Implementation of an immersive videogame: Legends of Girona

R. J. García[1], A. Rodríguez[1], J. M. García[1], M. Magdics[12], Philippe Bekaert[3] and M. Sbert[1]
1 Universitat de Girona, Girona, Spain,
2 Budapesti Műszaki és Gazdaságtudományi Egyetem, Budapest, Hungary
3 Universiteit Hasselt, Hasselt, Belgium

ABSTRACT

Legends of Girona is a serious game which can be used to teach local history to students in the province of Girona in Spain. The game is a graphic adventure taking place in both present-day and medieval Girona. We describe in this paper the design of the game, including aspects related to artificial intelligence, advanced computer graphics, exergaming, mobile platforms and immersive devices. A collection of different devices is supported, including input devices such as head trackers and output devices such as cylindrical and spherical domes, immersapods, and 3D helmets. We provide an overview of all the stages required for the configuration of immersive devices (calibration procedure, adaptation and rendering). We also present some directions of future work.

Keywords: Serious game, history, immersive devices, position and orientation tracking, multiplatform, non-photorealistic rendering, 1st and 3rd person video game, systems and software design.

1 INTRODUCTION

Historical games can be used to help students remember the most important aspects of history lessons. Some examples of these games include Time Mesh (Gouveia et al 2012), Stories from the History of Czechoslovakia (Šisler, 2012) or Past/Present (Muzzy Lane Software, 2012). We present here *Legends of Girona*: a serious game aimed at learning the local legends of the city of Girona in Spain (a serious game is a game whose main purpose is not pure entertainment: examples include education, training etc). The game can be used by high school students to ease retention of important happenings and dates of local history. The game is a graphics adventure which provides players with an immersive environment in the relevant period.

The game is a result of collaboration between our Computer Graphics group and the TIC Education group. The storyline and plot was designed by the Education group to maximize the learning experience while maintaining the enjoyability of the game, while the Computer Graphics side used the storyline to design and implement the game.

A second objective of the game is to provide a testbed for the integration of emergent technologies (mobile platforms, immersive environments, position and orientation trackers, gesture recognition software), designed to provide portable, easily reused libraries and resources. Our game pretends to go further than previous approaches by providing a much more advanced

immersion and interaction experience, which we expect will significantly increase retention of the historical facts.

Our paper is structured as follows: the next section provides a historical background of the events described in the game, while section 2 describes the game architecture. The state diagram of the game can be seen in section 2.1; then section 2.2 describes the Artificial Intelligence algorithms programmed. Section 2.3 and 2.4 provide a description of photorealistic and non-photorealistic algorithms used in the game. Section 3 describes the use of immersive display devices and trackers (including Microsoft Kinect), and section 4 shows the adaptation of the game to run on iPhone and iPad devices. Finally, section 5 concludes the paper and section 6 provides directions of future work.

1.1 Historical Background

The game starts in present day Girona to provide a familiar setting to users, and to increase believability and identification with the main game character. After spending some time familiarizing themselves with the city (or recognizing familiar streets in the case of players living in Girona), time travel is used to lead the player to the desired date (in our case, the siege of Girona during 1285). A similar technique has been used in the Assassin's Creed series of games.

We have modeled a realistic walkthrough from the Stone Bridge in Girona to the St. Feliu Church along the Rambla and the Ballesteries streets. This provides a view of two landmark medieval structures, and their situation as extremes of a modern-day touristic, cultural and commercial street.

After the walkthrough, the player is transferred to Girona in the year 1285, during the siege of the city. The siege of Girona in 1285 was part of the Aragonese Crusade (1284-1285), in which the pope declared the crusade against the Aragonese king who had just conquered Sicily (a territory under papal control then). The French armies attacked the Roussillon, and besieged Girona in 1285, which was taken. A further battle saw the French defeat, with further losses due to a dysentery epidemic. The crusade ended in 1291 with the lift of the church ban on the Aragonese king (Strayer, 1953).

With these historical facts as a basis, the Girona legend of the flies of St. Narcis took the following form: the legend states that when the French invaders opened the St. Narcis sepulcher searching for loot, a large quantity of flies exited the sepulcher instead, covering the whole city and bringing illnesses to the French soldiers, which had to retreat in haste (Alberch-Fugueras, 2001).

1.2 Game Description

The game starts in Girona's boulevard, near the tourist's office. In the office, the player is given some initial comments; then he is free to walk along the boulevard. On the other side of the boulevard is the St. Feliu church, where the player is given a puzzle to solve. After solving the puzzle, the player is transferred to 1285.

The player is now outside the city and notices the siege. He must flee from the soldiers. After getting help from some citizens, he manages to enter the city. During the process, he notices that the inhabitants of Girona consider him a hero which will save the city.

Inside the walls, the player should collect some flies under the guidance of a sorceress, and search for the entrance of a labyrinth leading to St. Narcis' sepulcher. The flies are to be put inside the sepulcher, so that when the soldiers open it later, the flies attack them. After the soldiers are defeated, the player is transferred back to the present.

2 GAME ARCHITECTURE

The game was implemented using Unity (Unity 2013), which is a multiplatform game engine running on Microsoft Windows and Apple OSX which can create games for the Web, PC, OSX, IOS, Android, Flash, Xbox, PS3, and Wii platforms. Currently, we are targeting the PC, OSX and IOS platforms for our game.

Figure 1. Game Structure diagram

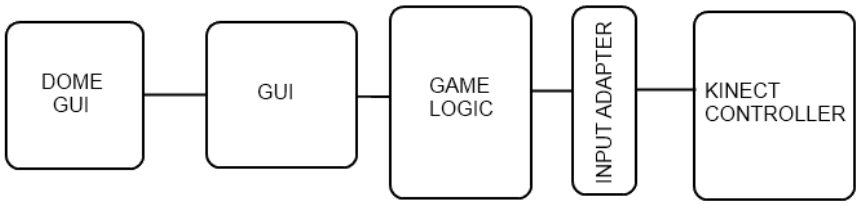


Figure 1 shows how the game class structure is divided in different blocks. The decoupling of Input and Output has been highlighted to show how this architecture can allow the use of non-conventional immersive displays (which traditionally require higher integration in the game engine) and new input devices with little effort.

The scripts and cameras required for managing the different output devices have been centralized in the MainCamera object. A simple interface is used to indicate the current rendering mode and activate the corresponding back-end. The details on the implementation of each back-end can be seen in Section 3. The DomeGUI objects use the current rendering mode to display GUI information in a suitable way.

The different trackers provide information to the input adapter, which is queried by the game logic. The input adapter integrates the different inputs. Position and orientation of the player are also calculated from the different inputs.

The game logic block is composed of the game state and of the actors affected by the game state. There are two types of actors: friendly and hostile. A friendly actor helps us advance in the story and interacting with them changes the game state. Hostile actors produce a penalty when we interact with them. The next section describes the game states.

2.1 Game levels and State diagram

The game is structured into six levels:

1 Current-day Girona
2 The St. Feliu Church
3 Outside of the city walls during the 1285 Siege
4 The city wall gate puzzle
5 Inside of the city walls
6 The labyrinth

Screenshots of the different levels can be seen in figure 2. A description of the state diagram shown in figure 3 follows.

Figure 2. Levels of the Legends of Girona game

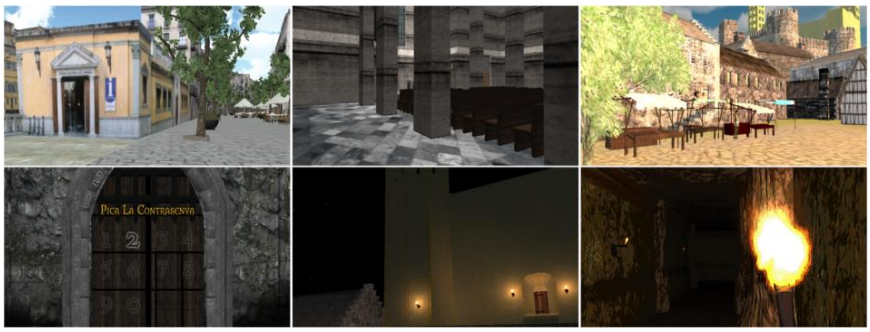
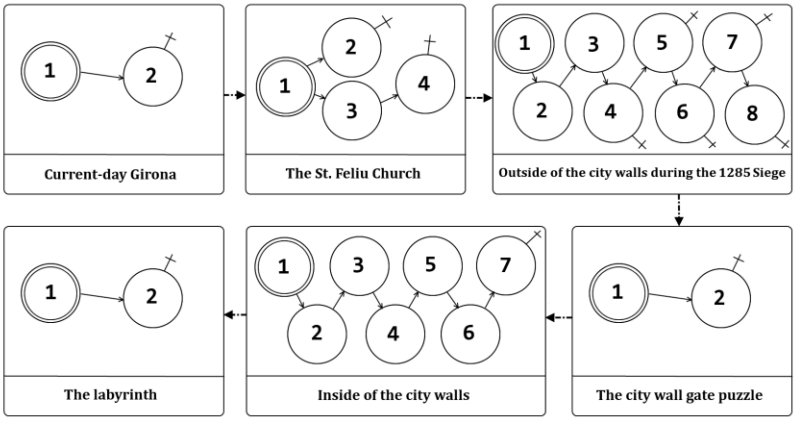


Figure 3. State diagram of the Legends of Girona game



Current-day Girona

Current-day Girona is divided in two states: the initial state, in which movement is limited to a five meter radius of the original position, and which only allows interaction with the tourist office.

After interaction, the second state allows the user to move along the streets leading to the St. Feliu Church. Entering the church loads the second level.

The St. Feliu Church

This level has four states.

1 In the initial state we may either find a key or a door.

2 The key produces a change in state prompting the user to search for the door.

3 Seeing the door without the key will prompt the user to search for the key.

4 After the key has been found in state 3, the user is prompted to return to the door.

In states 2 and 4, the door may be activated to load the third level.

Outside of the city walls during the 1285 Siege

This level is the most complex one, with eight states which include an optional intermediate sub-mission.

1 In the initial state we see how the protagonist wonders where he is and concludes that he needs to search for help. After the initial monologue is ended, there is an automatic transition to the next state.

2 We are outside of the city of Girona surrounded by guards to avoid and with the objective of finding help. The player is shown in a first-person perspective. To change state, we must find the Merchant and ask for help, after which state 3 is achieved.

3 In this state we wear medieval clothes to be less conspicuous and a pendant marking us as the chosen one. A third person perspective is used to ease navigation in the city outskirts. The objective is to talk to a lame man in the city while avoiding the guards. Once we talk to the man state we reach state four.

4 When entering state 4 we see night fall on Girona and the lame man has told us how to enter inside the city walls. From this moment we may go to the city gates and that will induce the load of the next level. If we want the walk to the gates to be easier, a sub-mission may be done so that the lame man helps us distract the guards. The sub-mission consists in getting 500 coins for the lame man. To change the state, we can go and talk to the Merchant, which triggers state 5.

5 In this state we have a package given to us by the Merchant. We may take the package to the blacksmith's house so that the Merchant gives us the coins we need. After delivering the coins, state 6 is entered.

6 In this state we need to tell the Merchant that the package has been delivered. Once we do, we receive the 500 coins and enter state 7.

7 In this state we have the coins and giving them to the lame man ends the sub-mission, leading to state 8.

8 The lame man distracts most of the French guards which were patrolling outside of the city wall. We only need to go to the city gates to change to the next level.

The city wall gate puzzle

This level has two states, the initial one until the year of the siege is provided by the user, and a final state to provide the transition to the next level.

Inside of the city walls

This state has seven states, but without optional sub-missions.

1 We start in the inside of the city walls and must investigate to know what to do. After talking to a boy near the gates, state 2 is reached.

2 In this state, our purpose is clearer. We need to search for a sorceress and request her help. After crossing the city to find her and talking to her, state 3 is triggered.

3 Now we need to use a basket (given to us by the sorceress) to collect a series of flies which are located all around the city. The higher the difficulty level, the more flies we need to find and the less time we have to complete the task. If the task is complete, state five is triggered; otherwise state four will occur.

4 Time is over, and the flies have escaped the basket, breaking it. We need to talk to the sorceress again to get another basket, after which we reach state 3 again.

5 The basket is full of flies, so we need to go back to the sorceress' place. After talking to her, she gives us a hint and state six is reached.

6 We now need to talk to a Jew to get the labyrinth's key. After talking to him the transition to the 7^{th} state is triggered.

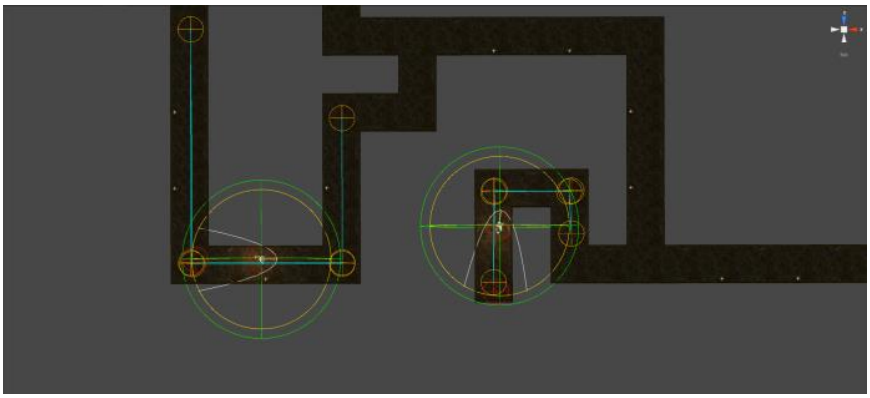
7 We have the flies, the hint and the key, so when we reach the labyrinth door the next level is loaded.

The labyrinth

This level also has two states, the initial one and a final one with the end of labyrinth animation, which leads to the end of the game.

2.2 Artificial Intelligence

Figure 4. Pathfinding and detection zones for soldiers



The control of the behavior of the non-player characters is based on artificial intelligence rules to predict and anticipate the player (Rich & Knight, 1991). This is the most relevant aspect in the city guard characters, which follow the player through medieval Girona.

Three aspects need to be taken into account:

Pathfinding algorithms to patrol and move All soldiers and some scripts which will be applied in the future to simulate the population of both current-day and medieval Girona use the standard pathfinding integrated in the Unity 3 engine to calculate the path to their destination. The colliders of the objects and the terrain of the scene are used to create a walkability map; this map can be used to calculate a path to any point, as long as the point is contained in the map. Furthermore, it includes a waypoint system so that soldiers patrol in the zones delimited by them. Soldiers will decide which waypoint to visit next using a probabilistic function. To simulate more characters, waypoints which are not visible for the player, and which allow NPC to appear and disappear also exist. Figure 4 provides an overview.

Detection algorithm All soldiers have a detection algorithm to find objectives in a circular detection zone. Additionally, a vision zone exists, modeled by a parabola in the (x,z) plane. When an enemy enters the intersection of both zones, the soldier will prosecute the enemy. The player has two ways to flee: move farther than a predefined distance of the guard, or moving out of sight by hiding behind a building. If the guard cannot see the player, he will go to the last point the player was visible from the guard position and search around randomly for a few meters. If the player stays out of view, the guard will find the closest point in its patrol area and return there.

Prediction and anticipation algorithm This algorithm uses the last movement vector of the player to calculate the future position of the player, so that the guard tries to intercept the player at that point. This is especially effective for soft turns (easily anticipatable) and hard turns, as this reduces the player speed and allows the guard to close in.

Additionally, all behaviors are affected by the difficulty level, thus:

Pathfinding algorithms to patrol and move The patrol and persecution speeds are affected. At high difficulty levels, the guard and the player run at the same speed, so the user cannot exit the prosecution area and needs to hide.

Detection algorithm The detection and prosecution radii are increased in high difficulty levels and decreased in low difficulty levels. The latus rectum of the vision field parabola (defining the amount of periferic vision) is also increased or decreased depending on the difficulty level.

Prediction and anticipation algorithm The prediction distance for the guard is changed, using optimal levels for high difficulty levels and disabling the feature for low difficulties.

We have requested users' impressions on the behavior of the guards, and the response has been positive. These easy to implement procedures create a behavior for the guards which the users of the game consider realistic.

2.3 Realistic Graphic Effects

The realism in a video game contributes to game immersion and enjoyability. We are using the GameTools routines (García et al., 2012) to produce realistic materials such as metals, including pseudo-raytracing reflections, in realtime and with low computational cost.

The routines work by creating and updating distance and color cube maps centered at the objects with the GameTools materials, and using a GPU shader to evaluate the correct pixel color using a binary search on the distance cube map followed by a texture access to the color cube map. The use of this technique provides realistic images even when the maps are only updated rarely, providing highly realistic images with low computational cost. The slight decrease in frames per second (FPS) does not affect gameplay. These routines can be integrated easily in already existing games using drag-and-drop of templates to create the necessary environment maps. The resulting effects can be seen in figure 5.

Figure 5. Realistic reflections using GameTools effects

2.4 Stylized Rendering

In certain cases such as historical storytelling, artistic stylization may provide greater experience and a more memorable atmosphere than realistic depiction. We adapted a collection of screen-

space Non-Photorealistic Rendering (NPR) techniques to Unity, enabling game designers to easily change the rendering style of the game. We designed our effects based on popular techniques from visual arts, especially comics. A common characteristic of comics is simplified depiction with large flat surfaces and minimal texture details, while enhancing important information such as character contours. In order imitate this style, we implemented texture simplification methods and luminance quantization based on image and video abstraction (Winnemöller et al., 2006, Kyprianidis & Döllner, 2008) to remove unwanted details, and the artistic edge detection of Winnemöller (2011) to enhance important ones. Changing the color of rendered shadows can create a variety of effects, such as impressionism or chiaroscuro (Sauvaget & Boyer 2010). Shadowed pixels are determined by rendering the image once with and once without shadows and comparing the two results as a post processing; shadow color is changed using the obtained mask. To enhance the illusion of depth, we provide depth varying detail level in textures, edge thickness and color saturation; each technique commonly used by painters and comic illustrators (Magdics et al. 2013). The atmosphere of the rendered images can be changed by adapting the color palette to a given example image, as in Reinhard et al. (2001). Most of the effects can be used with no measurable effect on speed. At Full HD resolutions, the Abstraction effect can reduce FPS around 25 %, while shadow effects are the most expensive, at 33 % reduction in frame rate. All our effects are applied to a camera object as a post-processing filter (Magdics et al., 2013), permitting them to be used in any games. Figure 6 illustrates several styles created by our implementation.

Figure 6. Stylized renderings of Legends of Girona (left to right): black and white, depth-based desaturation, color style transfer.

3 OUTPUT DEVICES AND TRACKERS

Output devices can be classified in two types: large, room-sized displays which maintain orientation as the user moves inside, and small, wearable displays which are positioned in front of the users' eyes and change orientation following the user's movement. In this last case, orientation trackers are needed to maintain coherence between the user's orientation and the virtual world orientation, in order to avoid motion sickness. Position trackers can also be used to move in the virtual world, but normally the presence of walls and other obstacles confine the user to a small footprint. Therefore, position tracking is used in conjunction with more traditional input devices to move in the virtual world.

3.1 Domes

A dome is a large, curved structure surrounding the viewer capable of displaying images and video. The following sections contain specific examples of types of devices, a description of

their calibration procedure and our method to support rendering and interaction with the devices. The use of these devices increases the computation load of the GPU, as some extra processing needs to take place.

We have developed a unity package which can integrate easily in existing in games and add the required functionality to display in immersive displays. The package consists of a cubemap-generating game object and a set of post-processing GPU shaders to generate the final image (some details on the internal structure of the package can be seen in Garcia (2013)). In a 3.1 GHz Intel Core i5 Mac with 8 GB of RAM and an AMD Radeon HD 6970M with 1GB of RAM, the game can be rendered at 200 FPS in a flat monitor, while the immersive visualizations render at 66 FPS. The reduction in performance is compensated by the increase in realism, and in any case does not impede gameplay.

3.1.1 Equirectangular projection

The equirectangular projection is commonly used for panorama viewers, and some dome devices use it as input. Figure 7 shows the output of the corresponding shader in our system.

Figure 7. Equirectangular projection.

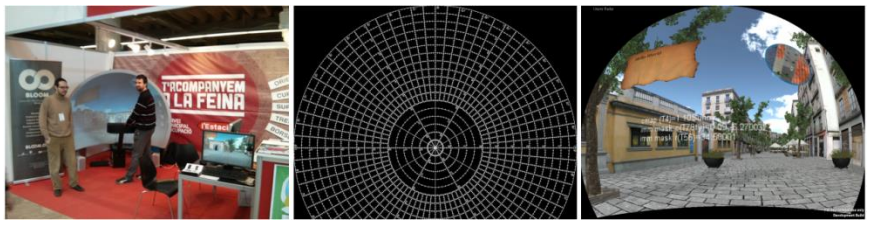
3.1.2 Immersapod

An immersapod consists of a curved, sphere cap reflective screen and a base containing a projector (figure 8 (a)).

The projector is located in the black metal box situated in front of the screen. To calibrate the positioning of the projector, a grid is used to align the image to the device (Figure 8, (b)). Care should be taken to position it so that the borders of the image match the borders of the screen.

Since minute changes in position and orientation of the projector can make a noticeable difference in position of the bottom border (projecting a strip of the image on the walls behind the immersapod), we have added a circular mask aligned to the bottom of the immersapod to avoid the issue. Figure 8, (c) shows the procedure to calculate the parameters of the mask using the keyboard. Also visible is the selection of the correct cubemap for the sky; the size of the screen requires some tweakings in the ambient light of the scene and the intensity of the colours in the skymap.

Figure 8. Immersapod. (a) Photograph at the Expojove Fair in Girona, Spain (b) Grid to aid in positioning of the projector. (c): Finetuning of the shader masks



The immersapod master shader requires two angles: the angle at which the sphere cap is cut at the bottom (figure 8, (d)), and the field of view, in addition to the center and radius of the bottom mask. With this information, the use of an azimuthal equidistant projection with the correct field of view (over the forward direction) provides an output image adapted to the display device.

3.1.3 Spherical Dome

Our spherical dome uses a fisheye lens from Immersive Adventures (Pla 2013). External and internal views of a spherical dome can be seen in figure 9. Other systems may use multiple projectors and use either a standard projection (see section 3.1.1) or a specific calibration (section 3.1.4).

Figure 9. Spherical dome at the BLOOM Centre in Girona (Spain). External and internal views.

The projector should be located in the centre of the dome, at the desired height. A crossbar image can be used to help in the task, since the dome surface is composed of a circular patch in the top plus lateral patches. Commonly, these devices are used as planetariums, so the projector is located in the center of the sphere. This gives a 90º vertical viewing angle. For videogame visualization, however, this obscures half of the dome, and characters are only displayed from breast height upwards when standing. Placing the projector on the ground creates a visualization in which characters are displayed from knee height upwards, providing a much higher immersion. The visualization shader requires two parameters: the vertical angle the projector spans, and the distance between the center of the room and the projection position.

To generate the output image, we again use an azimutal equidistant projection with the correct field of view, but this time over the vertical direction.

3.1.4 General Environments

As home cinema systems increase in quality (screen size, resolution, surround sound), we can see them becoming an increasingly immersive experience. In recent systems, the addition of projectors can transform the room into a fully immersive display, although still using a lower pixel density than televisions or monitors. For example, the IllumiRoom system (Jones et al., 2013) uses a Kinect to recover the 3D information and colour of a wall and nearby furniture. This information can be used to create an overlay image which may transform both the appearance and perceived geometry of the wall (glossy effects and geometry are only undistorted from a specific point of view (the virtual center of projection) which corresponds to the user's head position.

To provide a fully immersive environment, we have used a set of projectors covering a 360º field of view in a variety of setups, using between 2 and 14 projectors on different surfaces such as cloth (figure 10 (a),(b),(e)) or aluminium (c). The surfaces included imperfect cylinders ((a), (c), (f), (g)), and irregular shapes (b).

Figure 10. Different setups using autocalibration for general surfaces. (a) First setup (b) Use of sheets (c) Aluminium screen (EDM lab, Hasselt, Belgium) (d) Heinrich Herz Institute TIME lab (Berlin, Germany) (e) Future Internet Week 2010 (Ghent, Belgium) (f) Pukkelpop festival, 2011 (Hasselt) (g) Creativity World Forum (Hasselt)

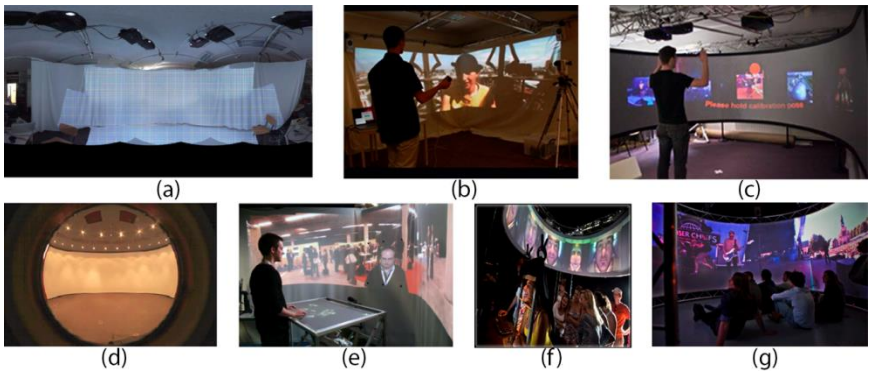
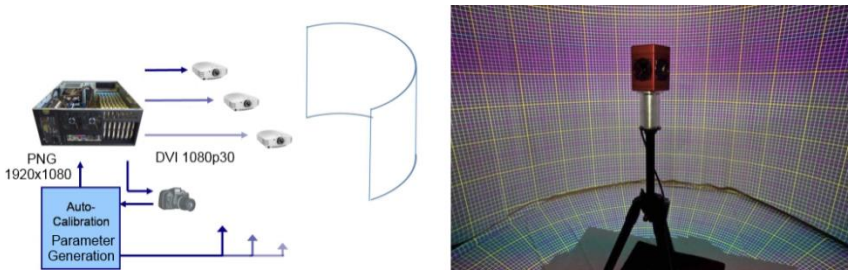


Figure 11. Calibration for ad-hoc surfaces. Scheme (left) and setup (right)



The calibration setup can be seen in figure 11, left. The calibration may use a surround video camera or a camera with a fisheye lens to provide a view of the whole virtual screen.

At the beginning the projectors are roughly aligned to ensure an overlapping area between neighbouring projectors. The camera is positioned in the centre of the projection and nearly at

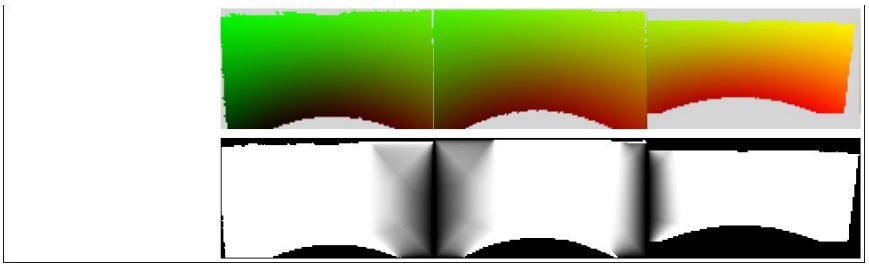
the height of eyes. The camera's shutter speed is adjusted to the frame rate of the projectors to expose at least one frame; the fisheye lens focuses on the screen, the aperture is used to avoid burnt lights with respect to the minimum depth of field which gives a sharp image of the screen (figure 11, right). The position of the calibrating camera defines the spot from where projected imagery will be seen as correctly perspective no matter the actual shape of the screen. A simple monochrome, low framerate, surround video camera is enough for the calibration.

The auto-calibration procedure shutters all projectors except the one in sequence. It activates the patterns and triggers the camera for every calibration pattern. The captured image is stored on hard drive. Once the capturing is done all data is assigned to the auto calibration tool. The return values of the auto calibration tool like a warping field for distorting the images, blending masks and gamma tables provide the transformation between the environment of the camera in the virtual world and the screen coordinates in each projector to achieve a multi projection with seamless overlapping.

We use a non-parametric approach to the calculation of 2D warping fields for geometric alignment of projector images, taking into account all relevant properties in the stitched surround image projection format. In contrast to parametric approaches, our method produces seamless images also on screen which are not perfect cylindrical or spherical, but also all casual setups we have realised and which are more similar to what could be expected in the immersive home cinema of the future. We most often use an equirectangular projection format for stitched surround images and have occasionally exploited a stereographic projection format. The former is better suited for approximately cylindrical displays, while the latter is more appealing for screens with dome- or sphere-like shape. By mapping to the same surround image projection space, projector overlap is easily established allowing gain and color equalisation between projectors. It turns out that the stitching approach is better conditioned than the screen shape reconstruction approach. Our method allows to obtain visually seamless calibrations of completely ad-hoc multi-projections on curtains for instance. Even with imprecise stitching, seamless projector image alignment results. Only absolute localisation accuracy is compromised.

The end result is a multi-projection calibration approach that has the potential of being applicable by non-experts. Under certain conditions it is even fully automatic.

Figure 12. Top: UV coordinates of the projectors. Bottom: Intensity mask of the projectors. The left quarter is blank since it is mapped to the controlling monitor.



The calibration pattern sequence we used in our work is an original sequence that combines advantages of calibration patterns used for different purposes including camera lens distortion calibration, camera rig intrinsic and extrinsic calibration, and structured light depth scanning. We most often use a sequence of 20 patterns per projector. By synchronising pattern projection and camera capture, this stage of the calibration procedure could be performed in the order of a few

seconds per projector. Figure 12 shows a set of projector-camera correspondence maps obtained by the algorithm.

3.1.5 Cylindrical Dome

Some cylindrical domes use as input the equirectangular projection mentioned in section 3.1.1, especially if the callibration was performed by the manufacturer. In our case, we have built an ad-hoc device (figure 13). Therefore, we are using the approach described in section 3.1.4.

Having a perfect calibration can be a time consuming process, especially for high-density control points. In some cases, the calibration created in the previous section might not be perfectly converged to the correct solution, or other biases might be present. However, we have found that averaging the central texel with its immediate corner neighbours can provide convincing images. These types of artifacts disappear if an analytical mapping exists between the pixels and the environment.

Figure 13. Cylindrical dome at the EDM in Hasselt (Belgium)

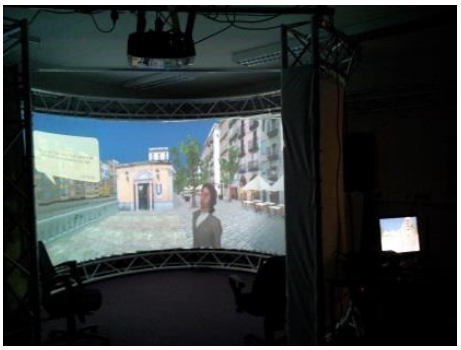
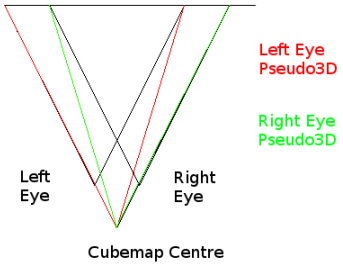


Figure 14. Creation of pseudostereoscopic views by changing the projection angles from the cubemap. Compare the real projection (in black) with the skewed projection from the cubemap centre (in colour).



3.2 Stereoscopic Glasses, Helmets and monitors

We have tested the re-use of the cubemap to create pseudo-stereoscopic views of the scene (see figure 14), but the images generated were not convincing. Therefore, we decided to create a real stereoscopic projection by the use of two cameras.

3.2.1 Sony HMZ-T1 Stereoscopic Helmet (Sony, 2011)

This device, shown in figure 15 (a), contains two small screens to cover the view area of the eyes of the user. We attached a PNI Sensor Corporation SpacePoint Evaluation Board for orientation tracking. The helmet has three modes of operation: 3D off, Top-bottom split screen, and Side-by-side split-screen (see figure 15 (b, c, d)).

Figure 15. Appearance (a) and modes of operation of the Sony HMZ-T1 Stereoscopic Helmet. (b): off; (c): Top-bottom split screen; (d): Side-by-side split-screen

To generate the image corresponding to the correct mode of operation, the camera images for the two eyes are combined using simple translation and scaling. If the 3D mode is off, the creation of the image for the right eye is disabled.

3.2.2 Head Trackers

Using the information about the position and orientation of the head of the user to guide the visualization can provide a high boost to immersion. Different types of sensors exist. The Kinect is able to recognize the user's position and orientation by the use of a depth camera. Other sensors are worn by the users and include accelerometers, gyroscopes or magnetic detectors to recognize their position and orientation in space.

We have used a PNI Sensor Corporation SpacePoint Evaluation Board (PNI, 2014) to add tracking capabilities to the Sony HMZ-T1 mentioned in section 3.1.7. The device is detected as a seven-axis joystick, where the first three axes contain accelerometer data in the X, Y and Z directions, while axes four to seven contain a quaternion describing the orientation of the device with respect to the magnetic field. We have used this quaternion to provide orientation data to the camera.

This provides an increased level of immersion, but the movement of the body is not taken into account to produce position displacements. Therefore, a residual motion sickness feeling may appear in some cases. If some space is available for the user to move in, adding position tracking is a solution. Maesen et al. (2013) developed a system which uses an array of LED lights located in the ceiling and a camera to provide a low-cost, high-precision position and orientation tracker which is immune to drift and which scales to very large workspaces (figure 16). The LEDs' on or off status encodes position information. Computer Vision techniques are used to detect the LEDs

and to retrieve the position and orientation of the camera. We have augmented this system to provide a TCP server which sends position and orientation data over a socket to client applications. In the virtual world, a translation, rotation and scaling is used to create the correspondence between the user's position and orientation and that of the avatar. Due to the possibly small space available and to avoid large neck movements which are tiring to the user, traditional input is added to the system so that the user can explore a small space before using traditional means to move to a different area (as Antonov et al., (2013), section 5.4 suggests).

Figure 16. Hardware required for position and orientation tracking based on Computer Vision. Left: Camera to be attached to the Virtual Reality helmet. Right: Arrays of LED lights in the ceiling.

3.2.3 Other stereoscopic devices

Some device manufacturers provide drivers which take the information provided by an unmodified program and adapt it to create stereoscopic effects, not allowing the programmer to specify left and right eye images. Instead, the graphics drivers intercept drawing calls and create the left and right eye images from these calls. The idea of this approach is that no additional work is required by the software developer.

However, the use of these very low level primitives makes it difficult for the driver to understand the intent of the programmer, so artifacts will appear unless the programmer follows specific guidelines (see for example McDonald (2011)). In particular, the document cited mentions a few very un-intuitive caveats such as activating an unused z-buffer and disabling reading and writing to it, and applying transforms to the coordinate to change eye separation without affecting rendering.

Since there is no guarantee that the guidelines for different devices will be compatible, we consider that the best quality results will be obtained by tweaking the output according to the specific device, in a similar way to the approach described in this paper. Examples of devices following this approach are the Nvidia Geforce 3D Vision glasses (Nvidia, 2013) and the LG D2500 PN autostereoscopic display (Shinymedia, 2011).

3.3 Simultaneous use of multiple devices

Current graphics cards allow the use of multiple monitors to provide a virtual monitor with higher resolution. The real monitors are mapped to rectangles in the framebuffer of the virtual

monitor. In the case of immersive devices, these rectangles may map to different projectors or monitors.

Our architecture based on shaders allows us to render to multiple immersive devices at the same time with little additional cost. Since our vertex shader is common, we can create a pixel shader which tests the coordinates to know to which device the pixel belongs to, and call the corresponding pixel shader after a linear transformation of the coordinates. For example, our cylindrial dome contains a control monitor and three projectors, mapped to a 4x1 Full HD display, in left to right order. Figure 16 shows the control monitor displaying one view of the scene while the projectors use their own transformation.

3.4 Kinect

In addition to realistic graphics and game physics, functional realism can greatly enhance immersion in the virtual world. Natural User Interfaces (NUIs) allow users to perform different actions in the game as they would do so in the real world. We have integrated the Microsoft Kinect into our game, and implemented a Unity package based on the official OpenNI wrapper for Unity and the NITE middleware that supports control by user gestures. The package contains a compact Kinect control object dealing with user selection, skeleton extraction and gesture tracking and provides a selection of gestures that can be added to any scene using drag-and-drop. The Kinect control integrates seamlessly into the system, it works in conjunction with the standard mouse-keyboard input without any modification of the underlying game logic.

Game control in Legends of Girona consists of navigation (moving, rotating and jumping), interaction with game objects and help (world map and quest log). Our implementation is an "exergame" requiring the user to mimic the real world actions, for example, walking in place toggles the avatar's forward movement, jumping makes the avatar jump and opening the map requires the arms to be spread (similarly to holding a large map). Figure 17 shows examples for the different gestures in the game. The performance of the game was not significantly affected by the use of Kinect.

Figure 17. Examples of Kinect gestures (left to right): jump, walk, open map.

4 IOS PORT

To port the Legends of Girona game to the Apple iPad 2 hardware, some modifications and adaptations had to be carried out to ensure correct performance. In particular, the polygonal load, the draw calls and the memory load had to be reduced. Additionally, specific controllers were designed. The game was ported during a period of three weeks by a single developer.

The polygonal load in the scenes was reduced by simplifying some of the models in the current-day Girona level. The decorations in the Rambla street have been remodeled, and the tree models have been simplified. Level three (outside of the city walls) required similar changes, plus a limitation on the number of soldiers. Furthermore, night scenes suffered from an excessive number of lights (the torches). Although Unity will disable non-visible lights, a more stringent method was developed and programmed to ensure that only the most nearby torches are active.

To reduce the memory load, in order to allow execution on the iPad 2, the resolution of all game textures was lowered, and the game audio was changed to use streaming from disk, alleviating the memory pressure. The current-day Girona level required more drastic measures: the scene was changed to take place during the evening, in order to justify a decrease in visibility. The far plane of the camera was reduced to a quarter of the original distance, and black fog was added to hide the transition. The number of loaded textures was thus sufficiently reduced to fit the whole scene in memory. However, during scene transitions both scenes are loaded, so in order to avoid this (which would put the memory use above the permitted limits) a new, empty scene was added to enable the unloading of all the resources from the previous level before loading the next one.

To adapt the controls, the solution chosen was to add two simulated analog controls, one on the bottom right corner and another one in the bottom left corner of the touch screen. The right one controls the movement of the player while the left one controls camera rotation. To ease gameplay, the actions to take and activate objects, and to talk with other characters, are now performed automatically when the player comes in close vicinity to them. The gameplay has also been changed to use a first-person perspective in all scenes. The game performance is 25-30 FPS, so playability is conserved.

5 CONCLUSIONS

We have shown the detailed architecture and design of the Legends of Girona game. The main purpose of this serious game is teaching the history and legends of the city of Girona to high school students, while providing an enjoyable experience.

The use of different expressive rendering modes (abstraction, edge enhancement, etc.) provides an attractive, comic-like appearance. Photorealistic effects are also available to increase verisimilitude in the virtual world.

We also added support for virtual reality helmets, large curved displays (cylindrical, spherical and general) and position and orientation trackers to enable the player to move seamlessly in the virtual world.

These devices and rendering modes was included in the game by using a clear separation of input and output modules, to enable future inclusion of more devices and to be able to port other games to use the devices easily. We also show how to port the game to mobile architectures (iOS).

The result is a much better immersive experience with respect to previous serious games for teaching history.

6 FUTURE WORK

The work presented here can be extended in different directions. Our future work will focus in two different aspects: a) creating new content to show other historical periods and legends, and b) extend and abstract our libraries so that more graphic techniques (both photorealistic and non-photorealistic effects) and other interaction devices and techniques can be integrated seamlessly into games.

We are interested in further abstracting the input interface by using GestIT (Spano et al. 2014). We would also like to support other position and orientation tracking devices. In particular, we have plans to integrate support for the Oculus Rift (Oculus VR, 2013).

UHotDraw (Sagredo-Olivenza et al., 2013) is a project which aims to simplify GUI development in the Unity engine. We plan to collaborate with the authors in order to provide an easy-to-use, immersion-device-aware GUI combining the benefits of our two libraries.

We also plan to integrate the routines in different game engines such as Ogre (Ogre, 2014).

Many auto-stereoscopic displays can often create realistic left and right eye views for commercial games by intercepting the OpenGL or DirectX drawing calls. Specific buffers are created for the left and right eye, and the drawings calls are replicated for each eye with different camera parameters. We plan to use the same approach to generate a cubemap centered at the camera position, and add a post-processing call to the shaders described previously. This would allow the final user to use immersive devices for commercial games not prepared for them, as long as the game does not do software frustrum culling (the speed of frustrum culling in the GPU is reducing the need for software culling except in specific situations with very large geometry sets).

7 ACKNOWLEDGEMENTS

This work has been supported by the research projects coded TIN2013-47276-C6-1-R, IPT-2011-1516-370000 and IPT-2011-0885-430000 (Spanish Commission for Science and Technology), and by grants CONES-2010-0027 and 2014 SGR 1232 (Catalan Government). We would also like to thank the support of the BLOOM Centre 3D i Tecnologies Emergents, Girona.

The images from figure 6 were created in the 2020 3D Media EU FP7 integrated project (responsible Christian Weissig / Philippe Bekaert).

REFERENCES

Alberch-Fugueras, R. (2001). El miracle de les mosques i altres legendes. Barcelona, Spain. Curbet Comunicació gráfica.

Antonov, M., Mitchell, N., Reisse, A., Cooper, L.,LaValle. S., & Katsev, M. (2013) Oculus VR SDK Overview. SDK Version 0.2.5. Retrieved December 18, 2013, from http://static.oculusvr.com/sdk-downloads/documents/Oculus_SDK_Overview.pdf

Apple (2014) Apple - iPad 2 - Technical specifications for iPad 2. Retrieved January 16, 2014, from http://www.apple.com/ipad-2/specs/

García, R., Magdics, M., Rodríguez A.,& Sbert, M. (2013) Modifying a game interface to take advantage of advanced I/O devices: a case study. In: Proceedings of the 2013 International Conference on Information Science and Technology Application

García, R.J., Gumbau, J., Szirmay-Kalos, L., & Sbert, M. (2012) Updated gametools: Libraries for easier advanced graphics in serious gaming. In: Asian-European Workshop on Serious Game and Simulation, 25th Annual Conference on Computer Animation and Social Agents (CASA 2012). Nanyang Technological University

Gouveia, D., Lopes, D., de Carvalho, C.V., Batista, R. (2012) Time Mesh: An Educational Historical Game. Digital Game and Intelligent Toy Enhanced Learning (DIGITEL), 2012 IEEE Fourth International Conference on. Pp 171 – 173.

Jones. B. R., Benko. H., Ofek. E., & Wilson, A. D. (2013) Illumiroom: Peripheral projected illusions for interactive experiences. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '13, pages 869-878, New York, NY, USA. ACM.

Kyprianidis, J.E., & Döllner, J. (2008) Image abstraction by structure adaptive filtering. In: Proc. EG UK Theory and Practice of Computer Graphics. pp. 51-58

Maesen, S., Goorts. P., and Bekaert, P. (2013) Scalable optical tracking for navigating large virtual environments using spatially encoded markers. In Proceedings of the 19th ACM Symposium on Virtual Reality Software and Technology, VRST '13, pages 101-110, New York, NY, USA. ACM.

Magdics, M., Sauvaget, C., Garcia, R., & Sbert, M. (2013) Post-Processing NPR Effects for Video Games. 12th ACM International Conference on Virtual Reality Continuum and Its Applications in Industry (VRCAI 2013). Hong Kong, China, November 17-19, 2013.

McDonald, J (2011) Nvidia 3D vision automatic. Best Practices Guide. Retrieved December 4, 2013 from https://developer.nvidia.com/sites/default/files/akamai/gamedev/files/sdk/11/3DV%20-%20BestPracticesGuide.pdf

Muzzy Lane (2012). Past/Present. Retrieved June 1 2014 from http://pastpresent.muzzylane.com/

Nvidia (2013) NVIDIA 3D Vision 2 Full HD Stereoscopic 3D glasses for your PC. Retrieved December 4, 2013, from http://www.nvidia.com/object/product-geforce-3d-vision2-wireless-glasses-kit-us.html

Oculus VR (2013) Oculus Rift - Virtual Reality Headset for 3D Gaming | Oculus VR. Retrieved December 16, 2013, from http://www.oculusvr.com/

Ogre (2014) OGRE - Open Source 3D Graphics Engine. Retrieved January 15, 2014, from http://www.ogre3d.org/

Pla, A. (2013) Immersive Adventure. Retrieved December 18, 2013, from

http://www.immersiveadventure.net/en_hard.html

PNI (2014) SpacePoint Scout | PNI Sensor Corporation. Retrieved January 16, 2014, from http://www.pnicorp.com/gaming/SpacePoint_Scout

Reinhard, E., Ashikhmin, M., Gooch, B., & Shirley, P. (2001) Color transfer between images. IEEE Computer Graphics and Applications. 21(5), 34-41

Rich, E., & Knight, K. (1991) Artificial intelligence (2. ed.). McGraw-Hill
Sagredo-Olivenza, I., Flórez-Puga, G., Gómez-Martín, M. A., and González-Calero, P. (2013) UHotDraw: a GUI framework to simplify draw application development in Unity 3D. In P. A. González Calero and M. A. Gómez Martín, editors, Actas del Primer Simposio Español de Entretenimiento Digital, pp 131-142.

Sauvaget, C., & Boyer, V. (2010) Stylization of lighting effects for images. In: 2010 Sixth International Conference on Signal-Image Technology and Internet-Based Systems, pp. 43-50.

Shinymedia (2011) IFA 2011: LG D2500N glasses-free 3D monitor. Retrieved December 4, 2013, from https://www.youtube.com/watch?v=eJFnhDCiAUk

Šisler, V., Brom, C., Cuhra, J., Činátl, K., Gemrot, J. (2012) Stories from the History of Czechoslovakia, A Serious Game for Teaching History of the Czech Lands in the 20th Century – Notes on Design Concepts and Design Process. Entertainment Computing - ICEC 2012. Lecture Notes in Computer Science Volume 7522, 2012, pp 67-74

Sony (2011) Head Mounted Display. Reference Guide. Retrieved January 16, 2014, from https://docs.sony.com/release/HMZT1_refguide.pdf

Spano, L. D., Cisternino, A., Paternò, F., Fenu, G. (2014) GestIT: a declarative and compositional framework for multiplatform gesture definition. ACM SIGCHI Symposium on Engineering Interactive Computing Systems 2013: 187-196

Strayer, J. R. (1953) The Crusade against Aragon. In Speculum, Vol. 28, No. 1, pp. 102-113

Unity (2013) Unity Technologies: Unity - Game Engine. Retrieved February 18, 2013, from http://www.unity3d.com/ Accessed 18 February 2013

Winnemöller, H. (2011) XDoG: advanced image stylization with eXtended Difference-of-Gaussians. In: Collomosse, J.P., Asente, P., Spencer, S.N. (eds.) Non-Photorealistic Animation and Rendering (NPAR). pp. 147-156. ACM

Winnemöller, H., Olsen, S.C., & Gooch, B. (2006) Real-time video abstraction. ACM Transaction on Graphics. 25(3), 1221-1226