critical when judging jumping distance in a platformer.

Selective Transparency

This camera, already in place in a number of platforming and third-person games, causes objects in the scene to become transparent when they occlude the player from the camera. This allows the player to still see their avatar while moving about in space for situations where the camera would occlude him behind a piece of level geometry.

5. PROCESS

The first step in implementing the dynamic camera system was to create the game world in which the camera operates. A ginel dynamic camera was created, which assumed a standard over-the-shoulder, third-person perspective. Then, a node system was added to provide the camera system with information about where platforms were located. Finally, a system was put in place to track a player's in-game performance, and which camera they are using at the time, and store that information out into an XML file. This XML data could then be read by the camera system to help it determine what the ideal camera for each node might be, based on the current player's performance and the history of gameplay performance across each player.

5.1 Camera and Node System

The decision making of the camera system is guided by an over-arching camera manager and a sequence of nodes throughout each level. These nodes indicate for the system where important points along a level are, and keep track of which cameras apply to their particular node. This was designed so that some sections could be set up with one particular set of cameras while other sections could use a different set. The manager keeps track of the player's position in space, and specifically their distance from each node. By determining which node is nearest, the manager can decide, based on that node's list of possible cameras, which camera to show. Though the camera list could be automatically generated, What cameras work for each node was left up to the level designer. Every node needs at least one camera, but the more cameras there are, the more information a camera designer might get from the tracking system about which cameras work in a particular section and which do not.

5.2 Tracking the Player

While determining which camera to use at which point, the system is also tracking information about a player's successes and failures. When a player successfully completes a certain section of the level using a camera, the manager makes note and is then

more likely to use that camera in the future. Additionally, player deaths or failures in a certain section cause their current camera to be less likely to occur at that time in future playthroughs.

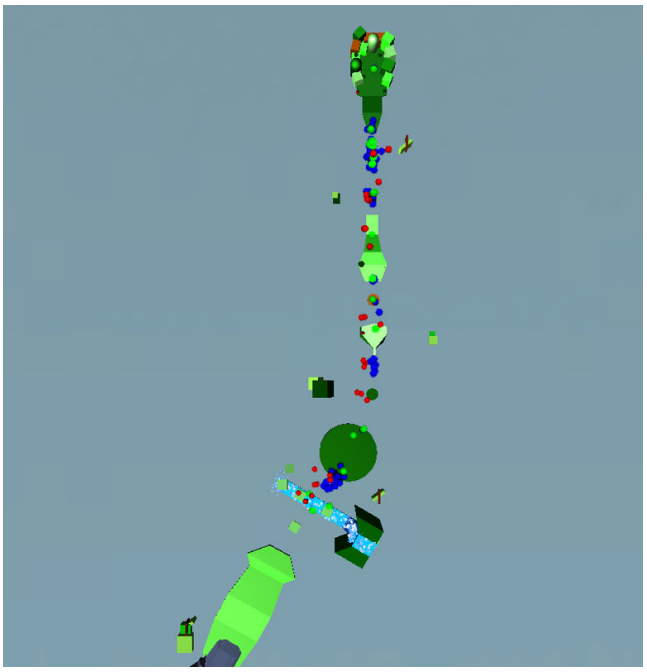

Figure 5: Heatmap of Player Failure and Success

Furthermore, upon player death, the manager changes the active camera for that node to the next most successful camera in its list. Players are also given the option to skip over their current camera if its viewpoint is not satisfactory for them. These skips are also recorded, and used to decrease the likelihood of that camera showing up at that node in the future. This information is collected across multiple players or playthroughs of a level. Over time, the system will move towards more optimal, in terms of aggregate player success and preference, camera choices for each node. The data gathered about each node by the manager is collected and added into an XML database at the end of each level. At the start of each level, the data for that level is loaded in to the system and applied to each node, allowing the manager to easily use all aggregate data collected across all players when determining which camera to feature on each node.

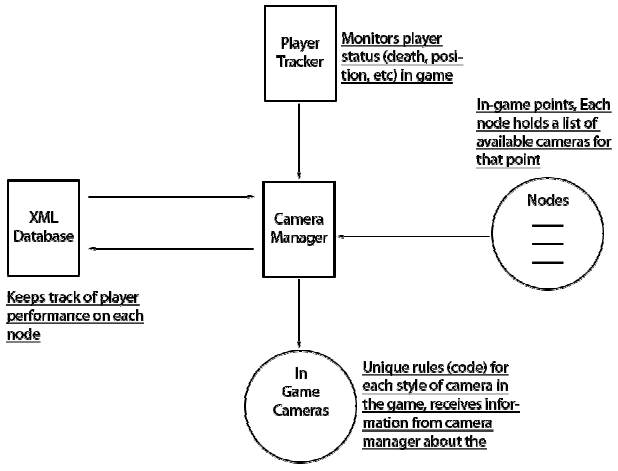

Figure 6: Camera Tracker Technical Diagram

6. RESULTS

This system was tested by eight different players with varied results. Many still felt that the camera was fighting their progress through a level at times. Some users requested a manual camera, a solution which might work well for players who want total control over their perspective. Additionally, players disliked when a camera perspective changed dramatically while they were navigating a level. Framing which was too dynamic made a player's motion relative to the camera too confusing. Better camera solutions will require more static framing and careful motion between camera positions.

The results of the camera implementations are somewhat inconclusive in regards to which camera is "ideal" for certain situations. Players avoided the orthogonal cameras outright, suggesting a need for additional work on those implementations. Though players may have been able to estimate the on-screen distance between two objects, they had trouble determining their position in the third dimension ('into' the screen) when a truly orthogonal camera perspective was used. This made successfully jumping between obstacles especially difficult in the top down perspective, since players could not tell how far above a platform they were while jumping and would consistently over or under estimate their trajectories.

The most helpful and least "skipped" camera was the simple tracking camera that positions itself behind the player while pointing in the direction they are required to go. Not surprisingly, based on the rest of the feedback, this was one of the more static and reliable framing solutions without a lot of quick camera motion. Players also responded well to the "picture in picture" camera, which was used most often when a player was not sure where to go. Further testing is necessary in order to refine the existing camera implementations, examine their success at aiding the player, and to experiment with other possible solutions.

7. CONCLUSION AND FUTURE WORK

Contextual, rather than location-based, camera selection is one extremely important area to develop further. Instead of forcing a designer to decide what parts of an environment might work well with certain camera solutions, a more developed system should interpret aspects of the environment and respond accordingly. For example, the system could determine when a very long horizontal jump is occurring and select a camera which historically works well for all such jumps. In addition to interpreting the context of a certain part of an environment, the system could choose framing techniques which borrow from traditional cinematic methods and contextually apply those techniques to the camera control. Ideally, this would allow for a more emotionally affective experience while still helpfully framing the environment.

Finally, a more in-depth player tracking system could be developed that records more information about the player. Information regarding how long the player is spending on an area and exactly what actions the player is taking would help the system in deciding if the active camera implementation is helping the player accomplish their current in-game goals.

As games become more complex, and increasingly feature techniques like procedural world generation, a dynamic and multi-purpose camera solution will be necessary for creating an experience that is as compelling as current scripted and on-rails camera solutions. As this paper has argued, this camera should be capable of changing itself to suit each player's preferences and navigational goals. A camera system that monitors player performance and is capable of reacting to it will be necessary for

creating an experience which appeals to both novice and expert gamers and works as a useful tool for play instead of as an obstacle. A reactive camera could open up new possibilities for 3D platforming games and enable new player experiences by presenting virtual worlds from a more dynamic perspective.

8. REFERENCES

[1] Nitsche, M. *Games, Montage, and the First Person Point of View*, Georgia Institute of Technology, 2005

[2] *Super Mario 64*. Nintendo, 1996. Computer software.

[3] Pfautz, Jonathan. *Depth Perception in Computer Graphics*. Thesis. University of Cambridge, 2000. Print.

[4] *Bulletstorm*. Epic Games, 2011.

[5] *L.A. Noire*, Rockstar Games, 2011.

[6] *Psychonauts*, Double Fine Productions, 2005.

[7] Double Fine Productions. "Psychonauts Patch V1.04." *FilePlanet*. IGN, 10 July 2005. Web. 02 May 2012. <http://www.fileplanet.com/155960/150000/fileinfo/Psychonauts-Patch-v1.04>.

[8] *Prince of Persia: The Sands of Time*. Ubisoft, 2003.

[9] *Assassin's Creed*. Ubisoft, 2007.

[10] *Batman: Arkham City*. Rocksteady Studios, 2011.

[11] Bares,W. 2000. A Model for Constraint-Based Camera Planning. Center for Advanced Computer Studies, University of Louisiana at Lafayette.

[12] Jhala, A. 2009. Investigating the Interplay between Camera Viewpoints, Game Information, and Challenge. Center For Computer Games Research IT University Of Copenhagen

[13] Kneafsy, J McCabe, H, 2005. Camera Control through Cinematography for Virtual Environments: A State of the Art Report. School of Informatics and Engineering, Institute of Technology, Blanchardstown, Dublin 15, Ireland.