Eurographics Italian Chapter Conference (2007)
Raffaele De Amicis and Giuseppe Conti (Editors)

The Dance of the Air: a case study in interactive digital art

E. Cuoghi[1], R. Gherardi[2] and R. Toldo[2]

[1] Fine arts academy “G.B. Cignaroli”, Verona
[2] Computer science department, University of Verona

Abstract

This paper describes the conception and realization of an interactive art installation called “The Dance of the Air”. The whole process is taken as a case study for the different challenges encountered during the implementation of the system. The project is followed from the seminal idea to the artistic specifications, the translation into technical requirements and the final deployment step.

Categories and Subject Descriptors (according to ACM CCS): J.5 [Arts and Humanities]: Fine Arts, Performing arts

1. Introduction

The use of computers and interactive media is enhancing and changing the expressive power of fine arts, and redefining the role of the viewer as integral part of the artistic process. These new types of art are not only exploring new forms of communication but are also a test-field for innovative or provoking ideas in intelligent environments and natural interfaces. The creation of these types of artistic platforms is usually a complete engineering task, requiring the confluence of several competences and backgrounds spanning from humanities to the more technical. This article describes the creation of one of them from conception to deployment.

“The Dance of the Air” is an artistic opera that can be either used for live performances or static installations. Its primal, most basic artistic statement and poetic suggestion consists in enabling its user to *see* the “air” that constitutes their environment, actually augmenting the perception capabilities of those who enter in its area of action. To achieve this result, the environment is monitored with the aid of an infrared camera. The sensed movements are then used to drive a physically-correct fluid simulation, which is composited in real-time with the video source and visualized by means of projection. The final effect is as engaging as thought-provoking: the fun is accompanied by a new perception of space that lasts after the experience.

The rest of the paper is structured as follows: in section 2 the artistic motivations and suggestions that gave birth to the opera are explained. In section 3 the diverse engineering challenges are covered, describing the adopted solutions. Finally in section 4 some conclusions are drawn.

2. From conception to requirements

One of the central issues in the critical debate on contemporary art and on the artistic explorations that are being conducted with the support of the new communication technologies is about the topic of *interaction*. In this scenario, the opera emerges from the fruition process established between the user and the “*technologically-aware*” reply of the opera itself. This kind of process, a new concept in the historical prospective of visual arts, inevitably puts the active viewer in a relation of reciprocal interdependence with the environment: they are inseparable and undistinguishable because they equally contribute in shaping the relation dynamics that constitutes the opera. The art produced is in this sense *relational*, containing both the user and the augmented, listening environment. The modality of interaction can be either *internal*, in which the user recognizes herself as part of the opera and consciously leverages its dynamics, or *external*. In this second case the viewer cannot perceive her influence relegating her role to a spectator.

The opera “*The Dance of the Air*” is part of a bigger and more articulated project called “*The Time Factor-y*”, comprising both static installations and live performance. The cohesive link is the concept of time, which fuels this opera as well. But more than the other works, “The Dance of the Air” is able to capture the aspects of indistinguibility and inseparability of the user and its technologically-aware environment with striking efficacy.

The intuitive idea behind this whole work is that the body moves itself in a *space-time* from which it can't be separated and that reacts to body movements with self-consistent dynamics. People move inside the air that surrounds them without perceiving directly with senses its "shape" and complex dynamics. We can't see the air moving around us, even if we are the direct cause of the motion. The process could be more evident if moving inside a fluid, or thick fog or dense smoke: in all those cases the frenetical and apparently chaotic motion can be directly observed, even if normally the physical characteristics of the air do not consent this perception.

The opera "The Dance of the Air" has this poetic but also technical objective: to make evident, *visible*, what happens to air while we move. When we move with the spontaneous and coherent motion of a conscious gesture, or when engaged harmonically in a athletic feat, or again when repeating quietly a usual action: in all these cases it's fascinating, and poetic, to think of air as "*dancing*" with us.

3. Construction of the hardware and software platform

In this section we describe all the components that actually constitute the final system. As stated in the introduction, the movements of the user acquired with an infrared camera are used to drive a physically-correct fluid simulation that is then composed with the original video and sent as output. In the following we will discuss the algorithm used for the fluid simulation, background modeling, foreground detection and fluid forces modeling. The software was written in highly-optimized C++ with a OpenGL back-end for visualization, enabling it to run on commodity hardware. Common webcams were used for acquisition; they were equipped with infrared filter when a dark setting was required. In these cases, an infrared illuminator was used to lit the scene.

3.1. Fluid simulation

For our work we used the fluid solver presented in [Sta99, Sta03], which is a fast and stable implementation of a fluid dynamics solver. The algorithms was originally designed for games and it is capable of sustaining real-time framerate without sacrificing physical correctness. Figure 1 shows four frames of a simulation.

Instead of using particles for faking the transfer of mass, this technique is based on the physics of the fluid flows. Most fluids are represented with mathematical precision by the Navier-Stokes equations; however this model only admits analytical solution in very simple cases. The Navier-Stokes equations for velocity (1) and density (2) are reported below.

$$\frac{\delta \mu}{\delta t} = -(\mu \cdot \nabla)\mu + \nu \nabla^2 \mu + f \tag{1}$$

$$\frac{\delta \rho}{\delta t} = -(\rho \cdot \nabla)\mu + \kappa \nabla^2 \rho + S \tag{2}$$

The technique maps the fluid velocity and density into a two-dimensional discrete grid and it finds a numerical solution of equations 1 and 2. The equation can be split into three terms: an additive term (f and S), a diffusion term ($\nu\nabla^2\mu$ and $\kappa\nabla^2\rho$) and an advection term ($-(\mu\cdot\nabla)\mu$ and $-(\rho\cdot\nabla)\mu$).

The additive term represents the external forces. At each step the new external forces (velocity and density) are simply added to the grid.

The diffusion term can be calculated expanding the ∇^2 term. At each step, for every element of the simulation grid the term is calculated from its previous value and its four nearest neighbours. However, a direct evaluation of this term leads to an unstable behaviour: for large diffusion rates the values start to oscillate and diverge. For these reasons a stabler method must be used, that is solving for the quantities that when diffused backward in time yield the starting densities. The resulting system is linear and can be solved with various stable algorithms. Since the matrix is very sparse a simple Gauss-Seidel relaxation can be used.

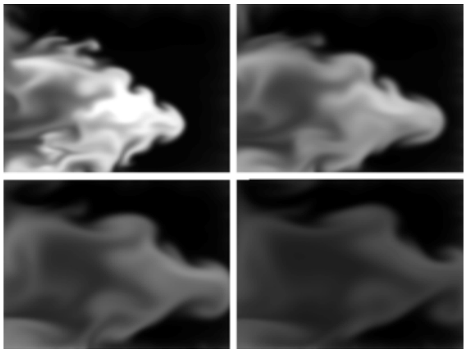

Figure 1: *Four consecutive frames of a simulation.*

The advection term is the most complex to solve, because the equivalent linear system now depends also on the velocity. However, considering the moving density as a set of particles, a possible solution is to trace them to the velocity field and to find those which, over a single time step, end up exactly in the considered grid cell's center. The carried density can then be found with a simple linear interpolation.

When updating the velocity grid, mass conservation must also be enforced constraining the velocity field. To achieve this result the Hodge decomposition is used: every velocity field is the sum of a mass conserving field and a gradient field. The gradient field can be found by solving a Poisson equation. The linear system is solved again with Gauss-Seidel relaxation. By simply substracting the gradient field to the velocity field we finally find a mass conserving field.

In the final simulation an additional term was added to inject some randomness in the modeled fluid from the borders. The motion of the fluid was bounded within the window border, fact that kept us from using the more elegant solver based on the fast fourier transform presented in [Sta01].

3.2. Background modeling and foreground detection

The state of the art offers many algorithm which combine very good background modeling capabilities with real-time performances [CMPC06, SG00]. For the sake of prototyping however, we first coded a quick background detection scheme based on a simple median filter applied to a bunch of consecutive frames.

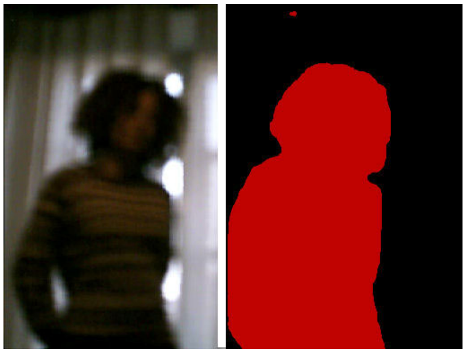

Figure 2: *Foreground segmentation with the median algorithm.*

This approach, which leads to a sub-optimal segmentation, revealed itself very effective and was kept in the final version. All this was possible also because we designed the algorithm that produces the velocity vector to be injected in the simulation to be very forgiving to segmentation errors. Finally, this approach was selected because it can successfully detect foreground on foreground motions, like for example a person waving a hand in front of herself. Figure 2 shows an example of the described algorithm in action.

3.3. Fluid simulation update

When the frame segmentation is known, the fluid simulation must be updated with velocities and densities compatible with the detected movements. For each cell, the velocity vector cell is extracted by calculating the gradient of the difference between the current and the previous grid on a eight-point nearest neighbourhood. Each foreground point adds a fixed amount of density: however, both densities and velocities are modulated by a value inversely proportional to the number of times that cell was detected as foreground in the last k frames. In conclusion at each cell, if foreground is detected, the values of density and velocity are added according to the following formulas:

$$\mu = \frac{K_\mu * K_t * \frac{\mu}{\delta t}}{prevCount + 1} \tag{3}$$

$$\rho = \frac{K_\rho * K_t}{2^{prevCount}} \tag{4}$$

where K_t represents the inverse of the frame interval time and is useful to make the simulation frames independent. The reason we multiply the additive terms by the inverse of the previous consecutive frames counts and by the exponential function in that value is to modulate the added density over the velocity and magnitude of the detected gesture.

4. Conclusions

In this paper we presented the construction of a interative digital art from concept to engineering. After being completed, the opera has been installed in a crowded mall in Verona, where it has generated a lot of curiosity and warm reactions.

References

[CMPC06] CALDERARA S., MELLI R., PRATI A., CUCCHIARA R.: Reliable background suppression for complex scenes. In *VSSN '06: International workshop on Video surveillance and sensor networks* (2006), ACM Press, pp. 211–214. 3

[SG00] STAUFFER C., GRIMSON W. E. L.: Learning patterns of activity using real-time tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence 22*, 8 (2000), 747–757. 3

[Sta99] STAM J.: Stable fluids. In *SIGGRAPH 99 Conference Proceedings* (August 1999), pp. 121–128. 2

[Sta01] STAM J.: A simple fluid solver based on fft. *Journal of Graphics Tools 6*, 2 (2001), 43–52. 3

[Sta03] STAM J.: Real-time fluid dynamics for games. In *Proceedings of the game developer conference* (2003). 2

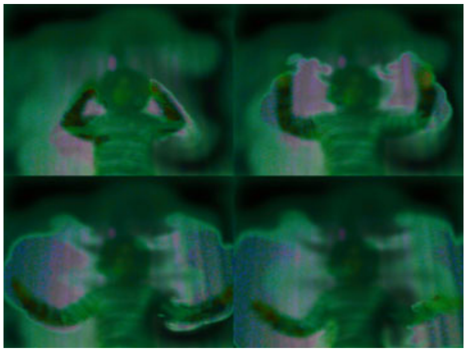

Figure 3: *A example of the system in use.*