



Field-based Coordination

Franco Zambonelli & Marco Mamei
March 2005

1



Outline

- Part 1: What is Field-based Coordination
 - Definitions
 - General aspects of field-based coordination
 - Fields, self-organization, and stigmergy
- Part 2: Examples of Field-based Coordination in Nature
 - Gravitational systems
 - Flocking in birds and fishes
 - Wolves surrounding a prey
 - Morphogenesis
 - Modeling Fields
- Part 3: Applications of Field-based Coordination in Distributed Systems Engineering
 - Location-awareness
 - Motion coordination
 - Self-assembly
- Part 4: Programming Field-based Coordination
 - The TOTA Approach
 - Examples of TOTA applications
- Conclusions and Open Issues

2



Part 1

- What is Field-based Coordination

3



What is Field-based Coordination

- Field-based coordination is a physically-inspired indirect form of coordination
 - The space in which “agents” live is spread by sorts of force fields
 - E.g., gravitational fields, electromagnetic fields, chemical fields
 - Agents in their turn can contribute to spreading fields in space
 - E.g., the same as masses contribute to gravitational fields
 - Agents are affected in their activities by the locally perceived fields (type and strength)
 - E.g., the same as masses are attracted by gravitational fields
 - Fields are automatically updated to reflect the actual conditions
 - E.g., the same as gravity changes as planets move

4



What are “Agents” in Field-based Coordination

- Where talking about field-based coordination we will refer to the term “agent” to refer to
 - Any kind of autonomous entity situated in an environment
 - Sensing and effecting the environment via fields
 - Thus indirectly interacting with each other
- Examples
 - Animals
 - Software processes and software agents on a network
 - Mobile devices and sensor networks
 - Humans
 - Robots

5



What is “Environment” in Field-based Coordination

- When talking about “environment” in field-based coordination we generally refer to
 - The “space” in which agents execute
 - The “space” in which fields propagate
 - It is mostly an abstraction that assume actual meaning in actual examples
- Examples
 - A landscape in which animals live and interact
 - A network
 - A building in which humans and robots live

6

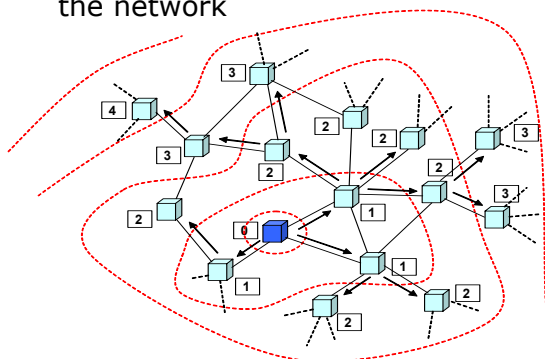
What are Fields?

- Assuming the presence of an environment
 - Fields are “distributed properties” of that environment
 - Describing “something” related to that environment in a distributed way
- Examples
 - The gravitational field is a property of space somewhat describing the structure of masses around
 - The pressure field is a property of the atmosphere describing the density of the air
- More in general, given
 - An metric environment E (i.e., an environment whose points can be somewhat mapped in terms of some system of coordinates $\mathbf{X}$)
 - A field F can be defined as a function mapping a set of value over the coordinates of the environment $V=F(\mathbf{X})$, $\mathbf{X} \in E$

7

Example of a Field over a Network

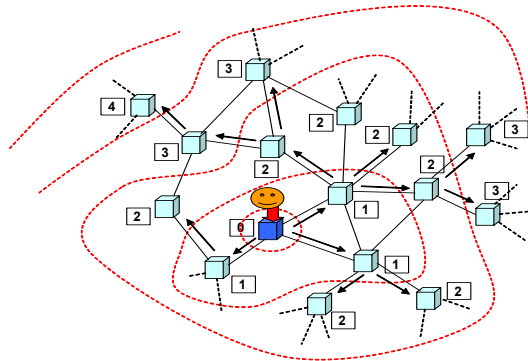
- Given a specific node of a network
 - We can imagine having a field that spread from that node and
 - Propagates over the network with a local value that increases each hop while traveling the network



8

Spreading Fields

- There are several possibilities about spreading fields
 - Fields can be an intrinsic property of space, independent of the presence of agent (e.g. pressure in atmosphere)
 - Fields can be spread by agents, and different agents can spread different types of fields independently (e.g., a cry)
 - A field can be an aggregation of different fields spread by individual agents (e.g., gravity)



9

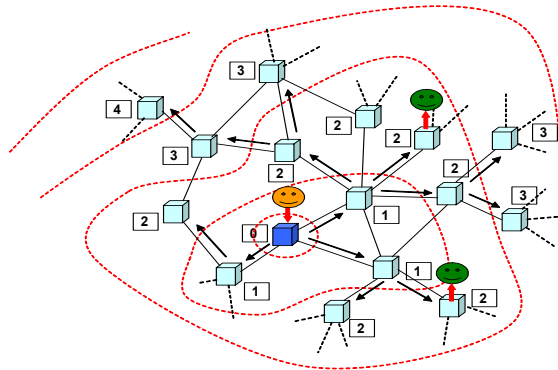
Sensing Fields (1)

- In general, in field-based coordination we assume (as in nature) only strictly local sensing capabilities
 - An agent can perceive the local value of a field, not the overall distributed structure of a field
 - At least, it can perceive the local gradient of the field (i.e., in which direction it grows and how much it grows)
- This is very important in distributed systems because
 - It does not require global communications
 - It promotes scalability (what the agent can sense is independent of the actual "size" of the environment)
 - It promotes decoupling (the agent sense the field but do not care about who or what causes it)

10

Sensing Fields (2)

- The same field, spread by the orange agent and sensed by two green agents
 - At different places
 - And so with different values



11

Fields and Context-Awareness

- To some extent, fields represent a way to spread contextual information over a network
 - The presence of mass in the case of gravitational fields
 - The presence of some data or of some agent doing something in the case of computational fields
 - The presence of an agent at a specific distance in the network
- Clearly, this is a partial information
 - I only know there is an agent somewhere at distance "3"
 - And if I can sense the gradient I can perceive the direction in which it is
- But it is obtained in a very effective way, at very low local costs!

12



Fields as Distributed Data Structures

- To some extent, fields can be considered as a specific type of structured data
 - The structure is distributed
 - The same "information" is distributed on different regions of an environment
 - The information has a meaning that depend on the place on which it is perceived
 - And thus it varies to reflect in different regions the same meaning
- In the example of the network field
 - The meaning could be the distance from an agent
 - And thus the value has to change as the fields propagate farther from the source agent

13



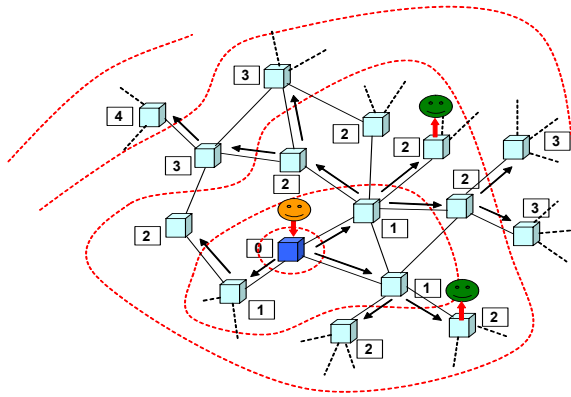
Dynamics of Fields (1)

- Fields are expected to continuously reflect the actual state of what they represent
 - To reflect changes or movements of the originating agents
 - To reflect changes in the environment
- Examples
 - The gravitational fields of a star changes as it explodes or moves
 - The actual gravitational fields in a point of space depend on the position of all nearby masses
 - The structure of the network field should change if the structure of the network changes (due to, e.g., dismissing or mobility of nodes)

14

Dynamics of Fields (2)

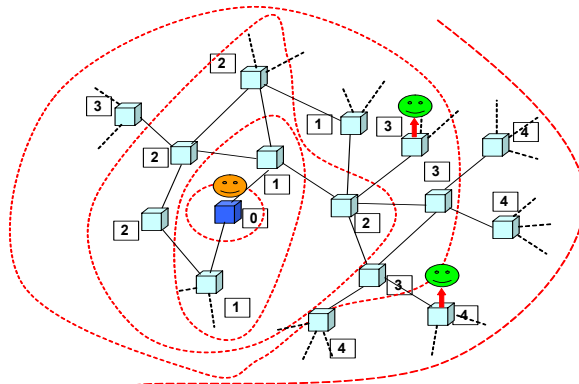
- The node with the source agent moves
- And the field is updated



15

Dynamics of Fields (3)

- The node with the source agent moves
- And the field is updated



16



Fields and Adaptive Self-organization

- When agents rely on the local perception of fields for their activities
 - An agent locally perceives fields propagated by other agents
 - It may consequently change the structure of fields it own propagates
- This imply the intrinsic presence of feedback cycles
 - Which could be positive
 - Or negative
 - And which in any case we already know are the basic ingredients for adaptive self-organization
- Adaptive self-organization with fields
 - Adaptivity because fields changes to reflect changes in the environment (fields represent dynamic contextual information)
 - Self-organization because fields are a form of stigmergic coordination

17



Field-based Coordination and Stigmergy

- Field-based coordination is a specific type of stigmergic coordination
 - It is an indirect form of coordination
 - Fields represents property of the environment
 - Agents can affect the environment by spreading fields
 - Agents are affected by the environment, i.e., by the fields that are in it
- We will see that pheromones and fields are strictly related
 - After all, pheromones can be considered as sorts of "slow" fields
 - Thus, we will be able to implement pheromones with fields
 - Sometimes, the distinction between pheromones and field will be rather blurred

18



Part 2

- Field-based Self-Organization in Nature
 - And field modeling...

19



Gravitation, of Course...

- Well, gravity is field-based coordination
 - Is there self-organization?
- Yes, and indeed gravitational system exhibits all the possible status of dynamical systems
 - Equilibrium
 - All masses collapse into a single mass
 - Periodic Cycles
 - The regular orbiting of a mass around another
 - Self-organization ("edge of chaos")
 - Consider the asteroids belt
 - It is rather "stable" and exhibit an overall organized, persistent, and adaptive, "shape"
 - Still, the movement of asteroids within is by no means periodic of stable
 - It is a system at the edges of chaos
 - Chaos (e.g., clusters of isolated asteroids)
 - Indeed, even a small number of isolated asteroids of similar mass moves around each other in a chaotic way

20

Flocking

- Birds and Fishes, as well as several other species, move in groups in a rather organized and ordered way
 - There is no leader in general
 - There is not agreement
 - Animals per se are not so intelligent
- So how does it work such expression of swarm intelligence?
 - Very similar to humans escaping fires

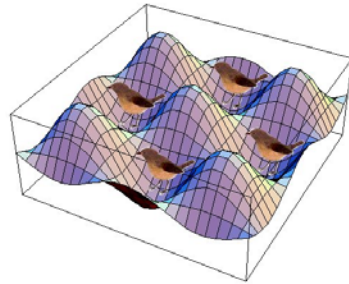
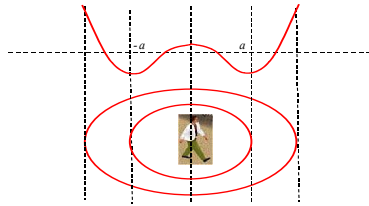


Mechanisms of Flocking

- Two possible explanations, same results
- 1. They do so because they want to stay together
 - possible explaining mechanism for each bird
 - Target closest bird
 - Get at a specific distance from it
 - Match its speed and direction
 - But this assumes a sort of mimicking capabilities in bird
- 2. They do so because in that way their movements are more effective (as a cyclist knows well!)
 - Possible explaining mechanism for each bird
 - Just fly
 - Trying to always stay in the zone of lower pressure
 - This is a mechanism of field-based coordination
 - The pressure field of birds as they fly
 - Influences the fly of other birds
 - Ending up in organized patterns of movement

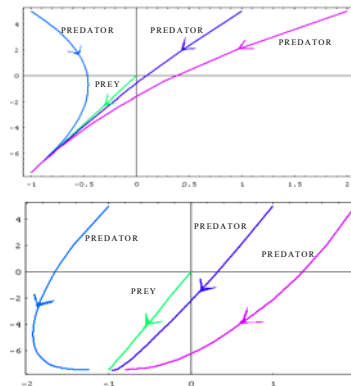
Field-based Flocking

- The pressure field generated by each bird has a form like that:
- A bird close to another bird will try to stay in the minimum
 - i.e., will be attracted by the field
- Overall, when birds get flying together, they will end up in regular grid formations
- See the NetLogo simulation



Sheep Surrounded by Wolves

- When wolves go preying and see a e.g., sheep
 - They do not simply try to capture it in a greedy way
 - Rather they circumvent the prey
 - And leave it not way of escaping
- Is this wolves' intelligence or there are swarm intelligence mechanisms behind?
 - The behavior is isomorphic to that of insects collectively carrying big objects
 - So it is likely to be a swarm intelligence phenomenon



Mechanisms of Surrounding

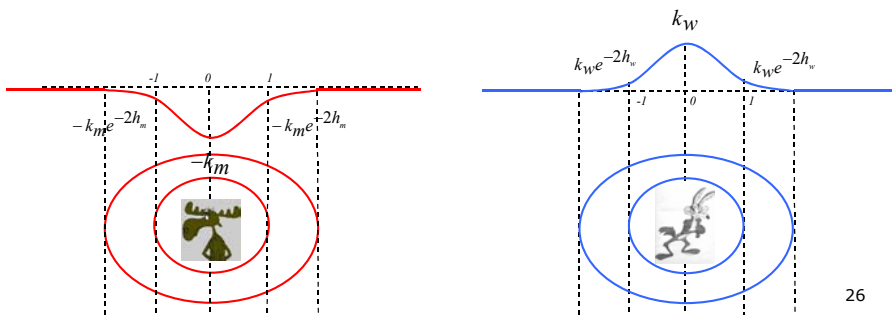
- Very simple indeed
 - Wolves are attracted by the sheep
 - But repel each other (because of competition)
 - The result is surrounding!
- This can be easily modeled in terms of fields
 - Attractive force fields: the wolves tries to reach the minimum of a force field originated by the sheep, and having its minimum by the sheep
 - Repulsive force fields: the wolves tries to reach the minimum of the aggregated force field generated by wolves, and having its minimum at infinity
 - Of course, the sheep fields must dominate!
- Ah, and obviously
 - It is expected that the sheep will try to escape, being repelled by wolves' fields

25

Field-based Surrounding

- The Sheep field and the wolves fields
- The resulting force that drives wolves movement is something like:

$$FORCE = sheep_field - sum(wolves_fields)$$



26

Modeling Field-based Coordination

- In the previous examples, fields act as forces attracting/repelling animals
 - Coordination, in this case, is related to motion and position
 - **MOTION COORDINATION**
 - Fields influence the relative positions and movements of animals in the environment
- In these case, it is possible to model field based coordination in terms of a simple dynamical system in which:
 - Fields are modeled as function of the position of the sources (i.e., of the animals but more in general of the agents)
 - The next position of animals/agents is determined by the composition of the fields that they sense at the current time in their current positions
 - This is simply a system of differential equations...

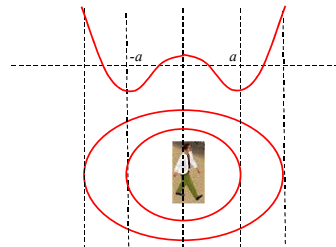
27

Modeling Fields

- Let's express field as a function of time and space (i.e., position of the source agent)
 - E.g. the Flock field

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

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



28

The Coordination Field

- In general, agents do not direct their action based on a single field, but based on a combination of locally perceived field
- If the specific combination of these fields, that we call **coordination field** (*CF*) that determines where the agent will go
- In the case of flocking, any agents tries to reach the minimum of all the locally perceived FLOCK fields
- Thus is operate based on a min combination of local FLOCK fields

$$CF(x_1, x_2, \dots, x_n, t) = f(FIELDa, FIELDb, \dots, FIELDn)$$

$$CF(x, y, t) = \min (FLOCK_i(x, y, t) : i = 1, \dots, n)$$

29

The Evolution Equations (1)

- Thus, the laws governing the movement of an agent i along the x_j direction are in the form below
- Where the spatial derivative of the coordination field expresses the fact that an agent follows the gradient of the coordination field

$$\frac{dx_j^i}{dt} = v \cdot \frac{\partial CF_i(x_1, x_2, \dots, x_n, t)}{\partial x_j} (x_1^i, x_2^i, \dots, x_n^i, t) \quad j = 1, 2, \dots, n$$

30

The Evolution Equations (2)

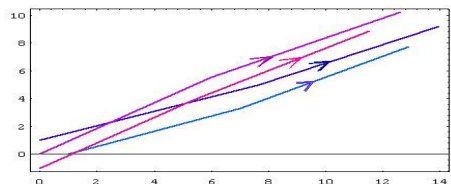
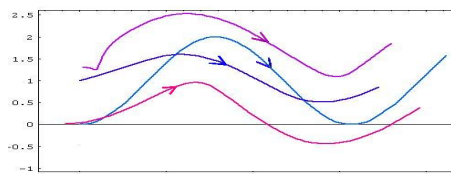
- Thus, for an example in the 2-dimensional case, the evolution of the flocking behavior for an agent i can be described by the following system of differential equations

$$\begin{cases} \frac{dx_i}{dt} = -v \frac{\partial \min(FLOCK_1, FLOCK_2, \dots, FLOCK_n)}{\partial x}(x_i, y_i) \\ \frac{dy_i}{dt} = -v \frac{\partial \min(FLOCK_1, FLOCK_2, \dots, FLOCK_n)}{\partial y}(x_i, y_i) \end{cases}$$

31

Integrating the Equations

- Once the equations are defined
 - We can write them for each agent
 - And initialize them with the initial positions of the agents
 - Obtaining a global set of related differential equations
- The equations can be numerically solved by mathematical software tools to see the evolution of the system (i.e., the trajectories of agents in space)
 - Aside two examples for a flocking system with 4 agents in 2-d
 - With different initial conditions



32

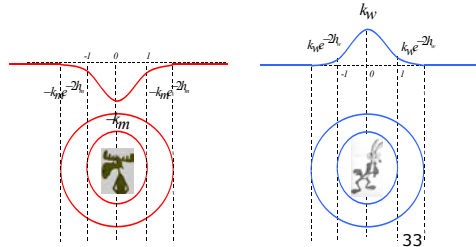
Modeling Surrounding (1)

- The FIELD of wolves (pred)

$$FIELD_i^{pred}(x, y, t) = 1 + k_p - k_p e^{-h_p \left((x - X_p^i)^2 + (y - Y_p^i)^2 \right)}$$

$$k_p, h_p > 0; \quad 1 + k_p - k_p e^{-2h_p} \approx 1$$

- The FIELD of sheep (prey) is simply the opposite of that of wolves...



Modeling Surrounding (2)

- Coordination field of wolves (pred)

$$CF_i^{pred}(x, y, t) = FIELD^{prey}(x, y, t) + \sum_{j=1, j \neq i}^n -FIELD_j^{pred}(x, y, t)$$

- Coordination field of sheep (prey)

$$CF^{prey}(x, y, t) = \sum_{i=1}^n -FIELD_i^{pred}(x, y, t)$$

Modeling Surrounding (3)

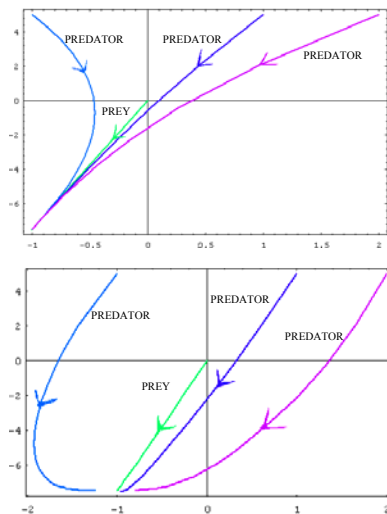
- The overall equations

$$\left\{ \begin{array}{l} \frac{dx^{prey}}{dt} = -v^{prey} \frac{\partial CF^{prey}(x, y, t)}{\partial x} (x^{prey}, y^{prey}) \\ \frac{dy^{prey}}{dt} = -v^{prey} \frac{\partial CF^{prey}(x, y, t)}{\partial y} (x^{prey}, y^{prey}) \\ \frac{dx_i^{pred}}{dt} = -v^{pred} \frac{\partial CF_i^{pred}(x, y, t)}{\partial x} (x_i^{pred}, y_i^{pred}) \quad i = 1, 2, \dots, n \\ \frac{dy_i^{pred}}{dt} = -v^{pred} \frac{\partial CF_i^{pred}(x, y, t)}{\partial y} (x_i^{pred}, y_i^{pred}) \quad i = 1, 2, \dots, n \end{array} \right.$$

35

Modeling Surrounding (4)

- The Integrated equations (compared with the no-surrounding case)





Why Modeling?

- It can be useful to understand, prior to developing a system, and prior to build a simulation, if an approach make sense
 - The mathematical simulation, even if unrealistic, or developed with a limited number of agent, can outline specific problems of a solution
- In any case, this must be coupled with simulation, including a larger number of agents

37



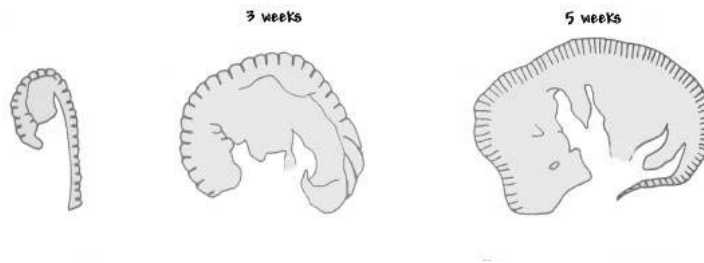
Motion vs. Activity

- So far, in the examples, fields have been used to determine the movement of agents
 - Pure motion coordination
 - The reaction of agents in perceiving fields was simply that of following the gradient of the fields
- However, field-based coordination may be more general
 - The perception of a specific field with a specific value by an agent
 - Can cause it to change its internal state
 - Can make it react to emit other types of fields or to inhibit the propagation of some fields
- In other words, fields can be used to enforce coordination of activities other than of motion
- For example, in morphogenesis...

38

Morphogenesis

- Morphos + Genesis = Shape Birth
 - The process by which something grows into a specific shape
- This is definitely one of the biggest open issues in science
 - How an embryo grows?
 - How cells understand how to differentiate each other?
 - How can they understand where to grow and when to stop?
- What is being understood is that a field-based coordination model is behind all (most of) that



39

Morphogen Gradients = Fields

- Cells emit chemicals
 - Which diffuse from cell to cell
 - As in the example of the field spreading in a network hop by hop
- The behavior of a cells, that is
 - The activation/deactivation of specific genes and – thus – the type of cell it has to become and the way it must reproduce
 - The type of chemicals that the cell, in its turn has to diffuse
- Is determined by the perceived concentration of morphogen gradients
 - The same as in field-based coordination the actions of an agent are determined by its local perception of field

40



Fields vs. Pheromones

- Pheromones are something that gets deposited in an environment
 - And there rest
 - Slowly diffusing
 - And slowly evaporating
- Fields, on the other hand, are something that exists because some entities are propagating them
 - They exists until they gets propagated
 - They immediately change when the conditions of propagation change
- Clearly, it is possible to conceive a field
 - With a very very slow propagation
 - With a very very slow reaction time in changing
- In that case, the field would become a pheromone
- i.e., it is possible to model pheromones in terms of “slow” fields
- In any case, it is important to note that for both fields and pheromones
 - The **environment plays a vital role** in sustaining propagation and updates

41



Part 3

- Applications of Field-based Coordination

42



The Applications are Various...

- Location-awareness
 - As a way to distribute and access contextual, location-dependent, information
 - In pervasive and mobile computing scenarios, as well as in sensor networks
- Motion Coordination
 - For mobile users, robots, or software agents
- Videogames
 - Using fields to drive characters
- Self-assembly
 - As a way to mimic morphogen gradients and produce self-assembly artifacts in which parts can self-differentiate
 - This will be dealt with separately, in another lecture

43



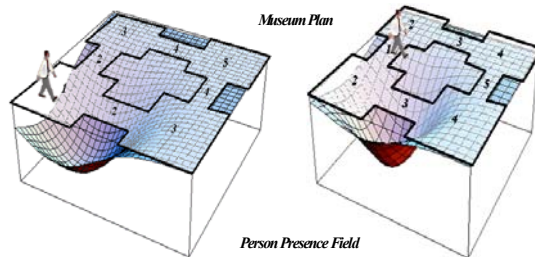
A Scenario: a Big Museum

- A big museum, such as the Louvre
 - Difficult to explore (where the hell is Mona Lisa?)
 - With a structure and topology often varying (due to temporary exhibitions or restructuring)
 - With a lot of information to access about art pieces
- Let us assume that
 - There is a network running in the museum whose topology mimic the topology of the museum (if not, the museum topology can be simply mapped as an overlay over the physical network)
 - This could also be an-hoc wireless network, to avoid the costs and the damages of wiring
 - Each piece of art has an internal device with info about it, integrated in the network
 - Each users and tourist guide has a PDA via which to access the network from the closest access point
- The scenario, in any case, is representative of a wider range of distributed applications...

44

Location-awareness in the Museum

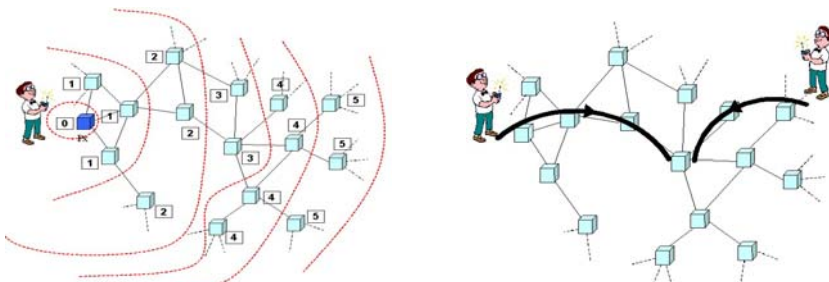
- Fields, propagated in the museum by tourists, or be piece of arts, or by museum guards, can be used to have
 - Tourists discover where a specific piece of art is
 - Tourists and Guards discover where other tourists/guards are
 - Etc.
- To this end, we can assume that each tourist and each guards
 - Diffuse a PERSON PRESENCE field, propagating in the museum network with an hop-increasing value, and having also as a content the ID of the person



45

Discovering Other People

- Based on the PERSON PRESENCE field, any user can simply
 - Perceive the corresponding field (with the specific ID) to understand distance and direction
 - Eventually, decide to follow the field to reach that person





Discovering Pieces of Art (1)

- If each piece of art diffuse a similar field
 - Any person can locally read the location of that piece of art
- When a person is interested and reads the local field of a specific piece of art
 - Can eventually send a message-field to that piece of art to discover further information
 - The message field propagates by following downhill the field of the piece of art, and will reach it as a ball down a surface
 - When the piece of art receive the message field, can react by sending back to the user information

47

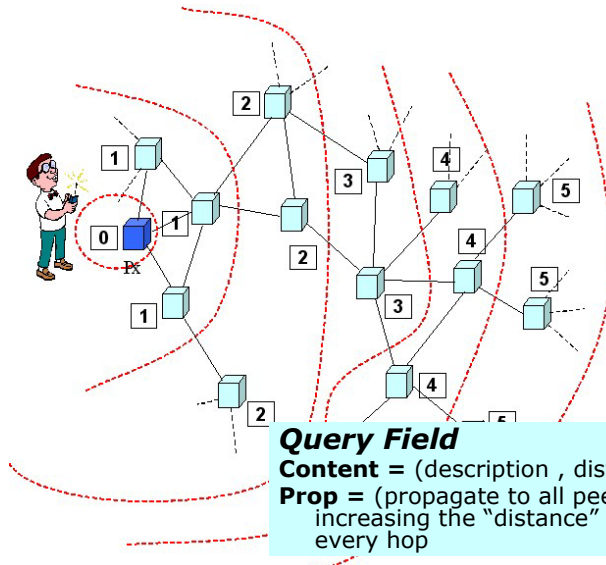


Discovering Pieces of Art (2)

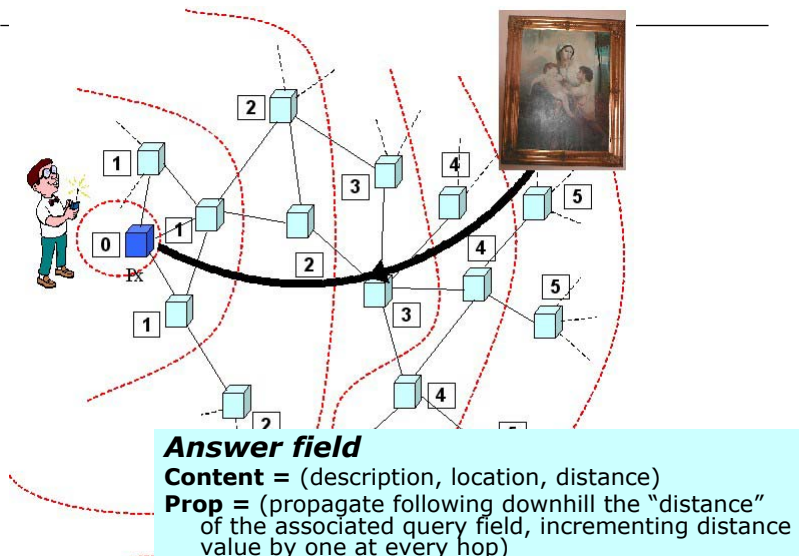
- As an alternative solution
- To avoid pieces of art diffuse their fields in the whole museum, one can think at
 - The pieces of art have no fields diffused
- But can react to "Query fields" diffused by users
 - When a users wants to know something about a piece of art
 - Spread a query field that increase hop-by-hop as it travels
 - The query field reach the specific piece of art
 - Which react by spreading and "Answer field" that follows downhill the query until it reaches back the user

48

Where is Mona Lisa? Propagate a Query Field



Where is Mona Lisa? The Answer Follows Query Field Downhill





Location-dependent Computing

- In general, one can think at
 - Limiting the propagation of fields at specific distances
 - Directing the propagation in specific directions
- In that way, it is possible to
 - Search information in specific locations
 - Avoid being discovered from “too far”
 - Very useful in mobile and ubiquitous computing
- Example, as a tourist in a big town who ask the wireless city network for good chinese restaurants, I want to know about close restaurants, not about all restaurants in other – far – parts of the town

51



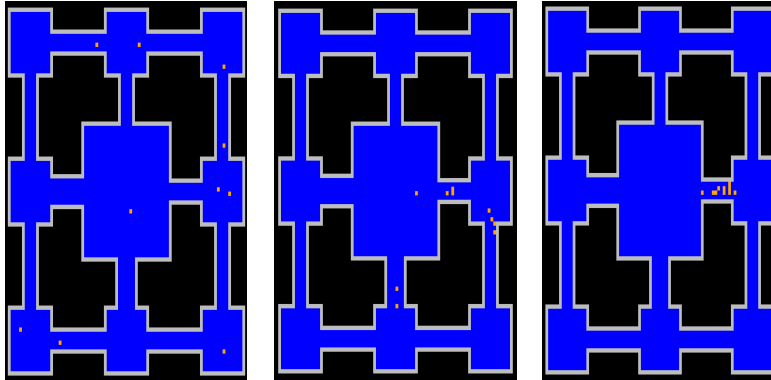
Motion Coordination

- By using the introduced PERSON PRESENCE field, one can think at coordinating the activities of tourists and guards, as they move in the museum
 - Meeting at a specific place, using PRESENCE FIELDS as gravitational fields (the meeting place will end up being the center of gravity, minimizing global efforts for meeting)
 - For guards, finding a children lost in the museum, or a thief using PRESENCE FIELDS to implement
 - Moving in formation (e.g., for guards) using FLOCK fields similar to that of flocking in birds
- What is interesting is that the resulting behavior is extremely adaptive
 - There is no need to know a priori the map of the museum
 - It is the propagation of fields that automatically adapt to the map
 - And enables any form of motion coordination, in an adaptive and environment independent way

52

Examples of Motion Coordination

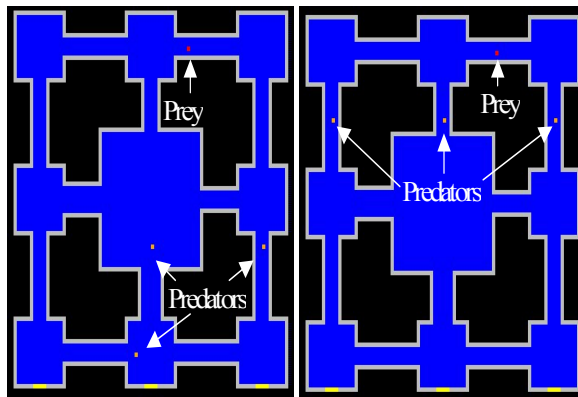
○ Meeting



53

Examples of Motion Coordination

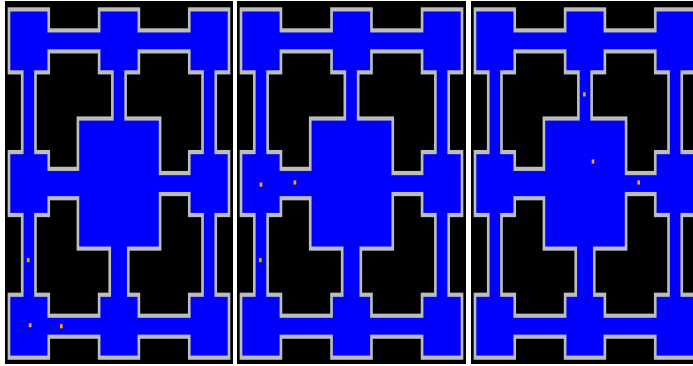
○ Surrounding



54

Examples of Motion Coordination

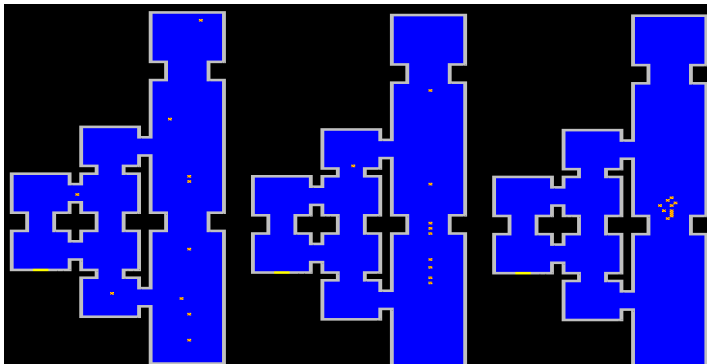
- Moving in Formation



55

Adaptivity of Motion Coordination

- The algorithms work the same independently of the museum map
 - Without changing a bit in the code of agents
 - E.g., for meetings



56

Fields in Videogames

- Field-based approaches can be exploited in videogames to drive the actions of characters
 - In "The Sims" fields drive characters toward what they want ("I am hungry!" → let's follow the Fridge field)
 - In Quake, we have integrated field-based motion coordination as a strategy to improve cooperative fighting in the "bad boys" (Mamei and Zambonelli, 2004)



57

Part 4

- Programming Field-based Coordination

58

What is Required?

- To program and deploy field-based coordinated application we need
- Some **network infrastructure** to be used as the space in which to propagate fields
 - The Internet, an ad-hoc network, a pervasive network, a sensor network
- A fully distributed **middleware (P2P)**, providing the following minimal services
 - A local dataspace where to store the value of fields as they get propagated, and to have application agents access the local values of fields
 - Communication mechanisms, to have the various middleware communicate with each other and enable propagation of fields in the network
 - System-level event-based mechanisms, to understand how the network situation changes and update fields accordingly
 - Application level event-based mechanisms, to have application agents understand when fields arrive and/or when they change

59

The TOTA Approach

- A tuple-based system for field-based coordination in dynamic networks
 - Implemented for PC, PDAs, Lego Robot
 - Mamei and Zambonelli, 2003-2005
- Extends traditional tuple-based coordination models with a model in which
 - tuples are not stored in a single tuple space
 - Tuples, other than a content **C** as usual, can have their own propagation rules **P**
 - The propagation rule can specify how the content of the tuple can change during propagation

$$T=(C, P)$$

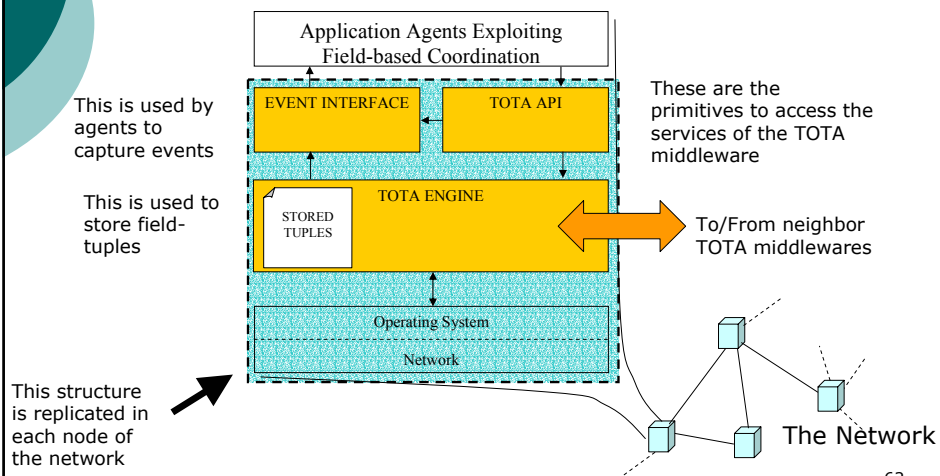
60

TOTA Key Characteristics

- Fields are represented by tuples
 - This enables to include some specific content with fields
 - The programmer can invent and program its own types of fields/tuples
- Tuple libraries are available
 - To exploit with minimal efforts a variety of different types of fields
- Propagation rules are fully programmable
 - This enables programmers to decide how and to which extent a field-tuple should propagate
- Maintenance of the distributed field structure is automatically preserved
 - The middleware automatically react to changes in networks and applications updating the structure of propagated fields/tuples

61

TOTA Middleware Architecture



62

The TOTA API

```
void    inject (TotaTuple tuple);
// inject a tuple-field in the network
// the middleware will automatically propagate it on the basis of
// propagation rules defined withing the tuple itself

Vector  read (Tuple template);
// read local tuples matching the template

Vector  readOneHop (Tuple template);
// read also the tuples of the neighbourhood, to evaluate gradients

Tuple   keyrd (Tuple template);
Vector  keyrdOneHop (Tuple template);

Vector  delete (Tuple template);
// delete a local tuple

void    subscribe (Tuple template, ReactiveComponent comp, String rct);
void    unsubscribe (Tuple template, ReactiveComponent comp);
// tu subscribe/unsubscribe to event occurring in the tuple space
// related to the arrival or change of field-tuples
```

Programming TOTA Applications

- Programming TOTA coordinated applications basically means to
- Programming the tuples
 - i.e. specifying the tuple classes in terms of the type of content they have to include, their propagation rules, and their maintenance rules
- Programming the agents
 - That is, the code in which agents exploit the TOTA API, create tuple instances and propagate them in the network, and read tuples from the local tuples space

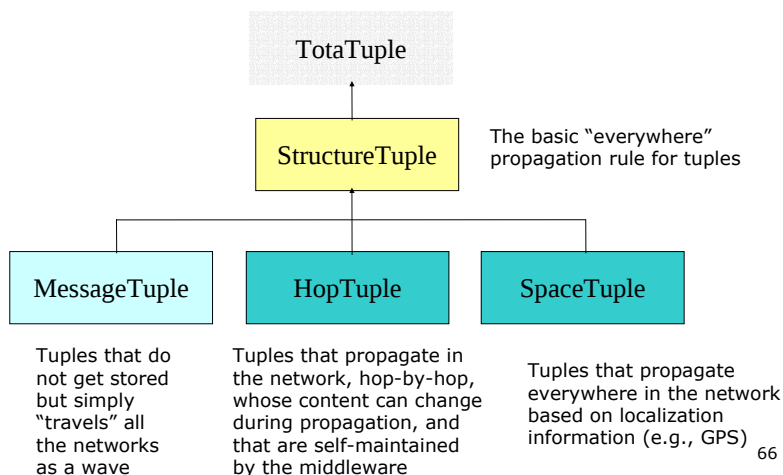
TOTA Tuples

```
abstract class TotaTuple {  
    protected TotaInterface tota;  
    /* the state is the tuple content */  
    public void init(TotaInterface tota) {  
        // the init method initialize the tuple  
        this.tota = tota;  
    }  
    public abstract void propagate();  
    // the propagate method is abstract  
    // has to be implemented by subclasses  
    // to specify how the tuple should propagate  
    public void react(String react, String event){  
        // this method specify how the tuple should  
        // react, e.g., maintain, in response to  
        // network dynamic  
    }  
}
```

65

The TOTA Tuples Library

Partial view of the TOTA tuple hierarchy/library



66

Tuple Programming

Other than defining its content, programming a tuple may imply programming its propagation rule, which we suggest structuring as below and implementing some of the sub-method specified below

```
public final void propagate() {
    if(decideEnter()) {
        boolean prop = decidePropagate();
        changeTupleContent();
        makeSubscriptions();
        tota.store(this);
        if(prop)
            propagate();
    }
}
```

67

The HotTuple Class

Tuple that propagates in the whole network and simply increment a **hop** value by one at each hop in the network

```
public class HopTuple extends StructureTuple {
    public int hop = 0;
    public boolean decideEnter() {
        super.decideEnter();
        NMGradient prev =(NMGradient)tota.keyrd(this);
        return (prev == null || prev.hop > (this.hop + 1));
    }
    protected void changeTupleContent() {
        super.changeTupleContent();
        hop++;}
}
```

68

The DownhillTuple Class

A tuple that propagates by following downhill the gradient of an already stored *HopTuple* (it propagates in a single path across the descent of an *HopTuple*)

```
public class DownhillTuple extends StructureTuple {
    public int oldVal = 9999; HopTuple trail;
    public boolean decideEnter() {
        super.decideEnter();
        int val = getGradientValue();
        if(val < oldVal) {
            oldVal = val; return true;}
        else return false; }
    private int getGradientValue() {
        Vector v = tota.read(trail);
        int min = 9999;
        for(int i=0; i<v.size(); i++) {
            HopTuple gt =(HopTuple)v.get(i);
            if(min > gt.hop) min = gt.hop;    }
        return min;}}
```

Programming TOTA Agents

- Let's refer to the example of a tourist in a museum that wants to know where "Mona Lisa" is, and wants to get information about
- The agent in the user PDA
 - inject a Query field in the form of a *HopTuple*, which propagates in the network, and then wait for the income of some Answer tuple
- The agent of the Mona Lisa picture
 - Can be programmed to react to incoming Query tuples
 - By injecting an Answer tuple in the form of a *DownhillTuple* that follows the path of the Query tuple to reach back the user

The UserAgent

```
public class UserAgent implements AgentInterface
{
    private TotaMiddleware tota;

    public void start() {
        NMGradient query = new NMGradient();
        query.setContent("Monna Lisa");
        tota.inject(query);
        DownhillTuple answer = new DownhillTuple();
        answer.setContent("Monna Lisa *");
        tota.subscribe(answer, this, "display");
    }

    public void react(String reaction, String event) {
        if(reaction.equalsIgnoreCase("display ")) {
            System.out.println("Monna Lisa:" + event);
        }
    }
}
```

The MonaLisaAgent

```
public class MonaLisaAgent implements AgentInterface {
    private TotaMiddleware tota;
    private String description, location;

    public void start() {
        // subscribe to query. Queries will be conveyed in HopTuple
        HopTuple query = new HopTuple();
        query.setContent(description);
        tota.subscribe(query, this, "answerQuery");
    }

    /*code of the reaction, here it injects the answer tuple. The
    answer will be conveyed in a DownhillTuple following the
    query. The query is here referenced as HopTuple event */
    public void answerQuery (OneHopIncTuple event) {
        DownhillTuple answer = new DownhillTuple(event);
        answer.setContent(description+" "+location);
        tota.inject(answer);
    }
}
```

A Snapshot of TOTA Implementations

Simulator

The screenshot displays the TOTA Simulator interface. On the left, a network graph is shown with numerous green nodes connected by red lines, representing a complex network topology. On the right, a command window titled 'Tota P39' is visible, showing a list of commands and their execution status. The commands include 'insert', 'remove', and 'query' operations on the network. The interface also includes a menu bar with options like 'File', 'Edit', 'View', and 'Help'.

PDA

A photograph of a PDA (Personal Digital Assistant) device. The screen displays the TOTA interface, showing a network graph and a command window. The device has a small screen and a keypad.

Lego Robot

A photograph of a Lego robot, specifically a Mindstorms NXT, equipped with a PDA and network components. The robot is shown from a side view, highlighting the PDA mounted on top and the various sensors and actuators connected to it. The robot is positioned on a light-colored tiled floor.



Tuple Maintenance

- Tuples must be **propagated** and **maintained**.
- Propagation is based on a simple hop-by-hop epidemic mechanism.
- Maintenance is based on local perception of network events and re-shaping of the propagated structure accordingly

The diagram illustrates tuple maintenance in a network. It shows two network topologies. The top topology is a linear sequence of nodes A, B, C, D, E, F, and G. Above each node is a red 'X' mark, indicating that the tuple is no longer maintained at that node. A large red 'X' is placed over the link between nodes A and B. The bottom topology shows a network where node C is connected to B and D, and D is connected to E. Node B is crossed out with a red 'X'. Node D is also crossed out with a red 'X' and has a '4' next to it, possibly indicating a hop count or a specific maintenance action.

-



Conclusions and Open Issues

- Field-based coordination as a peculiar form of stigmergy
 - Promoting context-awareness
 - Promoting self-organization
- That finds several useful applications
 - Supporting motion coordination
 - As well as various applications in pervasive computing
- There are several issues that would be worth exploring
 - Exploiting fields for P2P computing
 - Exploiting fields for multiagent coalitions and for computational economies
 - Using computational fields to understand the mechanisms of morphogenesis