

# Co-Fields: Towards a Unifying Approach to the Engineering of Swarm Intelligent Systems

Marco Mamei<sup>1</sup>, Franco Zambonelli<sup>2</sup>, Letizia Leonardi<sup>1</sup>

1 - Dipartimento di Ingegneria dell'Informazione – Univ. di Modena e Reggio Emilia  
Via Vignolese 905 - 41100 Modena – ITALY

2- Dipartimento di Scienze e Metodi dell'Ingegneria – Univ. di Modena e Reggio Emilia  
Via Allegri 13 - 42100 Modena - ITALY

**Abstract.** Complex software systems, as currently engineered, are brittle and fragile. New ideas and a new set of engineering principles are needed to effectively build flexible, robust, evolvable software systems, able to cope with the dynamics of modern execution scenarios. Swarm intelligence – in which the paths to problem solving emerge as the result of interactions between simple autonomous components (agents or ants) and between them and the environment – appears a very promising approach. However, the variety of swarm-based approaches that have been proposed so far still lacks a common modeling and engineering methodology. In the attempt to overcome this problem, this paper presents a general coordination methodology in which swarm's components are simply driven by abstract computational force fields (*Co-Fields*), generated either by agents, or by the environment, or by both. By having agents be driven in their activities by such fields, globally coordinated behaviors can naturally emerge. This model can provide a unifying abstraction for swarm intelligent systems and it can also be exploited to formalize these systems in terms of dynamical systems whose behavior is described via differential equations. Several example of swarm systems modeled with Co-Fields are presented to support our thesis.

## 1 Introduction

In the last few years, the research of radically new approaches to software engineering and computing has witnessed a great momentum. These efforts are well justified by the state of trouble of present day computer science. Actual software engineering practices, based on structured design, lead to brittle and fragile systems [ZamP02, Sus99]. While such practices can be acceptable when engineering software systems to be dived in closed and static scenarios, it becomes definitely unsuitable for those systems whose execution has to deal with dynamic and open environments, as it is the case of pervasive [Est02], mobile [KahKP00], and wide-area Internet applications [CabLZ02]. There, dynamic and openness call for systems architectures capable of supporting both partial failures and dynamic unsupervised re-organization of interaction patterns.

To face this situation, several researchers turned their attention to engineering approaches that take inspiration from the collective behavior of social animals, e.g., ants. From the engineering point of view it is important to observe that animals solve a wide variety of different problems (e.g., finding food, use shortest paths to reach the nest, organize task division) in a very robust and flexible way. Robustness arises from the fact that the colony has the ability to achieve its goal even when

some individuals die or fail to perform their task; flexibility arises from the fact that the patterns of interactions between the individuals are not fixed by contract and can be instead dynamically re-shaped to self-adapt to changing environments. Following other authors [BonT00], we refer to those kind of systems as swarm intelligent systems, to stress the fact that their features and capabilities are not embedded in the single components of the system, but emerge by the coordinated activities of a swarm of individuals.

The features of swarm intelligent systems attract more and more software engineering researchers [ParB02], due to the fact that the daily problems solved by groups of animals have direct counterparts in engineering and computer science: finding food strategies can be exploited in an information retrieval context; the strategies used to find the shortest path connecting two locations can be exploited in routing algorithms for telecommunication and computer networks; mechanisms for ant's division of labor can be exploited in manufacturing or workflow management scenarios. Therefore, it is no surprise that several works [Par97, Bon99, BonT00, Joh01, Ken01] describe peculiar applications of swarm intelligence in a variety of areas. Just to cite a few recent examples, the Anthill system exploits swarm intelligence to perform information search in a wide-area peer-to-peer system [BabBM02], while collective coordinated behaviors of simple components have been used to control the shape of modular robots [Spe02].

While the opportunity to exploit these new concepts has already been spotted, the next challenge is it to enabling their exploitation in a systematic and engineered ways. In our opinion, a possible obstacle to the widespread diffusion of these methodologies is the lack of a common and general framework in which all these swarm intelligent systems could fit.

The aim of this article is to present a model able to provide a unifying abstraction for a large class of swarm intelligent systems. This model is based on the physically inspired concept of computational fields (Co-Fields). Co-Fields are distributed data structures spread in some space, abstracting some features of the world perceived by the agents, and influencing agents' actions so as to implicitly encode coordination tasks. The key advantage of Co-Fields is that it enables to represent in a simple yet effective way both different types of "environments" in which agents can live and different coordination tasks they have to achieve. This makes it possible to adopt the same basic approach to model different classes of swarm intelligent systems. In addition, the physical inspiration also allows to formally treating Co-Fields coordinated systems in terms of dynamical systems [Par98, Par01], by integrating the differential equations governing their behavior. In the future we think that very interesting results could be obtained by applying classic dynamical system theorems and results to this mathematical description, like for example detecting the system's attractors and their basin. However, up to now, we used it as a very effective fast prototyping tool: by solving the differential equations it is possible to quickly analyze the behavior of the system, tuning coefficients and make experiments.

The paper is structured as follows. Section 2 describes the Co-Fields model in general. Section 3 presents different classic examples of swarm intelligence and shows how they can all be modeled in terms of the Co-Fields model. Section 4 selects one of the previous examples and shows how it is possible to describe it using a dynamical systems' formalism. Section 5 concludes the paper and presents our future work.

## 2 The Co-Fields Model

The Co-Fields model provides a modeling framework to design a multi agent system (MAS). Its primary focus is to consider and model the MAS as a “whole”, in which agents achieve their goal not because of their single capabilities, but because they are part of an self-organized system that leads them to the goal achievement [ParB02]; the fact that the goals are accomplished is not a merit of the single agents, but of the system as a whole.

The Co-Fields model can be schematized in the following four points:

1. The environment is represented and abstracted by fields, spread by agents and by the environment itself. These fields convey some useful information for the agents’ coordination tasks and provide agents with strong coordination-task-tailored context awareness.
2. The coordination policy is realized by letting the agents to move following the “waveform” of these fields.
3. Environment dynamics and agents’ movements induce changes in the fields’ surface, composing a feedback cycle that influences agents’ movement (point 2).
4. This feedback cycle lets the system (agents and environment) to auto-organize, so that the coordination task is finally achieved.

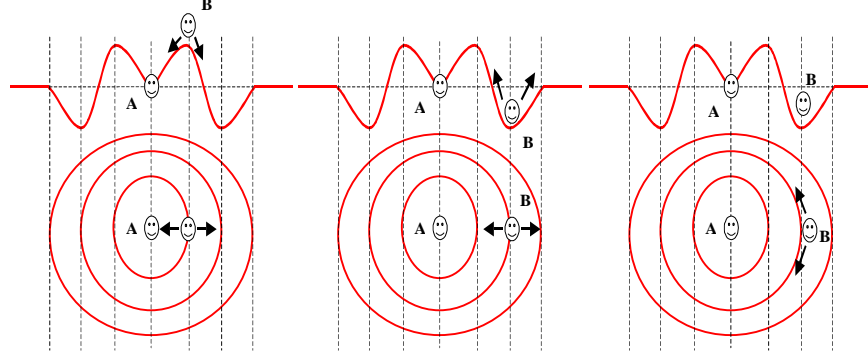
More in detail, a field can be defined as a distributed data structure composed by a unique identifier, a value (representing the field magnitude in that particular point), and a propagation rule. Fields can be generated by the agents or by the environment, and are propagated through the *space* as specified by their propagation rule, which thus determines the final shape of the field surface. Fields can be static or dynamic, basically a field is static if once propagated its magnitude does not change over time; it is dynamic if its magnitude does. A field can be dynamic because for example its source moves and the field, with some propagation delay, changes accordingly its magnitude, or because for example its propagation rule it is designed to remain active and to change field value after some time. In our model agents simply compute a sort of combination of the fields they perceive (e.g. a linear combination). The result of this combination is itself a field, that we will call the agent’s *coordination field*. Agents are forced to follow (deterministically or with some probability) the shape of their *coordination field*, like if they were walking upon the *coordination field* associated surface. Basically their actions will be based on following downhill the decrease of the *coordination field*, on following its increase uphill, or on following one of its equipotential lines (see Figure 1).

Complex movements are achieved not because of the agent will, but because of dynamic re-shaping of this surface. This is a strong point that will enable us to completely treat these field-based software systems as dynamical systems. In fact, having an analytical description of the coordination field, as will be shown in section 4, is rather easy to write down a set of differential equations that can mathematically express the bond between the agents’ movement and the gradient of the field’s surface, and thus govern and describe the system’s behavior.

Almost all the coordination tasks considered in this paper deal with agents’ movements and so the *coordination field* can be considered spread in the physical space and following the *coordination field* surface means moving from one place to another. However, the approach is more general and a *coordination field* spread in a more abstract space could encode coordination tasks that do not deal with some physical movements, but with other kind of actions. In these cases agents would follow its *coordination field*, not by moving from one place to another, but by

making other kind of actions. An example of this more abstract encoding will be presented at the end of the next section.

In this paper, we do not face implementation issue. However, we point out that Co-Fields can potentially be implemented, as an overlay network, on any middleware providing basic support for data storing, communication and event-notification.



**Fig 1.** Agent B following the surface of its *coordination field* which coincides with the field generated by agent A; **(left)** the agent B follows downhill the decrease of its *coordination field*; **(center)** the agent B follows uphill the increase of its *coordination field*; **(right)** the agent B follows an equipotential line of its *coordination field*.

### 3 Classic Swarm Intelligence Examples

In this section we are going to briefly survey different swarm intelligent coordination policies, mainly taken from [Par97, Bon99, Kenn01] and we will show how can they be modeled in terms of the Co-Fields model. This will serve both to justify the claim of unifying approach of this paper and to clarify the Co-Fields model itself.

#### 3.1 Wolves: surrounding a prey

To capture a moose, a pack of wolves have to act in a coordinated way, surrounding the prey. Their coordinated behavior can be explained by means of swarm intelligence without assuming long-range communication mechanisms or complex intelligent strategies [Par97, Kenn01]. Wolves simply hunt for the moose trying to maintain a suitable distance from other wolves. Simulations of this simple strategy, with a moose that simply try to escape by moving farthest away from the nearest wolf, shows that it is a viable and successful solution for wolves to surround and to capture the prey.

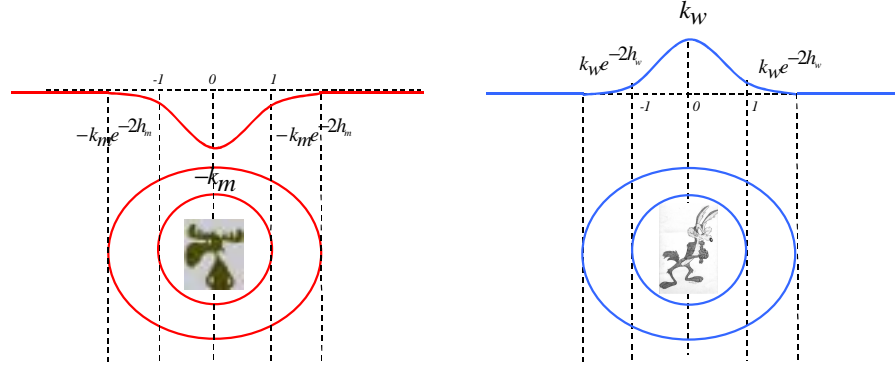
To model such behavior with Co-Field, we can imagine that the moose and the wolves generates the following fields, to be updated in real-time depending on agents' movements: a moose at the coordinates  $(X_m, Y_m)$ , generates a *moose's field* described by the following equation (see Figure 2 left):

$$moose(x, y, t) = -k_m e^{-h_m((x-X_m)^2 + (y-Y_m)^2)} \quad k_m, h_m > 0$$

Analogously, the wolf  $i$  at the coordinates  $(X_w^i, Y_w^i)$ , generates a *wolf's field* whose equation is (see Figure 2 right):

$$wolf_i(x, y, t) = k_w e^{-h_w \left( (x - X_w^i)^2 + (y - Y_w^i)^2 \right)} \quad k_w, h_w > 0$$

The contribution of this analytical description will be clear in section 4, when we will derive the differential equations describing the Co-Fields model in general, and we will apply those equations to this mathematical description of the fields' surfaces to derive moose's and wolves' behavior.



**Fig. 2.** Moose's (**left**) and Wolf's (**right**) generated field

Now, if we consider the moose's *coordination field* consisting in the linear combination of all the wolves' fields:

$$coord_{moose}(x, y, t) = \sum_{i=1}^n wolf_i(x, y, t)$$

by following the decrease of the resulting field, the moose runs away from wolves. Similarly, if we consider each wolf's *coordination field* consisting in the linear combination between the moose's field and all the other wolves' fields:

$$coord_{wolf_i}(x, y, t) = moose(x, y, t) + \sum_{j=1, j \neq i}^n wolf_j(x, y, t)$$

a wolf is directed towards the moose, but staying away from other wolves. It is clear that this simple description is perfectly analogous to the description based on distances and, by a proper tuning of the fields' coefficients, can lead to the same results.

### 3.2 Birds flocking

Flocks of birds stay together, coordinate turns, and avoid each other, by following a very simple swarm algorithm [Par97, Kenn01]. Similar problems happen in air-traffic control and convoys of ships and they could be possibly addressed by using similar methods. The coordinated behavior of flocks can be explained by assuming that each bird tries to maintain a specified separation from the nearest birds and to match nearby birds' velocity. The flock is a self-constraining structure in which each entity's individual action simultaneously responds to and changes the overall

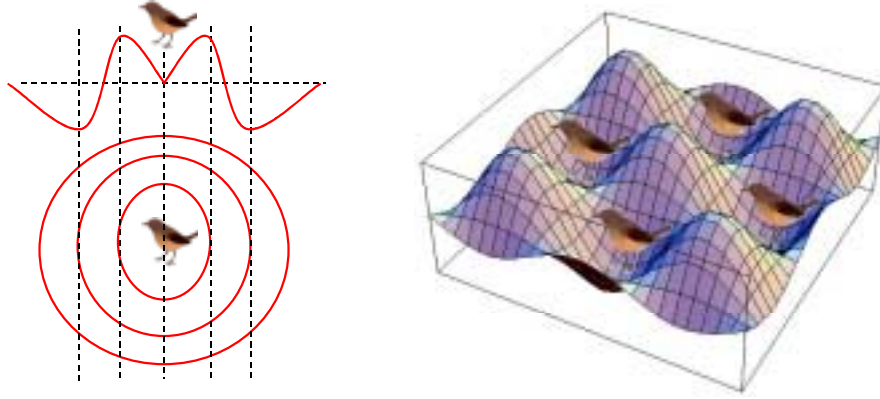
structure of the flock. Although each bird senses only the movements of its nearest peers, its responses to these movements propagate to others, so that the system as a whole exhibits global coordination.

To model this strategy under the Co-Fields modeling framework, we can imagine that each bird generates a *FLOCK* field like the one showed in figure 3 (left), which can be described by the following simple equation:

$$d = \sqrt{(x - X_p^i)^2 + (y - Y_p^i)^2}$$

$$FLOCK_i(x, y, t) = d^4 - 2a^2 \cdot d^2$$

This field is updated in real time to match the bird's movements. Now, saying that birds have to stay a specified distances from each other is equivalent to saying that birds has to follow the decrease of other birds' generated fields (see Figure 3 right). In fact the shape of the field constrains birds to stay close each other in an almost regular grid, the movement of a bird and the consequent change in the field it generates force other birds to move as well.

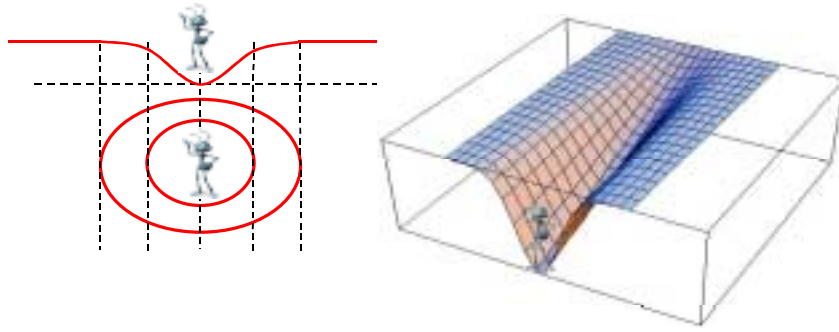


**Fig. 3.** Flocking field (**left**) Flocking birds (**right**)

### 3.3 Ant Foraging

Ants construct a network of paths that connects their nest with available food sources [Par97, Bon99]. This global structure emerges from the simple actions of the individual ants that do not even know they are involved in this distributed building process. Several authors spotted the possibility to exploit the technique used by ants in this process in a routing or in an information retrieval context. Basically each ant that forages for food is able to produce two kinds of pheromones (scent markers that many insects generate). The home-pheromone is produced after leaving the anthill wandering looking for food. The food-pheromone is produced when the ant goes back to the anthill after some food has been found. Ants wander randomly following the food pheromone when looking for food, following the home pheromone when bringing food back to the anthill. The overall system behavior is based on the fact that, pheromone tracks deployed by an ant can be exploited by the same or by other ants in the future. The natural tendency of the pheromones to evaporate if not reinforced, allows the pheromone network to remain up-to-dated. So when a food source is extinguished the corresponding pheromone trail disappears, because it is no longer used and reinforced.

To describe the ants' foraging strategy under the Co-Fields model, we can imagine that the environment is abstracted (i.e. perceived by ants) by means of two (initially flat) fields. These two fields, which we will call the home and the food fields, are generated and spread by the environment itself. The environment reacts to ants' movements by wrinkling the fields' surface, while ants' movements are affected by the "waveform" of the field. This feedback cycle constitutes the key to let the system auto-organize. The algorithm followed by ants can in fact be restated by supposing that each ant wanders, avoiding obstacles, following (probabilistically) the decrease of the food field when it is looking for food, following (probabilistically) the decrease of the home field when it is bringing food back to the anthill. Then to close successfully the feedback cycle we will simply imagine that the environment reacts to the ants' presence, by locally wrinkling the home field surface in correspondence of the points in which ants looking for food are located and it wrinkles the food field surface in the points in which ants carrying food are located. The form of the wrinkle is depicted in figure 4 (left). Moreover we will suppose that the environment is able to control the deepness of the wrinkle, so that each new (or renewed) wrinkle is deeper than all the wrinkles in its neighborhood. Following these principles, ants' movement creates a network of channels in the fields' surfaces. In the food-field surface these channels descend from the anthill to the food sources, in the home-field surface these channels descend from the food sources to the anthill. In figure 4 (right) a channel created by an ant movement is shown. Of course this is a pure abstraction, in no way a natural environment can provide the described capabilities. However this does not matter for our purposes, in fact we just want to demonstrate that the Co-Fields model can abstract these phenomena, and then use this model to build software systems in an artificial environment, where these functionalities can easily be gathered.



**Fig. 4.** A wrinkle induced on a field's surface by an ant presence (**left**). The channel in the field surface created by the ant movement (**right**)

Finally, to accustom for pheromone evaporation, we will imagine that each of the fields' surfaces has some form of memory and it reshapes to its original flat form if untouched. In particular this can be formally stated by saying that a field's wrinkle at coordinates  $(X_w, Y_w)$  created a time  $t_0$  can be mathematically described by the following function:

$$wrinkle(x, y, t) = -k(t - t_0)e^{-h((x - X_w)^2 + (y - Y_w)^2)} \quad k(t), h > 0$$

Where  $k(t)$  is a function that goes to zero as  $t$  increases, while  $h$  is a constant that expresses the fact the wrinkle is only a local deformation.

In general this kind of analytical description is not only useful to state these concepts formally, but it can be applied to provide a dynamical system description of the model. In section 4 this procedure will be applied to the example discussed in section 3.3, however the results can be easily applied also to this case.

### 3.4 Ant Labor Division and Task Succession

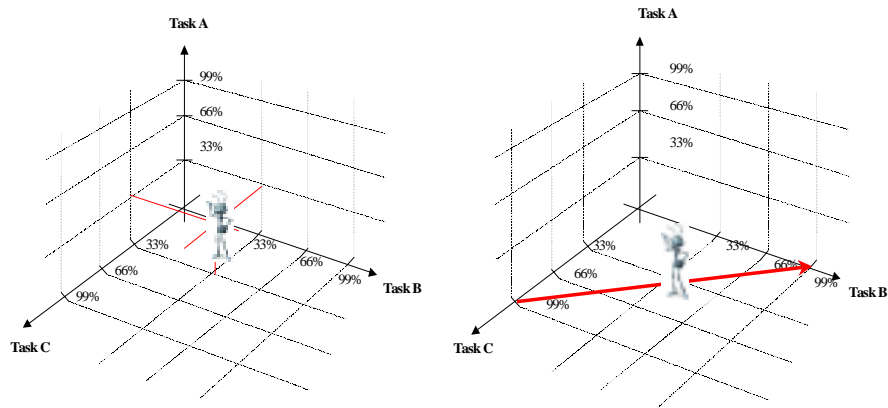
In social insects, different activities are often performed simultaneously by specialized individuals; this phenomenon is called division of labor. A key feature of division of labor is plasticity: the ratios of workers performing different tasks can vary in response to internal perturbations or external challenges. A simple model, which relies on response threshold, can explain how ants achieve flexibility and specialization in labor division [Bon99, Kenn01]. Each individual has a response threshold for every task, it engages in task performance when the level of the task-associated stimuli exceeds its threshold, it drops a task when the task associated stimuli falls under another threshold. In this way everyone adjust its duties according to the colony's needs. Moreover, by performing a certain task individuals' task associated threshold decrease. This simple strategy is the key for specialization: the more one individual performed a task in the past, the more likely the same individual will perform the task in the future.

This example is particularly interesting to be modeled with Co-Fields, because it involves fields propagated in a space, which is not the physical space in which ants are embedded.

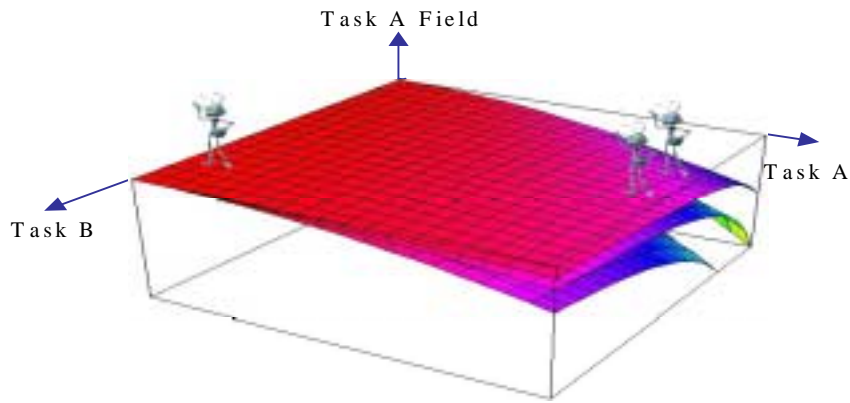
To model the above described coordination task within the Co-Fields approach, we can imagine that there exists a virtual space, separated by the physical one, in which ants are embedded. This is a multi-dimensional space containing one dimension for each of the possible tasks an ant may be involved. If an ant can be involved just in three tasks, let's say: Task A, Task B, Task C, then this space will be the one depicted in figure 5 (left). An ant is placed within the space depending on its duties: we can suppose that each axis measures the fraction of the time, an ant performs that particular task. So that the ant in figure 5 (left) performs Task A for the 33% of its time, Task B for the 33% of its time and Task C for the 33% of its time. Of course, in general, every ant is constrained to the subspace  $\sum_{x_i \in Tasks} x_i \leq 100\%$ . An ant can move in this space, but its movement does not

correspond to an actual movement in its physical space, but in a change in the ant's duties. So for example the movement depicted in figure 5 (right), represent an agent that gradually stops doing Task C and starts doing Task B. Of course fields can be spread and sensed by ants also in this space. In particular the environment generates fields encoding the stimuli that encourages ants in performing a task. Basically the shape of these fields is almost like a steep flat surface, decreasing towards the direction of the associated axis. The more the task achievement is urgent, the more the environment increases the steepness of the field surface. In figure 6, three different Task A's associated fields are shown. The more the surface get steep, the more Task A's achievement becomes urgent.





**Fig. 5.** Diagram to represent task space (**left**). Movement in the task space (**right**)



**Fig. 6.** Task-A stimuli associated fields. The more the Task A's achievement becomes urgent, the more the overall surface gets steep.

Ants' algorithm can thus be restated by means of the following actions: each ant evaluates a combination of the sensed fields, by considering only those fields whose steepness where the ant is located overcomes a certain threshold. Then the ant follows downhill the field obtained. As the ants move along the Task space the field's surface tend to get flattened, because the associated tasks are achieved. Ants' duties get stabilized in a suitable configuration, until new stimuli and thus fields' surfaces' deformations appear. As depicted in figure 6, a task-associated field's surface is not actually a steep flat surface, but its steepness increases both in the direction of the associated axis and both in the proximity of the associated axis itself. The reason for this non-linear shape is to enforce specialization. In fact, on the one hand by increasing the steepness towards the direction of the associated axis, the more an ant is placed towards that direction (i.e. the more it is performing that task), the more it will easy that the ant will perform the task in the future because the field's steepness is there particularly high. On the other hand the increase of the steepness towards the axis itself (i.e. towards the zero) rends easier that "unemployed" ants, rather than already fully committed ants, engage the task.

## 4 A Dynamical Description of Swarm Intelligence

In this section we are going to present a dynamical system description of the Co-Fields model, based on the differential equations that govern the system behavior.

The Co-Fields model suits really well to this kind of description, because individuals' behavior can be assimilated to the one of a ball rolling on a surface (the *coordination field* surface). It is thus rather easy to see that if we consider the individual  $i$ , and we denote its coordinates in a particular space as  $(x_1^i(t), x_2^i(t), \dots, x_n^i(t))$  and its *coordination field* as  $coord_i(X_1, X_2, \dots, X_n, t)$ . Then the differential equations that govern  $i$  behavior are in the form:

$$\frac{dx_j^i}{dt} = \pm v \cdot \frac{\partial coord_i(X_1, X_2, \dots, X_n, t)}{\partial X_j} \quad j = 1, 2, \dots, n$$

Where  $v$  expresses the "speed" of individual  $i$ , while the sign at the right of the equals is decided by whether  $i$  follows the increase or the decrease of the *coordination field*. This is because the direction of the gradient of the individual's *coordination field*, evaluated towards the spatial coordinates, point to the direction in which the *coordination field* increase. So the individual  $i$  will follow this gradient or will go in the opposite direction depending on if it wants to follow the increase or the decrease of its *coordination field*.

In a similar way it is possible to model those cases in which  $i$  follows the equipotential lines of the fields.

To clarify the above equations and to describe a concrete example, we are going to write the differential equations that govern the moose escape and the wolves surrounding strategy. The moose *coordination field*, following section 3.1, is given by the sum (linear combination with all the coefficients equal to 1) of the wolves' fields (let's suppose  $n$  wolves).

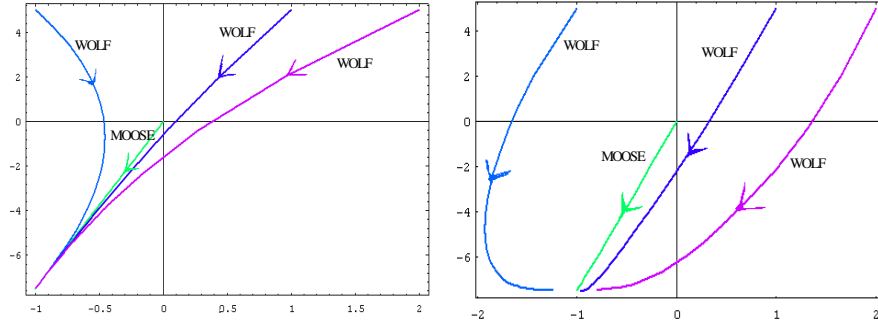
$$coord_{moose}(x, y, t) = \sum_{i=1}^n wolf_i(x, y, t)$$

While the wolf  $i$  *coordination field* is given by the sum (linear combination with all the coefficients equal to 1) of the moose's field (that attracts the wolf) and all the others wolves' fields (that repulse the wolf).

$$coord_{wolf_i}(x, y, t) = moose(x, y, t) + \sum_{j=1, j \neq i}^n wolf_j(x, y, t)$$

Now writing the differential equations is just a matter of substituting these *coordination fields* with the fields' equations described in 3.3 and to use the differential equations at the beginning of this section, in the case of individuals following the decrease of the *coordination field*. Because of space limit, we will not report the result of this substitution, however the system turns out to be described by a set of strictly coupled non-linear differential equations (because of the exponentials in the fields' functions in 3.3). Of course it is not easy to solve these differential equations analytically and to prove the soundness of the presented approach we integrated numerically, with the program Mathematica [Math], the differential equations in the case of a system composed by one moose and three wolves. The results are displayed in the following figure (see Figure 7), which shows a common x-y plane with the trajectories of the elements of the system (i.e. the solutions of  $(x_i(t), y_i(t))$  evaluated for a certain time interval). It is worth noting

that solving these equations numerically is a very effective way to simulate the system, and it can be regarded as a very easy and powerful tool to make experiments and tuning coefficients.



**Fig. 7.** (left) Wolves do not care about other wolves' fields and thus they are not able to surround the moose. (right) Wolves sensing other wolves' fields are able to surround the moose.

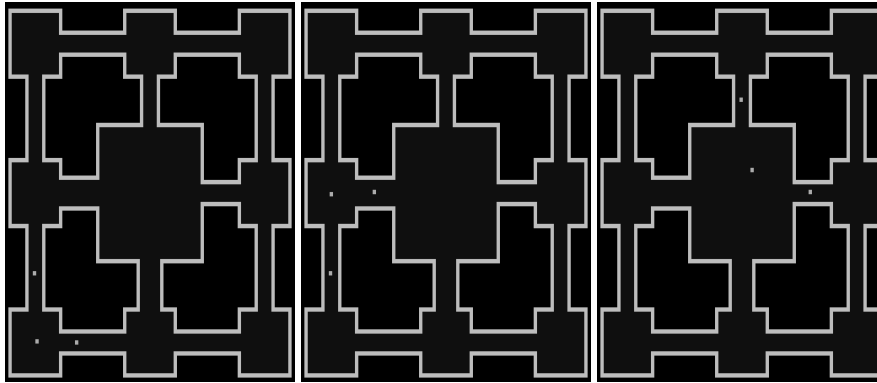
## 5 Conclusions, Current and Future Works

This paper presents a general modeling framework rooted on a field-based model that can provide a unifying abstraction for swarm intelligent systems. This abstraction has been also exploited to formalize some of these systems in terms of dynamical systems whose behavior is described in terms of differential equations. The numerical solutions of these equations provide a valuable tool to make experiments on the studied systems.

In our current work we are applying the Co-Fields model to concrete software systems. We think in fact that this model presents valuable features that could improve current software engineering best practices. In particular, we are dealing with those applications in which a group of autonomous components (i.e. agents), running on some mobile computing devices, have to coordinate their respective movements (or the movements of the users carrying them on). The goals of this coordination can be various: letting the agents to meet somewhere, to move avoiding traffic jams, to distribute themselves accordingly to a particular geometry, etc. The first results of this work have been the development of a simulation in which a field-based application guides the visitors and the clerks of a museum towards [MamLZ02]. In Figure 8, we sketch the results of such a simulation when applied to the flocking problem, and visualizing the coordinated movements of a set of security guards within the museum. We also emphasize that in Co-Fields, as in other swarm systems, adaptivity and self-organization is guaranteed: without any change to the agents' code, the security guards can continue move in a coordinated way even when new security guards arrive or when security guards have to take care of a totally different museum.

Our future work will proceed towards two main directions: on the one hand we will try to extend the Co-Fields model and to better formalize it. Our perception is that particularly significant results can be obtained by generalizing the use of the Co-Fields model to tasks that do not deal with some physical movements, but with other kind of actions, as we did in the example of ant's division of labor. On the other hand, we would like to develop a proper infrastructure to support the Co-Fields model. To use this model in a real computing scenario is in fact necessary to have a

proper middleware infrastructure able to manage the distributed data structures associated to fields. Finally it would be very interesting to explore other approaches like Amorphous Computing [Abe00], Dissipative Cellular Automata DCA [RolZ02] and Artificial Neural Networks [Joh01] to see if and how the Co-Fields model can be combined somehow with them.



**Fig. 8. (from let to right).** Different stages in the Co-Fields guided movement of a group of agents through the virtual building of a museum. Agents follows the

## References

- [Abe00] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. Knight, R. Nagpal, E. Rauch, G. Sussman and R. Weiss, "Amorphous Computing", *Communications of the ACM*, 43(5), May 2000.
- [BabMM02] O. Babaoglu, H. Meling, A. Montresor, "Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems", 22<sup>nd</sup> International Conference on Distributed Computing Systems, Vienna (A), July 2002, to appear.
- [Bon99] E. Bonabeau, M. Dorigo, G. Theraulaz, "Swarm Intelligence", Oxford University Press, 1999.
- [BonT00] E. Bonabeau, G. Theraulaz, "Swarm smarts", *Scientific American*: 72-79, March 2000.
- [CabLZ02] G. Cabri, L. Leonardi, F. Zambonelli, "Engineering Mobile Agent Applications via Context-Dependent Coordination", *IEEE Transactions on Software Engineering*, 28(9), September 2002.
- [Joh01] S. Johnson, "Emergence", Scribner, 2001.
- [KahKP00] J. M Kahn, R. H. Katz, K. S. J. Pister, "Emerging Challenges: Mobile Networking for Smart Dust", *Journal of Communications and Networks*, 2(3):188-196, Sept. 2000.
- [Ken01] J. Kennedy, R.C. Eberhart, Y. Shi, "Swarm Intelligence", Morgan Kaufmann, 2001.
- [MamLZ02] M. Mamei, L. Leonardi, M. Mahan, F. Zambonelli, "Coordinating Mobility in a Pervasive Computing Scenario with Co-Fields", 1<sup>st</sup> International Workshop on Mobile Teamwork, IEEE CS Press, Vienna (A), July 2002.
- [Math] Mathematica, <http://www.wolfram.com/products/mathematica>
- [Par97] H.V.D. Parunak, "Go to the Ant: Engineering Principles from Natural Multi-Agent Systems", *Annals of Operations Research* 75:69-101, 1997.

- [Par98] H.V.D. Parunak, "A Dynamical Systems Perspective on Agent-Based Going Concerns", IWMAS, Dedham, MA, USA, 1998.
- [Par01] H.V.D. Parunak, "Entropy and Self-Organization in Multi Agent Systems", in Proceedings of Autonomous Agents, Montreal, Canada, May 2001.
- [ParB02] H. V. Parunak, S. Brueckner, J Sauter, "ERIM's Approach to Fine-Grained Agents", NASA Workshop on Radical Agent Concepts, Greenbelt, MD, USA, Jan. 2002.
- [RolZ02] A. Roli, F. Zambonelli, "Emergent Behaviour in Dissipative Cellular Automata", 5<sup>th</sup> International Conference on Cellular Automata for Research and Industry, Geneva (CH), Sept. 2002, to appear in the LNCS.
- [Spe02] M. Yim, Y. Zhang, D. Duff, "Modular Robots", IEEE Spectrum, Feb. 2002.
- [Suss99] G. Sussman, "Robust Design through Diversity", DARPA/MIT Amorphous Computing Workshop, September 1999.
- [ZamP02] F. Zambonelli, V. Parunak, "Sign of a Revolution in Computer Science and Software Engineering", February 2002, Submitted for Publications.