Out of Context Augmented Navfields : Designing Crowd Choreographies

Guillaume Levieux[1], Stéphane Natkin[1], and Alexandre Topol[1]

Centre d'Etudes et de Recherches en Informatique du CNAM
Conservatoire National des Arts et Métiers, Paris, France
`forename.lastname@cnam.fr`

Abstract. This paper presents a way to dynamically influence the shape and movements of a simulated crowd. We propose a tool and system that allows to modify a crowd's dynamics in an intuitive, semantically rich and out of context fashion, while being independent from the global path finding architecture and having a low computational cost. We follow a mixed approach where user-specified navigation fields are combined with steering and global A* pathfinding.
Key words : crowd, choreography, navfield, simulation.

1 Influencing Crowds Dynamics

Many crowd simulation systems have been developed in the past few years. They have multiple goals like the study of sociological or psychological principles, simulation of a crowded building's evacuation, as well as digital entertainment or military training [1], [2]. These systems use different crowd models, ranging from fluids [3], particles [4], to basic entities influenced by a simple set of rules [5] or even autonomous agents with rich decision models [6] [7] [8].

The choice of a crowd simulation model depends on many factors. One of the biggest trade-off in crowd simulation is computing cost. The more complex an agent decision is to calculate, the lowest the number of agents can be simulated in real time. It may always be useful to have the most realistic agents, but the model has to scale with the number of simulated agents.

Moreover, as Zhou et al stated it, a crowd simulation system needs to be considered from the designer point of view [2] [9]. Having a complex decision system may lead to more realistic agents, but also make these agents less predictable and harder to manipulate. Indeed, using autonomous agents leads to a bottom up architecture, where the crowd's global behavior depends on the many local decision made by the agents. When a designer wants to influence the crowd's global dynamics, that is, the global shape and movement pattern of a large group of agents, we should provide him with a high level tools that directly deals with these dynamics. Our goal is to propose such a tool.

Our research takes place within the context of the Terra Dynamica project, in whichF a consortium of industrials and researchers got together to built a virtual version of Paris city. This simulator has a wide application scope, ranging from urban planning to artistic experiments, and follows an autonomous agents approach. In this project, our goal is to propose a simulation technique that can seamlessly be integrated in an autonomous agent architecture, and let a designer easily influence the shape and moving patterns of a crowd.

Designing tools to control a crowd simulation is a particularly interesting topic. Indeed, the crowd's dynamics emerges from the more and more complex agents behaviour models we set up, but still need simple methods to influence the crowds behaviour, especially in entertainment purposes, where the goal is not to always have a realistic simulation but to reach specific aesthetic goals. Researchers have already proposed such kind of tools. Crowd Brush allow to modify crowd scenes in Real Time [9]. Oshita and Ogiwara's system allows to sketch a crowd's path [10]. ARCrowd allows to control a crowd in real time using augmented reality [11]. Kwon et al systems let the user customize the motion of a group of individuals [12]. Many systems use navfields, that is, grids of direction vectors, to author the navigation of a crowd [13] [14] [3] [15].

Unfortunately, neither of these systems completely addresses our design goals for the Terra Dynamica project, that we describe as follows :

1. **Intuitive**: our representation of the crowds dynamics needs to be easy to author, and represented in the most intuitive way.
2. **Provide a rich semantic** : we do not only want to specify a direction an agent in the crowd should follow, but also allow this direction to be modified within time, or have some stochastic properties.
3. **Out of context**: we want to specify a crowd's dynamic for a group of agents, regardless of the size of the group, of it's absolute location and without knowing the agents that will be involved. That way we can define reusable, generic movements and instantiate them when needed.
4. **Independent**: we want our system to be used in combination with standard A* path finding and state of the art steering. We need it to be activated only when needed, and to be integrated seamlessly in the standard navigation architecture.
5. **Not computationally expensive**: as we use this system in addition with a complex AI system, and as we need to modify the simulation in real time, we need our system to be as light as possible in term of computational cost.

We extend the navfield principle to provide the designer with a more expressive tool. We call these specific navfields "*Out of Context Augmented Navfields (OCANs)*". In the next section, we show how OCANs addresses the needs we have specified in this section.

2 Out of Context Augmented Navfields

We draw inspiration from navfields based systems. Defining a grid of vectors is an intuitive way to represent a crowd's dynamics. Given the right tools, one can paint the vectors on the grid, and easily define the desired movement [15, 14]. The vector field is indeed a nice way to represent a crowd's dynamics, as the information displayed on screen is the resulting movement influence we try to specify. The basic navfield given in fig 1 (left one) could easily be encoded as a mathematical function or a piece of scripted code but we think that the navfield itself is the most intuitive representation to show and manipulate.

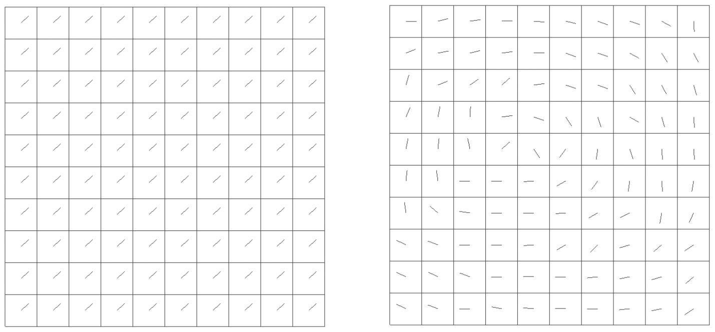
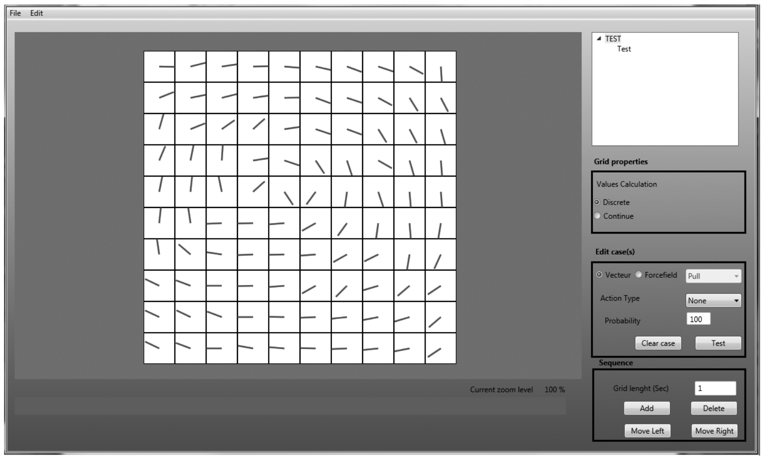


Fig. 1. Two OCANs

We created a GUI tool to define OCANs (fig. 2). The user can easily define an OCAN and edit the various properties that makes OCANs semantically rich. The user can edit the vectors one by one or using a brush. The different tools on the right side of the screen allows to modify the OCAN properties that we define in the reminder of this article. The user can then save all OCAN's in an XML file that will be imported by the choreography manager.

In our system, the user can define a **sequence** of OCANs. Each OCAN in the sequence has a pre-defined life span, and when this lifespan is over, the next OCAN of the sequence is instantiated. Thus, we can define a follow up of grids, which allows to define more complex crowd movements. Moreover, each OCAN is annotated by the user. It can be **continuous** or **discrete**, which means that the direction vector given to a specific agent only depends on it's discrete position within the grid. When continuous, the direction vector is the mean of surrounding vectors, weighted by the distance between the location of each cell center and the agent. The user can also assign a **probability** to each cell of the OCAN. That way, and agent will have a given probability to be influenced by the cell, and thus only a portion of the surrounding crowd will be affected. Last, any grid cell can be defined as a **placeholder**. If there is no agent in this cell, the nearest agent in the OCAN will be assigned a direction vector toward the placeholder. That way, it is possible to define formations that the agents within the grid will try to assume, in a best effort way.

OCANs are thus defined out of a specific context. One can define a grid representing a specific crowd movement, and instantiate it later at any location, in real time. Then, it will automatically impact the crowd at this location,



Values Calculation
[✓ Discrete] Continue

[✓ Vecteur] Forcefield Pull
Action Type None
Probability 100

Grid lenght (Sec) 1

Fig. 2. OCAN's Editor GUI

influencing their predefined direction vector. We use a specific software component, the choreography manager, that dynamically instantiate OCANs during the simulation, with the desired location, scale and orientation. An OCAN can be instantiated on various requests: for instance a scenario manager can ask for it, or an agent that is a squad leader and want his group to follow a specific dynamic.

However, as OCANs are defined out of a specific context and instantiated on demand, we must be able to integrate them regardless of the navigation algorithm that is used. Our only recommendation is that path planning and steering have to be defined as two different navigation steps. This is very often the case: in video games for instance A* is very often used to generate a global path, while different steering algorithms compute the fine grain movement of the agent [16]. As a result, OCANs will only override the global path finding direction when an agent steps on the grid. The steering mechanism will always be active and let the agent move in a way that is coherent with the other dynamic and static colliders, but it's main direction will be defined by the OCAN. When the agent leaves the OCAN, he will rely again on it's previous path-finding algorithm.

When under an OCAN's influence, the agent's direction is mainly determined by the cell he's in. We thus just have to convert the agent's position in the OCAN's local space to get this vector, defined by a 4*4 matrix, which is a very cheap calculus. If the grid is continuous, the calculation cost is just a bit higher because we compute a weighted sum of the surrounding vertices. The calculation cost is the highest with placeholder cells, because we need to find the closest agent. This process is speeded up by a previous space partitioning [17], allowing to find the closest agents without parsing all of them.

3 Integration Flow

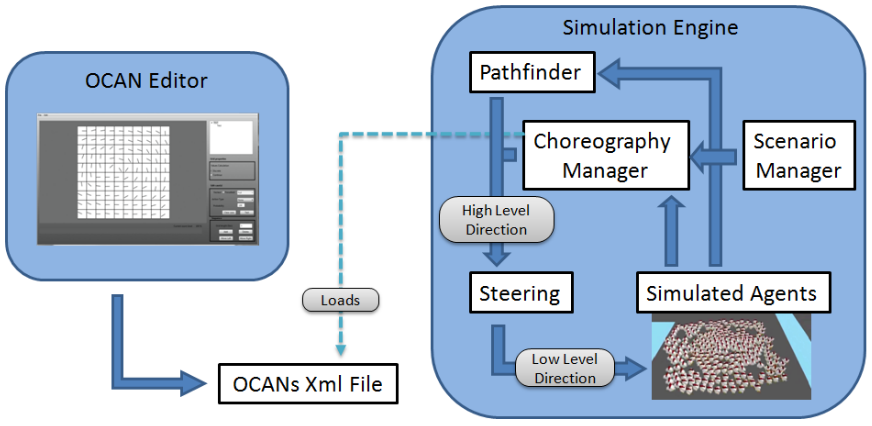


Fig. 3. The OCAN System

The previous diagram shows more in details how the OCAN system integrates into a standard path finding architecture, as provided by the TerraDynmica simulation engine (fig. 3). Using our dedicated tool, the designer creates sequences of OCANs. Each OCAN sequence is identified by a unique number and stored in a common XML file. When started, the choreography manager, part of the simulation engine, loads the XML file. The simulated agents can go to a desired location using the pathfinding system. The global pathfinding engine will generate a path, that will be filtered by the steering engine to accommodate with local constraints.

But when requested, the choreography engine will instantiate a specific OCAN. It will have the same properties than when designed in the editor, with a specific location and orientation decided at runtime. Many instances of the same OCAN can thus be active at the same time, at different locations. For evey agent whose position is inside one of the instantiated OCANs, global path finding will be replaced by a vector calculated by the choreography manager, based on the current agent position in the OCAN.

As a result, the choregraphy manager, the pathfinder and steering algorithm all work at the same time in an integrated fashion. The OCAN is just a way to by pass the agent decision process with regard to it's current global direction, only where and when needed. For instance, we plan to use OCANs to simulate military tactics in crowd management simulation. Indeed, extract a leader from a civilian crowd responds to a specific choreography : a first sqad runs toward the leader in a triangle shape to penetrate the crowd, expand it's position to make the crowd step back and isolate the leader. Meanwhile, another group follows and catches the leader. Then the first group concentrate again to secure the leader and everybody get back out of the crowd. This kind of choreography can be easily described using a sequence of OCANs, spawned by the squad leader when needed.

4 Conclusion

In this article, we propose a simple way to alter the dynamics of a crowd, using Out of Context Augmented Navfields. As we have shown, such a system can provide an intuitive, semantically rich way to define the shape and movement of a crowd, while having a low computational cost and be integrated in a standard navigation system. The OCAN system place itself between the global path finding algorithm and the local steering algorithm, overriding the global path finding only when needed. As a result, we do not have to take into account the global goal of the agents or the way they avoid obstacles, but just apply a local, smooth influence on their moving dynamics.

The next step of these research is to broadly evaluate the usability of our system, and to enhance it with an even richer semantic.

References

1. Musse, S.R., Ulicny, B., Aubel, A., Thalmann, D.: Groups and crowd simulation. In: ACM SIGGRAPH 2005 Courses. SIGGRAPH '05, New York, NY, USA, ACM (2005)
2. Zhou, S., Chen, D., Cai, W., Luo, L., Low, M.Y.H., Tian, F., Tay, V.S.H., Ong, D.W.S., Hamilton, B.D.: Crowd modeling and simulation technologies. ACM Trans. Model. Comput. Simul. **20** (November 2010) 20:1–20:35
3. Chenney, S.: Flow tiles. In: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation. SCA '04, Aire-la-Ville, Switzerland, Switzerland, Eurographics Association (2004) 233–242
4. Chen, Y., Wang, Z.: Formation control: A review and a new consideration. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. (2005) 3181–3186
5. Helbing, D., Molnar, P.: Social force model for pedestrian dynamics. PHYSICAL REVIEW E **51** (1995) 4282
6. Jiang, H., Xu, W., Mao, T., Li, C., Xia, S., Wang, Z.: A semantic environment model for crowd simulation in multilayered complex environment. In: Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology. VRST '09, New York, NY, USA, ACM (2009) 191–198
7. Shao, W., Terzopoulos, D.: Autonomous pedestrians. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation, ACM (2005) 19–28
8. Sakuma, T., Mukai, T., Kuriyama, S.: Psychological model for animating crowded pedestrians. COMPUTER ANIMATION AND VIRTUAL. WORLDS **16** (2005) 3–4

9. Ulicny, B., de Heras Ciechomski, P., Thalmann, D.: Crowdbrush: interactive authoring of real-time crowd scenes. In: ACM SIGGRAPH 2005 Courses. SIGGRAPH '05, New York, NY, USA, ACM (2005)
10. Oshita, M., Ogiwara, Y.: Sketch-based interface for crowd animation. In Butz, A., Fisher, B., Christie, M., Krüger, A., Olivier, P., Therón, R., eds.: Smart Graphics. Volume 5531 of Lecture Notes in Computer Science. Springer Berlin - Heidelberg (2009) 253–262
11. Zheng, F., Li, H.: Arcrowd-a tangible interface for interactive crowd simulation. In: Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on. (oct. 2010) 287 –288
12. Kwon, T., Lee, K.H., Lee, J., Takahashi, S.: Group motion editing. ACM Trans. Graph. **27** (August 2008) 80:1–80:8
13. Jin, X., Wang, C.C.L., Huang, S., Xu, J.: Interactive control of real-time crowd navigation in virtual environment. In: Proceedings of the 2007 ACM symposium on Virtual reality software and technology. VRST '07, New York, NY, USA, ACM (2007) 109–112
14. Patil, S., van den Berg, J., Curtis, S., Lin, M.C., Manocha, D.: Directing crowd simulations using navigation fields. IEEE Transactions on Visualization and Computer Graphics **99**(RapidPosts) (2010)
15. Alexander, B.: Flow fields for movement and obstacle avoidance. In: AI Game Programming Wisdom 3. Steve Rabin (2006)
16. Rabin, S.: Movement and Pathfinding. In: AI Game Programming Wisdom 4. Course Technology, Cengage Learning. (2008) 59–203
17. Reynolds, C.: Big fast crowds on ps3. In: Proceedings of the 2006 ACM SIGGRAPH symposium on Videogames, ACM (2006) 113–121