Life-sized Group and Crowd simulation in Mobile AR

Zikas Paul
Foundation for Research and Technology Hellas,
100 N. Plastira Str. 70013, Heraklion, Greece
zikas@csd.uoc.gr

Papaefthymiou Margarita
Foundation for Research and Technology Hellas,
100 N. Plastira Str. 70013, Heraklion, Greece
margarita@csd.uoc.gr

Mpaxlitzanakis Vasilis
University of Crete, Computer Science Department,
Voutes Campus,70013 Heraklion, Greece
csd3058@csd.uoc.gr

Papagiannakis George
Foundation for Research and Technology Hellas,
100 N. Plastira Str. 70013, Heraklion, Greece
papagian@ics.forth.gr

Abstract
When referring to augmented reality characters, their actual size comparing to their physical surroundings is an important factor responsible for creating more realistic and immersive environments. By creating life-sized augmented characters we are able to present them in real time with real characters and create applications where both real and digital characters co-exist, in ‘life-size’ dimensions, as opposed to mostly small-scale AR characters. In this work, we have created AR crowd and group simulations with life-sized augmented characters that simulate crowd behavior with real human locomotive motions and body gestures.

Keywords: Augmented Reality, AR Crowd Simulation, Virtual Characters, Crowd Simulation

1 Introduction

In recent years, Augmented Reality environments constitute a growing area of research and a medium for edutainment. Recently, due to the evolution of hardware technology is feasible to augment crowd simulations on mobile devices. One important aspect of the Augmented Reality environments is augmenting realistic characters in order to have more immersion. In order to have more realistic results we focus on creating life-sized characters. Our main novelty, is the creation of Augmented Reality environ-
ments with real life-sized crowd and group simulation, on current mobile devices ranging from phones, tablets to cardboard style mobile VR displays. To the best knowledge of the authors, such results have not yet appeared in the bibliography. Our virtual characters can support many real life behaviors like locomotion and performing gestures. [1] illustrated the use of an interactive life-sized virtual character in mobile AR and in this work, we extend this to include group and crowd simulation in mobile AR.

2 Previous Work

In this work, we have used the open source OpenGL Geometric Application (glGA) framework [2] for the AR group and crowd simulation. glGA is a lightweight, shader based C++ Computer Graphics (CG) framework which is supported by many mobile and desktop platforms. glGA contains many operations like compiling and loading shaders, textures, sounds, animations, 3D static meshes as well as rig meshes. glGA supports global Illumination for shadowed and unshadowed static meshes using Precomputed Radiance Transfer (PRT) methods, interactive VR characters with procedurally generated body and facial animation using the integrated Smartbody animation platform and AR camera tracking using the Metaio SDK.

[1] proposed a complete pipeline for robust AR authoring of interactive virtual characters using only mobile devices. These characters utilize capabilities from SmartBody platform which supports many behaviors such as locomotion, object manipulation, gazing, speech synthesis and lip syncing.[3] used marker-based AR to control the behavior of virtual characters. Behavior-markers correspond to certain behaviors, like locomotion on a path created by markers and attribute-markers specify certain attributes on the virtual characters.[4] proposed a system with crowd simulation in Augmented Reality that supports collision avoidance with objects in virtual space that represent objects of the real environment.[5] provides a survey with algorithms for creating crowd simulation in Virtual Reality and proposing methodologies for addressing certain problems when creating crowds like collision avoidance, interaction and locomotion. They provide a wide range of case studies in Virtual heritage and Safety systems with crowd simulation.[6] presents an immersive Augmented Reality environment with groups of life-size virtual characters with dramaturgical behaviors, at the ancient Pompeii Heritage site.

3 Markerless AR tracking

We employed the Metaio-SDK for the SLAM-based markerless AR camera tracking. For the AR character rendering and animation we employed the open-source glGA framework complemented with the open-source RVO crowd engine and SmartBody animation simulation toolkits. In order to augment our characters in real life we create a 3D feature-point map for our scene using the Metaio Toolbox application and pass it in the bundle of our application. The Metaio SDK determines the coordinate system base on the assigned 3D map and configures the tracking system. On every frame if is tracking we get the model matrix and the projection matrix and then, we apply these transformation matrices on every 3D object in our scene. Due to the fact that the transformation matrices got from Metaio are not life-sized, we had to give the functionality to the user to be able to manage the size, position and rotation of the entire scene. At the start up of the application the user manages the transformation of the scene using gestures.

4 Life-Sized AR crowd simulation

We have implemented life sized AR crowd simulation using the glGA framework. Our characters support PRT using HDR environment maps and locomotion behavior with collision avoidance, as shown in Figures 1, 2 below. Figure 3 shows the process followed for creating AR crowd simulation on a mobile device.

4.1 Collision Avoidance Algorithm

For the simulation of crowd movement, the RVO2 [7] collision avoidance library was used. Since RVO is agent based it matches well with
the needs of the project as each character is defined by a single agent. The initialization of the agents is done offline by specifying the characteristics of each agent such as the maximum speed and radius. After this step it starts to simulate the crowd and each agent moves towards the goal position that we have set while avoiding collisions with other agents. For the animation management we used Smartbody [1] to alternate between walking and idle animations. The path followed by the agent is most of the times a straight line. If any obstacle appear in front they will try to overpass it as close as possible. ORCA algorithm [8] used by RVO aims for the shortest path possible which is always a straight line, but this is not always the preferable especially when controlling a crowd. For example, when an agent should start walking backwards, is caused an instant rotation instead of walking in a circular way trying to approach the new direction. This drawback makes the character changes direction in one motion and creates a non-realistic result for the walking animation. However, RVO is very fast responding and have no delay between different commands providing smooth simulation and high framerate even with lots of agents.

In order to work properly we had to make some adjustments to the simulator considering the large scale of the agents caused by the life-sized characters. By changing parameters such as the time step of the simulation and the radius of each agent we overcome the large size of the agents and maintain a more accurate walking.

4.2 Path Roadmap

To make the agents walk into a certain path we need to define their goal positions and create a roadmap. RVO has already implemented an algorithm for roadmaps but it was not used in this application. Instead, a custom algorithm was made to make roadmaps usable. A roadmap is like a graph so it needs multiple points to represent a walking path. These points are stored in a unique vector for each agent as goal points. The agent has to reach all the goals of the roadmap to complete the path and come back to idle state. Once a goal is reached another will take its place in the vector and so on. Until every goal is reached, the agent continues to calculate the next velocity vector based on the goal points provided. Roadmaps are agent independent which means we can have multiple agents each one performing his own roadmap, as shown in Figure 3.

4.3 Quaternion based character rotation

On each frame RVO computes the velocity vectors for each agent. These vectors are used to determine the direction of the agent but we also use them to find their actual rotation. The agent’s body should always point in the direction of walking and to do that we need to know in what direction he is planning to go. Velocity vectors are two dimensional vectors so by calculating the angle of each one with a common coordinate system we can define the angle of the agent. This result is converted to a quaternion representation to generate the character’s rotation matrix.

4.4 AR group simulation with HMD

Taking advantage of mobile-driven AR we modified the existing crowd application to be used
with HMDs. In order to have stereoscopic rendering we took two slightly translated images from the camera’s feedback to create a unique image for each eye. Then we had to do the same procedure for the augmented characters to create the perception of depth. As we need to have stereoscopic rendering on a mobile screen, the field of view will be halved, causing less characters to fit in the view. By placing the mobile device inside the HMD we experience an AR simulation which enhance the presence of the application as the camera movements are done from the user’s natural point of view.

Figure 3: Process followed for creating interactive AR crowd simulation.

Figure 4: Process of Geometric and Photometric AR scene authoring of crowd simulation in indoor environment.

5 Results

In this section, we present our visual results and the performance of the AR application with crowd simulation, as shown in Figures 1, 2. Figure 4 shows the process of authoring the Geometric and Photometric AR scene with crowd simulation, based on [1]. At the beginning, the user creates a 3D map of the environment using the Toolbox application, send it via e-mail and open it with our application. The user manages the scale, position and rotation of the crowd to achieve life-sized augmentation and then manages the shading by adjusting the exposure. Finally, the user saves the current state of the application in order to have the same visual result when re-launching the application.

5.1 Performance metrics

When we start to build up the crowd and add more agents we faced a major memory problem. The mobile devices we used (iPad Air2/iPod5) were not able to run the application without being connected to a desktop for extra virtual memory usage. We concluded that the issue was due to the lack of sufficient mobile GPU texture memory to load the characters’ textures. The solution was to downscale the textures to 128x128. After that we were able to run the application with up to 10 characters on the iPad without having the device connected, maintaining a decent framerate of 28fps. Using the HMD and the iPod, which has less memory, we were able to load up to 7 characters with the same framerate, due to the stereo rendering of the same scene for each eye.

6 Conclusions and Future Work

In this work, we have presented AR environments with real life-sized crowd and group simulation. We have focused on creating real human size virtual characters that support human behaviors like locomotion and performing gestures. We have used the method described in [1] for including and authoring any virtual character in mobile AR and we have extend this method for populating group and crowd simulation in mobile AR. In the future, we aim to use OpenCV library for implementing AR camera tracking and compare its performance with MetaioSDK. Concerning the crowd simulation we plan to add real time guidance features in order to make the simulation more interactive. User will have more interaction with the virtual characters and be able to select and determine the path of each agent by touching the screen.
References


