

# Hovering Information - Self-Organising Information that Finds its Own Storage

Alfredo A. Villalba Castro  
Centre Universitaire d'Informatique  
University of Geneva, Switzerland  
alfredo.villalba@cui.unige.ch

Giovanna Di Marzo Serugendo  
School of Computer Science  
and Information Systems  
Birkbeck, University of London, UK  
dimarzo@dcs.bbk.ac.uk

Dimitri Konstantas  
Centre Universitaire d'Informatique  
University of Geneva, Switzerland  
dimitri.konstantas@cui.unige.ch

## Abstract

*A piece of Hovering Information is a geo-localized information residing in a highly dynamic environment such as a mobile ad hoc network. This information is attached to a geographical point, called the anchor location, and to its vicinity area, called anchor area. A piece of hovering information is responsible for keeping itself alive, available and accessible to other devices within its anchor area. Hovering information uses mechanisms such as active hopping, replication and dissemination among mobile nodes to satisfy the above requirements. It does not rely on any central server. This paper defines the hovering information concept in a formal way and presents the results of simulations performed for two algorithms aiming to ensure the availability of a piece of hovering information at its anchor area. The first algorithm is called Attractor Point: the anchor location acts as an attractor for the pieces of hovering information attached to it. The second algorithm is a Broadcast algorithm and is used as a benchmark for comparing the performances of the Attractor Point. The results of the simulations demonstrate that the Broadcast algorithm performs slightly better than the Attractor Point in terms of availability, while the Attractor Point algorithm outperforms the Broadcast algorithm in bandwidth and memory consumption. It also allows a better concentration of hovering information around the anchor location. Current assumptions, such as unlimited memory of mobile devices, or random way point mobility model need to be dropped in future simulations in order to obtain results closer to real-life scenarios.*

## 1 Introduction

Hovering information [14] is a concept characterising self-organising information responsible to find its own storage on top of a highly dynamic set of mobile devices. The main requirement of a single piece of hovering information is to keep itself stored at some specified location, which we call the anchor location, despite the unreliability of the device on which it is stored. Whenever the mobile device, on which the hovering information is currently stored, leaves the area around the specified storage location, the information has to hop - "hover" - to another device.

Current approaches in this area (cf. Section 6) try to either define a virtual structured overlay network on top of this environment offering a stable virtual infrastructure, or propose a system-based approach offering services such as information dissemination and storage. In these approaches, the mobile nodes decide when and to whom the information is to be sent. Here we take the opposite view; it is the information that decides upon its own storage and dissemination. This opens up other possibilities, not available for traditional MANET services, such as different pieces of hovering information all moving towards the same location and (re-)constructing there a coherent larger information for a user, e.g. TV or video streaming on mobile phones.

Hovering information is a *self-organised* user-defined information which do not need a central server to exist. Individual pieces of hovering information each use local information, such as direction, position, power and storage capabilities of nearby mobile devices, in order to select the next appropriate location. Hovering information benefits from the storage space and communication capacities of the underlying mobile devices. It is not residing in a centralized server, and is not bound to any mobile operator.

Main dependability requirements of hovering information are *survivability*, *availability* and *accessibility*. *Survivability* means that the information is alive somewhere in the environment (i.e. it is stored in some device) but not necessarily close to its anchor location. *Availability* means that the information has found storage in the vicinity of the anchor location. Finally, *accessibility* combines both availability and communication range of wireless mobile devices, and represents the possibility for a user located in the anchor location to access hovering information stored on nearby devices.

This paper presents the hovering information model as well as a preliminary algorithm allowing single pieces of hovering information to get attracted to their respective anchor locations.

Section 2 discusses potential applications of this concept. Section 3 presents the hovering information concept and model. Section 4 discusses the Attractor Point algorithm that we have designed where the information is "attracted" by the anchor location and keeps coming back to this location, and a general Broadcast algorithm we implemented in order to allow comparisons. Section 5 reports on simulation results related to availability and additional metrics such as number of messages exchanged or memory storage used. Finally Section 6 compares our approach to related works, and Section 7 discusses some future works.

## 2 Applications

This section highlights some future applications in very different areas that could all be developed from the concept of hovering information.

**Urban Security** The environment considered for this application is a dense urban area where each person carries a GPS enabled device. A hovering information service is available on the device, which allows users to enter comments or warnings related to dangers in the urban environment. Different types of information can be disseminated by the user: warnings about holes in the road or about the existence of thieves (pick-pockets); or comments like "this corridor is dark and I feel a danger". Each person entering the area where the information is kept will potentially receive it. Each user has a "profile" and chooses what types of dangers are relevant to her. For example a blind person will be interested in holes on the road; a weight lifter is not really concerned to be attacked by a thief, while an elderly will find these two pieces of information particularly relevant for her. Similarly, policeman and security guards operating in the same local urban area with high-rate crime could exchange information to each other. Each user attracts different information depending on his profile as soon as it enters the region

where the information is located. For such an application the trust/security aspect becomes then crucial, since any GPS owner (including a criminal) may enter fraudulent information. Although we are also working on trust and security issues related to hovering information, this discussion is beyond the scope of this paper.

**Archaeological Sites** The environment in this application is a real archaeological site, most likely an outdoor site, rather large, where visitors just move freely inside it. Users are the visitors of the site. Fixed sensors placed at several locations on the archaeological site provide either current information on actual weather and temperature or historical information about the different locations in the archaeological site. Users visiting the archaeological site wear mobile devices carrying information specific to the user itself (e.g. adult, child, man, woman, teacher, etc) or more likely about the virtual character they want to learn more about. The user receives on his mobile device a virtual reconstructed version of the site as it would be on the day they are visiting: with the same weather conditions, targeted in content to the character they wanted to learn more about, and populated with the other characters that are currently visiting the same location. For instance, consider a group of 3 people (characters) visiting a house in an archaeological site: a cook, a child, and a house's owner. All of them would have a virtual view of how the house looked like at that time. If it is sunny then the view shows a sunny area, if it is cold it could show heating aspects. Each visitor would have a specific tailored explanation (cooking, playing, and owners information) and could visualize the avatars of the others on his mobile device while moving around. There are different types of information going around: visitor-dependent information, weather information, and archaeological/historical information of a specific location in the archaeological site. Personalised information is attracted by the corresponding user's device, aggregates there and shows some virtual view of the site (audio only or both audio/video).

**Self-Generative Art** Self-generative art [8] refers to art practice where inputs from the creator of the piece of art are assembled together according to some rules (algorithm), such that the resulting piece of art, generated by a computer, is a real-time unfolding work which may display randomness, evolutionary aspects, or self-organising (swarm) behaviour. The piece of art may be music, painting, 3D construction, writing, etc. In this case, the users/creators would then be the multiple visitors of a "learning art experience centre". A piece of art could be a large scale 3D virtual shape

produced by inputs provided by each visitor: location in the experience area, weight/height and behaviour (jumping/walking), preferred colour or shape. The virtual shape would have holes where people are currently placed, and bumps where they have left; or heart beating bumps if they are jumping. Rules for combining the different inputs could vary: assembling the virtual surface according to actual or relative distances in the real world, summing up the colours and weights according to different algorithms, keeping visitors input for a random amount of time after they have left the experience area, etc. Sensors are required to determine the weight/size of the person. This value will then have an impact on the final virtual surface. Heavier people or groups of people will provide heavier holes, etc. People moving across the surface create temporary paths across it (that would dissolve completely after a certain time). People carry mobile device through which they provide additional personalised information: preferred colour and shape. Each input provided by the different visitors is a piece of hovering information whose goal is to travel to the centre of the experience area and aggregate there with the others in order to provide a visual 3D shape.

**Intravehicular Networks** Virtual tags are inserted at specific locations on roads or motorways either by cars' drivers or traffic management staff. The purpose here is to provide information to cars' drivers about road conditions, accidents, etc. In such a scenario, using the notion of hovering information, such tags will not be stored on a specific server and made available to users when they reach the zone of interest of the information. Instead the tags are locally stored in the cars and made available through wireless channels to nearby cars. Since data have a meaning for the specific location they have been attributed, data will have to "change" car as soon as the car they are currently stored in leaves the area of the anchor location. The data will then hop from one car to the next one.

**Emergency Scenarios** In an emergency scenario, virtual data present before a disaster may want to "survive" by using emergency crew or survivors devices. This data can also present useful information for emergency services. Additionally, disaster's survivors may want to indicate their position by placing the appropriate hovering information attaching it to their own location. Emergency crew member can place hovering information to areas where survivors have been found or where there is a chance to find some survivors. In this case, the information will hop from one emergency/survivor device to another one.

### 3 Hovering Information Concept

This section formally defines the notion of hovering information system, as well as the three main dependability requirements: survivability, availability and accessibility.

#### 3.1 Coordinates, distances and areas

We denote by  $\mathbb{E}$  the set of all pairs of geographic coordinates,

$$\mathbb{E} = [-90, 90] \times [-180, 180].$$

A *geographic coordinate*  $a$  is a pair:

$$a = (lat, long), \text{ and } a \in \mathbb{E}.$$

North latitude and East longitude are positive coordinates, while South latitude and West longitudes are negative coordinates. We do not consider here depths and heights.

An *area*  $A(a, r)$  is defined as the disk whose centre is the geographic coordinate  $a$  and has a positive *radius*  $r \in \mathbb{R}^+$ :

$$A(a, r) = \{b \in \mathbb{E} \mid dist(a, b) < r\}.$$

We consider  $dist(a, b)$  to be the distance in meters between two locations on a sphere, provided by any reliable method. See for instance<sup>1</sup>.

#### 3.2 Mobile Nodes

Mobile nodes represent the storage and motion media exploited by pieces of hovering information. They are defined as follows. A *mobile node*  $n$  is a tuple:

$$n = (id, loc, speed, dir, r_{comm}),$$

where:

$id \in \mathcal{I}$  is a mobile node identifier,

$loc \in \mathbb{E}$  is a geographic coordinate location,

$speed \in \mathbb{R}^+$  is a speed in  $m/s$

$dir \in \mathbb{E}$  is a relative geographic coordinate location,

$r_{comm} \in \mathbb{R}^+$  is the communication radius in meters.

We denote by  $\mathcal{I}$  the set of all mobile nodes identifiers. When referring to the  $id, loc$  or other field of a mobile node  $n$ , we will use the following notation  $id(n), loc(n)$ , etc. Field  $loc(n)$  represents the current location of node  $n$ , while  $dir(n)$  is a vector representing the direction of its most recent movement. The range of communication  $r_{comm}$  is the maximum distance in meters within which the mobile node may communicate wirelessly with another mobile node.

<sup>1</sup><http://www.fcc.gov/mb/audio/bickel/distance.html>

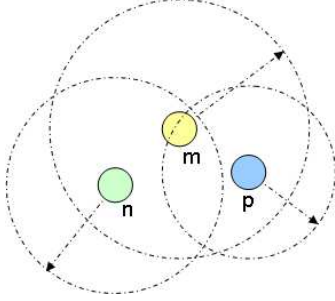
Let's consider  $\mathcal{N}$  a set of mobile nodes, we consider that identifiers of mobile nodes are unique, and we will say that  $\mathcal{N}$  is *well defined* if:

$$\forall n_1, n_2 \in \mathcal{N}, (id(n_1) = id(n_2)) \Rightarrow (n_1 = n_2).$$

Given a mobile node  $n$  with location  $loc(n)$  and communication radius  $r_{comm}(n)$ , the *communication area* of  $n$ ,  $A_C(n)$ , is the subset of  $\mathbb{E}$  given by:

$$A_C(n) = A(loc(n), r_{comm}(n)).$$

Figure 1 shows three mobile nodes,  $m$ ,  $n$ , and  $p$ . While the communication range of  $m$  is enough to let it be in range of both  $n$  and  $p$ , the communication range of  $n$  and  $p$  being much smaller prevents them to be directly in range.



**Figure 1. Mobile Nodes and Communication Range**

### 3.3 Hovering Information

Let  $\mathcal{N}$  be a well defined set of mobile nodes. A *piece of hovering information*  $h$  is a tuple:

$$h = (id, a, r, n, data, policies, size),$$

where:

$id \in \mathcal{J}$  is a hovering information identifier,

$a \in \mathbb{E}$  is the anchor location,

$r \in \mathbb{R}^+$  is the anchor radius,

$n \in \mathcal{N}$  is the mobile node where  $h$  is currently located,

$data$  is the data carried by  $h$ ,

$policies$  are the hovering policies of  $h$ ,

$size \in \mathbb{N}^+$  is the size of  $h$  in bytes.

We denote by  $\mathcal{J}$  the set of all hovering information identifiers. When referring to the  $id$ ,  $a$  or other field of a hovering information  $h$ , we will use the following notation  $id(h)$ ,

$a(h)$ , etc. Policies stand for hovering policies stating how and when a piece of hovering information has to hover. The size is an important element of a single piece of hovering information; however the simulation algorithms presented in this paper are not yet using this notion.

A piece of hovering information  $h$  is a piece of data whose main goal is to remain stored in an area centred at a specific location called the *anchor location*  $a(h)$ , and having a radius  $a(r)$ , called the *anchor radius*.

The *anchor area* of  $h$ ,  $A_H(h)$ , is the disk whose centre is the anchor location  $a(h)$  and whose radius is  $r(h)$ :

$$A_H(h) = A(a(h), r(h)).$$

Let  $\mathcal{H}$  be a set of pieces of hovering information. We consider that identifiers of pieces of hovering information are unique, but replicas (carrying same data and anchor information) are allowed on different mobile nodes, and we will say that  $\mathcal{H}$  is *well defined* iff:

$$\begin{aligned} \forall h_1, h_2 \in \mathcal{H}, (h_1 \neq h_2) \Rightarrow \\ (id(h_1) \neq id(h_2)) \vee \\ ((id(h_1) = id(h_2)) \wedge (a(h_1) = a(h_2)) \wedge \\ (r(h_1) = r(h_2)) \wedge (data(h_1) = data(h_2)) \wedge \\ (n(h_1) \neq n(h_2))). \end{aligned}$$

Let  $\mathcal{H}$  be a well defined set of pieces of hovering information. Let  $h \in \mathcal{H}$  be a hovering information, a *replica*  $h_r$  of  $h$  is a piece of hovering information  $h_r \in \mathcal{H}$  such that:

$$id(h) = id(h_r) \wedge n(h) \neq n(h_r).$$

From now on, we will consider only well defined sets  $\mathcal{H}$  of pieces of hovering information, where pieces of hovering information with the same  $id$  are either the same or a replica of each other. We also consider that there is only one instance of a hovering information in a given node  $n$ , any other replica resides in another node.

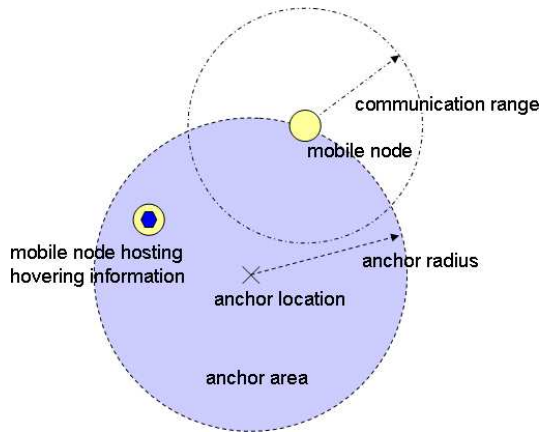
Figure 2 shows a piece of hovering information (blue hexagon) and two mobile nodes (yellow circles). One of them hosts the hovering information whose anchor location, radius and area are also represented (blue circle). The communication range of the second mobile node is also showed.

**Definition 3.1** (Hovering Information System at time  $t$ ). A *hovering information system at time  $t$* ,  $HoverInfo_t$ , is a tuple:

$$HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t),$$

where  $\mathcal{N}_t$  is a well defined set of mobile nodes,  $\mathcal{H}_t$  is a well defined set of hovering information over  $\mathcal{N}_t$ :

$$\forall h \in \mathcal{H}_t \Rightarrow n(h) \in \mathcal{N}_t.$$



**Figure 2. Mobile Nodes and Hovering Information**

A hovering information system at time  $t$  is a snapshot (at time  $t$ ) of the status of the system, the system then evolves at each tick  $t, t + 1$ , etc. Mobile nodes can change location, new mobile nodes can join the system, others can leave. New pieces of hovering information can appear (with new identifiers), replicas may appear or disappear (same identifiers but located on other nodes), hovering information may disappear or change node.

Figure 3 shows two different pieces of hovering information  $h_1$  (blue) and  $h_2$  (green), having each a different anchor location and area. Two replicas of  $h_1$  are currently located in the anchor area (in two different mobile nodes  $n_2$  and  $n_4$ ), while three replicas of  $h_2$  are present in the anchor area of  $h_2$  (in nodes  $n_2, n_3$  and  $n_5$ ). It may happen that a mobile device hosts replicas of different pieces of hovering information, as it is the case in the figure for the mobile node  $n_2$  that is at the intersection of the two anchor areas. The arrows here also represent the communication range possibilities among the nodes.

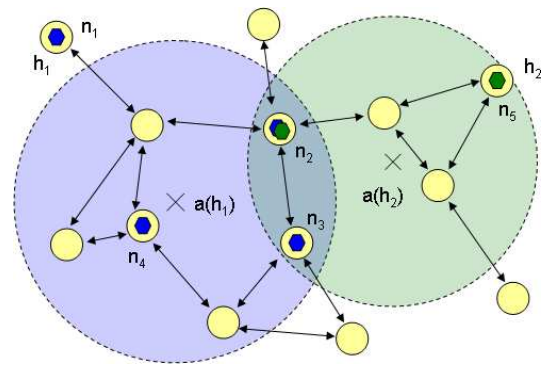
### 3.4 Notations

Before defining the notions of survivability, availability and accessibility, we will define the following additional notations.

Let's consider  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$ , a Hovering Information System at time  $t$ , let  $h$  be a piece of hovering information, and  $n \in \mathcal{N}_t$  be a mobile node at time  $t$ , we denote:

$$R_H(h, t) = \{k \in \mathcal{H}_t \mid id(h) = id(k)\}$$

as the set of replicas of a piece of hovering information  $h$  at time  $t$ ,



**Figure 3. Hovering Information System at time  $t$**

$$R_N(n, t) = \{h \in \mathcal{H}_t \mid n(h) = n\}$$

as the set of pieces of hovering information in node  $n$  at time  $t$ ,

$$P_N(n, t) = loc(n)$$

as the position of node  $n$  at time  $t$ ,

$$N_N(n, t) = \{m \in \mathcal{N}_t \mid (dist(loc(m), loc(n)) < r_{comm}(n)) \vee (dist(loc(m), loc(n)) < r_{comm}(m)), \}$$

as the set of neighbouring nodes of node  $n$  at time  $t$ , and  $S(X)$  as the surface area of a surface  $X$ .

Since  $\mathcal{H}_t$  is well defined, there are no two different pieces of hovering information (with different data) referred by the same identifier. It is also important to notice that  $h$  does not necessarily belong to  $\mathcal{H}_t$ , it may have disappeared from the system, but some of its replicas are still located in some mobile nodes. We consider that  $R_N(n, t)$  is actually a set (not a multi-set), i.e. there are no copies of the same hovering information stored at the same location. The neighbouring nodes in  $N_N(n, t)$  are those in range of communication to  $n$ .

### 3.5 Properties - Requirements

#### 3.5.1 Survivability

A hovering information is alive at some time  $t$  if there is at least one node hosting a replica of this information.

**Definition 3.2** (Survivability of Hovering Information  $h$  at time  $t$ ). *Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ . Let  $h$  be a piece of hovering*

information, the survivability of  $h$  at time  $t$  is given by the boolean value:

$$sv_H(h, t) = \begin{cases} 1 & \text{if } \exists n \in \mathcal{N}_t, R_H(h, t) \cap R_N(n, t) \neq \emptyset \\ 0 & \text{otherwise.} \end{cases}$$

The survivability along a period of time is defined as the ratio between the amount of time during which the hovering information has been alive and the overall duration of the observation.

**Definition 3.3** (Rate of Survivability of Hovering Information  $h$  at time  $t$ ). *Let  $h$  be a piece of hovering information, the survivability of  $h$  between time 0 and time  $t$  is given by:*

$$SV_H(h, t) = \frac{\sum_{\tau=0}^t sv_H(h, \tau)}{t}.$$

### 3.5.2 Availability

A hovering information is available at some time  $t$  if there is at least a node in its anchor area hosting a replica of this information.

**Definition 3.4** (Availability of Hovering Information  $h$  at time  $t$ ). *Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ . Let  $h$  be a piece of hovering information, the availability of  $h$  at time  $t$  is given by:*

$$av_H(h, t) = \begin{cases} 1 & \text{if } \exists n \in \mathcal{N}_t, (P_N(n, t) \in A_H(h)) \wedge \\ & (R_H(h, t) \cap R_N(n, t) \neq \emptyset) \\ 0 & \text{otherwise.} \end{cases}$$

The availability of a piece of hovering information along a period of time is defined as the rate between the amount of time along which this information has been available during this period and the overall time.

**Definition 3.5** (Rate of Availability of Hovering Information  $h$  at time  $t$ ). *Let  $h$  be a piece of hovering information, the availability of  $h$  between time 0 and time  $t$  is given by:*

$$AV_H(h, t) = \frac{\sum_{\tau=0}^t av_H(h, \tau)}{t}.$$

### 3.5.3 Accessibility

We distinguish availability from accessibility in the following way: a piece of hovering information (or one of its replica) present on some node located in the anchor area is said to be available. However, such a piece of hovering information may not be accessible to a mobile which is far apart from the mobile node where the hovering information (or its replica) is actually stored.

A hovering information is accessible by a node  $n$  at some time  $t$  if the node is able to get this information. In other

words, if it exists a node  $m$  being in the communication range of the interested node  $n$  and which contains a replica of the piece of hovering information.

**Definition 3.6** (Accessibility of Hovering Information  $h$  for node  $n$  at time  $t$ ). *Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ . Let  $h$  be a piece of hovering information, let  $n \in \mathcal{N}_t$  be a mobile node, the accessibility of  $h$  for  $n$  at time  $t$  is given by:*

$$ac_H(h, n, t) = \begin{cases} 1 & \text{if } \exists m \in \mathcal{N}_t, (m \in N_N(n, t)) \wedge \\ & (R_H(h, t) \cap R_N(m, t) \neq \emptyset) \\ 0 & \text{otherwise.} \end{cases}$$

We also define the accessibility of a piece of hovering information as the rate between the covered area by the hovering information's replicas and its anchor area.

**Definition 3.7** (Accessibility of Hovering Information  $h$  at time  $t$ ). *Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ . Let  $h$  be a piece of hovering information, the accessibility of  $h$  at time  $t$  is given by:*

$$ac_H(h, t) = \frac{S(\bigcup_{r \in R_H(h, t)} A_C(n(r)) \cap A_H(h))}{S(A_H(h))},$$

where  $S(X)$  denotes the surface of  $X$ .

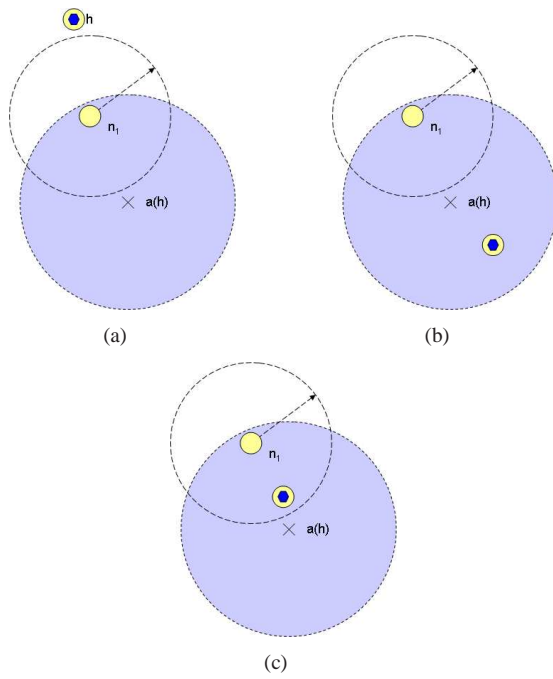
The accessibility along a period of time is defined as the average of the accessibility through that period of time.

**Definition 3.8** (Rate of Accessibility of Hovering Information  $h$  at time  $t$ ). *Let  $h$  be a piece of hovering information, the accessibility of  $h$  between time 0 and time  $t$  is given by:*

$$AC_H(h, t) = \frac{\sum_{\tau=0}^t ac_H(h, \tau)}{t}.$$

Let us notice that an available piece of hovering information is not necessarily accessible and vice-versa, an accessible piece of hovering information is not necessarily available. Figure 4 shows different cases of survivability, availability and accessibility. In Figure 4(a), hovering information  $h$  (blue) is not available, since it is not physically present in the anchor area, however it is survival as there is a node hosting it. In Figure 4(b), hovering information  $h$  is now available as it is within its anchor area, however it is not accessible from node  $n_1$  because of the scope of the communication range. Finally, in Figure 4(c), hovering information  $h$  is survival, available and accessible from node  $n_1$ .

It is thus important to distinguish availability from accessibility: a piece of hovering information may be available (i.e. present) in the anchor area, but due to actual communication ranges among the nodes, it is not necessarily accessible for all nodes into the anchor area.



**Figure 4. (a) Hoverinfo  $h$  is survival but neither available at its anchor area  $a(h)$  nor accessible. (b) Hoverinfo  $h$  is survival and available at its anchor area  $a(h)$  but not accessible from node  $n_1$ . (c) Hoverinfo  $h$  is survival, available at its anchor area  $a(h)$  and accessible from node  $n_1$**

Similarly, it is interesting to note that in some situations a piece of hovering information even though not available at its anchor location could be accessible for some nodes provided there is a node in communications range hosting  $h$ .

## 4 Algorithms for Hovering Information

Survivability, availability, and accessibility are among the most fundamental issues of hovering information as we discussed in [14] and [3]. Security and trust issues are important issues when considering hovering information, however they go beyond the scope of this paper, and will not be discussed here. Survivability addresses the problem of keeping a piece of hovering information alive as long as defined by the information itself. Availability deals with the problem of keeping the information present in its anchor area while accessibility relates to the possibility for a user to access a piece of hovering information stored on a device which is in communication range.

As mentioned in the previous section, these notions are closely related to each other, but none of them necessarily

implies the others.

This paper focuses on the study of the survivability and availability of a piece of hovering information. We propose an *Attractor Point* algorithm, whose aim is to keep the hovering information alive and available in its anchor area as long as possible. An anchor location  $a$  acts as an attractor point: all pieces of hovering information that have  $a$  as anchor location tend to converge towards  $a$ .

Besides the Attractor Point algorithm, we describe a *Broadcast* algorithm which is expected to have better survivability and availability performances than the Attractor Point, but at the cost of being more memory and network greedy. We use the Broadcast algorithm as a comparison threshold. Pieces of hovering information periodically broadcast (replicate) themselves to all the nodes in the communication range.

### 4.1 Assumptions

We make the following assumptions in order to keep the problem simple while focusing on measuring availability and resource consumption.

**Unlimited memory** All mobile nodes have an unlimited amount of memory able to store any number of hovering information replicas. The proposed algorithms do not take into account remaining memory space or the size of the hovering information.

$$Memory(node) = \infty, \forall node \in \mathcal{N}_t.$$

**Unlimited energy** All mobile nodes have an unlimited amount of energy. The proposed algorithms do not consider failure of nodes or impossibility of sending messages because of low level of energy.

$$Power(node) = \infty, \forall node \in \mathcal{N}_t.$$

**Instantaneous processing** Processing time of the algorithms in a mobile node is zero. We do not consider performance problems related to overloaded processors or execution time.

**In-built geo-localization service** Mobile nodes have an in-built geo-localization service such as GPS which provides the current position. We assume that this information is available to pieces of hovering information.

**Velocity vector service** Mobile nodes have an in-built velocity vector service providing the instantaneous speed and direction of the node. We assume that this information is available to pieces of hovering information.

**Neighbours discovering service** Mobile nodes are able to get a list of their current neighbouring nodes at any time. This list contains the position, speed, and direction of the nodes. As for the other two services, this information is available to pieces of hovering information.

## 4.2 Safe, Risk and Relevant Areas

Hovering policies are attached to pieces of hovering information. We consider here that all pieces of hovering information have the same hovering policies: active replication and hovering in order to stay in the anchor area (for availability and accessibility reasons), hovering and caching when too far from the anchor area (survivability), and cleaning when too far from the anchor area to be meaningful (i.e. disappearance). The decision on whether to replicate itself or to hover depends on the current position of the mobile device in which the hovering information is currently stored.

Given an anchor area  $A(a, r)$ , the *safe area*  $A(a, r_{safe})$  is the disk whose centre is the anchor location  $a$  and whose radius is the *safe radius*  $r_{safe}$ , a positive radius smaller than the anchor radius  $r$ , i.e.  $r_{safe} < r$  and  $r_{safe} \in \mathbb{R}^+$ :

$$A(a, r_{safe}) = \{b \in \mathbb{E} \mid dist(a, b) < r_{safe}\}.$$

A piece of hovering information located in the safe area can safely stay in the current mobile node, provided the conditions on the node permit this: power, memory, etc.

Given an anchor area  $A(a, r)$ , a *risk area* is a ring centred at the anchor location, which overlaps with the anchor area and is limited by the safe area.

The *risk area*  $R(a, r_{safe}, r_{risk})$  is the ring given by;

$$R(a, r_{safe}, r_{risk}) = A(a, r_{risk}) \setminus A(a, r_{safe}),$$

where  $r_{safe} < r < r_{risk}$  and  $r_{risk}, r_{safe} \in \mathbb{R}^+$ .

A piece of hovering information located in the risk area should actively seek a new location on a mobile node going into the direction of the safe area. It is in this area that the hovering information actively replicates itself in order to stay available and in the vicinity of the anchor location.

The *relevant area* limits the scope of survivability of a piece of hovering information. The relevant area  $A(a, r_{rel})$  is the disk whose centre is the anchor location  $a$  and whose radius is the *relevant radius*  $r_{rel}$  bigger than the risk radius:

$$A(a, r_{rel}) = \{b \in \mathbb{E} \mid dist(a, b) < r_{rel}\},$$

where  $r_{risk} < r_{rel}$ , and  $r_{rel} \in \mathbb{R}^+$ .

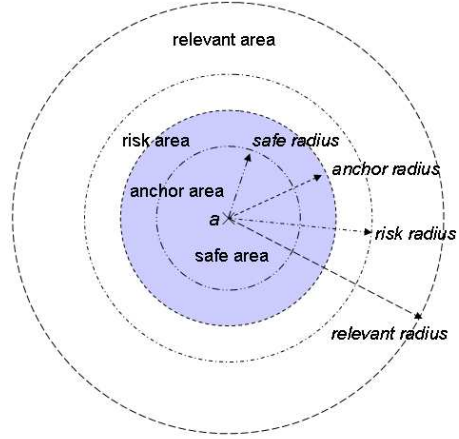
The ring area  $A(a, r_{rel}) \setminus A(a, r_{risk})$  represents the area where the hovering information seeks to survive but does not actively replicate itself (in order to avoid flooding). It may come back to the anchor area through mobile devices going in the direction of the anchor area.

The *irrelevant area* is all the area  $U(a, r)$ , outside the relevant area, it is given by:

$$U(a, r_{rel}) = \mathbb{E} \setminus A(a, r_{rel}).$$

A piece of hovering information located in the irrelevant area can disappear; it is relieved from survivability goals.

Figure 5 below depicts the different types of radiuses and areas discussed above centred at a specific anchor location  $a$ . The smallest disk represents the safe area, the blue area is the anchor area, the ring limited by the risk radius and the safe radius is the risk area, and finally the larger disk is the relevant area.



**Figure 5. Radiuses and areas**

The values of these different radiuses are different for each piece of hovering information and are typically stored in the Policies field of the hovering information. In the following algorithms we consider that all pieces of hovering information have the same relevant, risk and safe radius.

## 4.3 Replication

A piece of hovering information  $h$  has to replicate itself onto other nodes in order to stay alive, available and accessible. We describe two such replication algorithms for simulating two variants of these policies: the Attractor Point and Broadcast algorithms. Both algorithms are triggered periodically each  $T_R$  seconds and only replicas of  $h$  being in the risk area are replicated onto some neighbouring nodes (nodes in communication range) which are selected according to the replication algorithm.

### 4.3.1 Attractor Point Algorithm

The anchor location of a piece of hovering information acts constantly as an attractor point to that piece of hovering information and to all its replicas. Replicas tend to stay as

---

**Algorithm 1** Attractor Point Replication Algorithm

---

```
1: procedure REPLICATION
2:    $pos \leftarrow$  MY-POSITION
3:    $N \leftarrow$  MY-NEIGHBOURS
4:    $P \leftarrow$  POSITION( $N$ )
5:   for all  $replica \in REPLICAS$  do
6:      $anchor \leftarrow$  ANCHOR-LOCATION( $replica$ )
7:      $dist \leftarrow$  DISTANCE( $pos, anchor$ )
8:     if ( $dist \geq r_{safe}$ ) and ( $dist \leq r_{risk}$ ) then
9:        $D \leftarrow$  DISTANCE( $P, anchor$ )
10:       $D' \leftarrow$  SORT( $D$ )
11:       $M \leftarrow$  SELECT( $D', 1, k_R$ )
12:      MULTICAST(info,  $M$ )
13:     end if
14:   end for
15: end procedure
```

---

close as possible to their anchor area by jumping from one mobile node to the other.

Periodically and for each mobile node, the algorithm (see Algorithm 1) checks the position of the mobile node (line 2) as well as the list and position of all mobile nodes in communication range (lines 3 and 4). The algorithm then verifies whether there are some hovering information replicas being in the risk area that need to be replicated (line 8). The number of target nodes composing the multicast group is defined by the constant  $k_R$ . The distance between each mobile node in range and the anchor location is computed (line 9). The  $k_R$  mobile nodes with the shortest distance are chosen as the target nodes for the multicast (lines 10 and 11). The information part of the selected pieces of hovering information is then multicasted to the  $k_R$  mobile nodes, in communication range, closest to the anchor location (line 12).

Figure 6 illustrates the behaviour of the Attractor Point algorithm. Consider a piece of hovering information  $h$  in the risk area. It replicates itself onto the nodes in communication range that are the closest to its anchor location. For a replication factor  $k_R = 2$ , nodes  $n_2$  and  $n_3$  receive a replica, while all the other nodes in range do not receive any replica.

#### 4.3.2 Broadcast Algorithm

The Broadcast algorithm (see Algorithm 2) is triggered periodically (each  $T_R$ ) for each mobile node. After checking the position of the mobile node (line 2); pieces of hovering information located in the risk area (line 6) are replicated and broadcasted onto all the nodes in communication range (line 7). We expect this algorithm to have the best performance in terms of availability but the worst in terms of network and memory resource consumption.

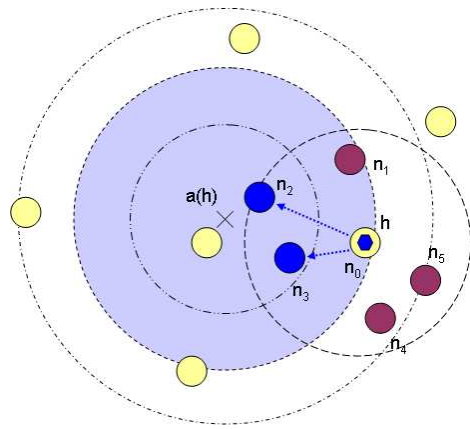


Figure 6. Attractor Point Algorithm

---

**Algorithm 2** Broadcast-based Replication Algorithm

---

```
1: procedure REPLICATION
2:    $pos \leftarrow$  MY-POSITION
3:   for all  $replica \in REPLICAS$  do
4:      $anchor \leftarrow$  ANCHOR-LOCATION( $replica$ )
5:      $dist \leftarrow$  DISTANCE( $pos, anchor$ )
6:     if ( $dist \geq r_{safe}$ ) and ( $dist \leq r_{risk}$ ) then
7:       BROADCAST( $replica$ )
8:     end if
9:   end for
10: end procedure
```

---

Figure 7 illustrates the behaviour of the Broadcast algorithm. Consider the piece of hovering information  $h$  in the risk area, it replicates itself onto all the nodes in communication range, nodes  $n_1$  to  $n_5$  (blue nodes).

#### 4.4 Caching and Cleaning Modules

Algorithm 3 shows the caching and storage algorithm. Each node is assumed to have an unlimited amount of memory. Therefore, when replicas are sent from one node to another, they are simply stored in the nodes memory. If a node receives two or more replicas of the same piece of hovering information  $h$ , the first replica to arrive is stored in the memory, and any subsequent one is ignored (lines 3 and 4). Therefore, at most one replica of each piece of hovering information is present in a given node  $n$ .

The cleaning algorithm (see Algorithm 4), periodically - each  $T_C$  seconds - and for each node, removes the replicas that are too far from their anchor location, i.e. those replicas that are in the irrelevant area (lines 6 and 7). Although the amount of memory is unlimited and replicas could stay forever in the nodes' memory, we remove the replicas that are too far away from their anchor location, this represents the cases where the replica considers itself too far from the an-

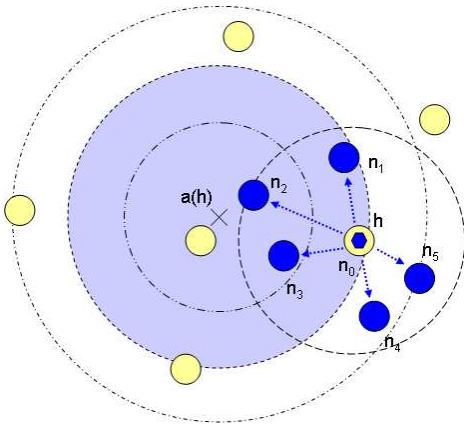


Figure 7. Broadcast Algorithm

---

**Algorithm 3** Caching Algorithm

---

```

1: procedure CACHING(message)
2:   replica ← GET-REPLICA(message)
3:   if CONTAINS(replica, REPLICAS) = false
4:     then
5:       INSERT(replica, REPLICAS)
6:     end if
7: end procedure

```

---

chor area and not able to come back anymore. This avoids as well the situation were all nodes have a replica.

---

**Algorithm 4** Cleaning Algorithm

---

```

1: procedure CLEANING
2:   pos ← MY-POSITION
3:   for all replica ∈ REPLICAS do
4:     anchor ← ANCHOR-LOCATION(replica)
5:     dist ← DISTANCE(pos, anchor)
6:     if (dist ≥ rrel) then
7:       REMOVE(replica, REPLICAS)
8:     end if
9:   end for
10: end procedure

```

---

## 4.5 Metrics

In order to evaluate and compare the above algorithms, the following values have been measured.

### 4.5.1 Messages complexity

We expect that the Broadcast algorithm exchanges globally more messages than the Attractor Point algorithm, i.e. that more replicas are broadcasted. We are also expecting that the Attractor Point algorithm reaches similar levels of

availability as the Broadcast algorithm while minimizing the number of exchanged messages (or multicasted replicas). This metric provides as well a feasibility criterion and will serve as a basis to extrapolate actual implementation results, such as latency and overhead of a real system.

**Definition 4.1** (Total Number of Messages sent at time  $t$ ). Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ , the number of messages sent at time  $t$  is given by:

$$msgs(t) = \sum_{n \in \mathcal{N}_t} msgs_n(t),$$

where  $msgs_n(t)$  represents the number of messages sent at time  $t$  by node  $n$ .

**Definition 4.2** (Accumulated Number of Messages sent by node  $n$ ). Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ , and  $n \in \mathcal{N}_t$  a mobile node, the accumulated number of messages sent by  $n$  between time 0 and time  $t$  is given by:

$$MSGs_n(t) = \sum_{\tau=0}^t msgs_n(\tau).$$

**Definition 4.3** (Total Number of Messages). Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ , the total number of messages sent by nodes in  $\mathcal{N}_t$  between time 0 and time  $t$  is given by:

$$MSGs(t) = \sum_{n \in \mathcal{N}_t} MSGs_n(t).$$

### 4.5.2 Space complexity

While the complexity of messages relates to the number of sent messages, the space complexity measures the amount of memory used by a device while hosting a piece of hovering information. Although, in this paper, we make the assumption that each mobile device has an unlimited amount of memory, we measure this in order to know how much memory is consumed by our algorithms. Again, in a real implementation this parameter will play an important role since we will prefer algorithms minimizing the memory utilization and maximizing the availability and survivability of hovering information.

**Definition 4.4** (Number of Pieces of Information in node  $n$  at time  $t$ ). Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ , and  $n \in \mathcal{N}_t$  a mobile node, the number of pieces of hovering information stored by node  $n$  at time  $t$  is given by:

$$mem_n(t) = |R_N(n, t)|$$

**Definition 4.5** (Total Number of Pieces of Information). Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ , the total number of pieces of hovering information stored by all nodes at time  $t$  is given by:

$$mem(t) = \sum_{n \in \mathcal{N}_t} mem_n(t).$$

**Definition 4.6** (Maximum Number of Pieces of Information stored by node  $n$ ). Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ , and  $n \in \mathcal{N}_t$  a mobile node, the maximum number of pieces of hovering information having been stored by a node  $n$  until  $t$  is given by:

$$MEM_n(t) = \max_{\tau=0}^t mem_n(\tau).$$

**Definition 4.7** (Maximum Number of Pieces of Information). Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ , the maximum number of pieces of hovering information having been stored between time 0 and time  $t$  is given by:

$$MEM(t) = \max_{n \in \mathcal{N}_t} MEM_n(t)$$

### 4.5.3 Concentration

The concentration of a given piece of hovering information  $h$  is defined as the rate between the number of replicas of  $h$  present in the anchor area and the total number of replicas of this hovering information in the whole environment.

**Definition 4.8** (Concentration of Hovering Information  $h$ ). Let  $HoverInfo_t = (\mathcal{N}_t, \mathcal{H}_t)$  be a Hovering Information System at time  $t$ , and  $h \in \mathcal{H}_t$  be a piece of hovering information, the concentration of  $h$  at time  $t$  is given by:

$$co(h, t) = \frac{|\{r \in R_H(h, t) | N_R(r, t) \in A_H(h)\}|}{|R_H(h, t)|}.$$

## 5 Evaluation

We evaluated the behaviour of the two above described algorithms under different scenarios by varying the number of nodes. For a given piece of hovering information  $h$ , we measured the availability of  $h$  (as given by Definition 3.5), the corresponding message complexity (Definition 4.3), the corresponding space complexity (based on Definition 4.7) and the concentration of  $h$  (Definition 4.8).

We performed simulations using the OMNet++ network simulator (distribution 3.3) and its Mobility Framework 2.0p2 (mobility module) to simulate nodes having WiFi-enabled communication interfaces.

## 5.1 Simulation Settings and Scenarios

The generic scenario consists of a surface of 500m x 500m with mobile nodes moving around following a Random Way Point mobility model with a speed varying from 1m/s to 10m/s without pause time. In this kind of mobility model, a node moves along a straight line with speed and direction changing randomly at some random time intervals. Before using this mobility model, the simulation has also been validated using two simple scenarios: in the first one nodes moved along a straight line at a constant speed following the same direction (one way road) and in the second one nodes moved along two opposite straight lines (double way road).

In the generic scenario, pieces of hovering information have an anchor radius ( $r_{anchor}$ ) of 50m, a safe radius ( $r_{safe}$ ) of 30m, a risk radius ( $r_{riks}$ ) of 70m, a relevance radius ( $r_{rel}$ ) of 200m, and a replication factor of 3 ( $k_R$ ).

Each node triggers the replication algorithm every 20 seconds ( $T_R$ ) and the cleaning algorithm every 60 seconds ( $T_C$ ). The caching algorithm is constantly listening for the arrival of new replicas. Table 1 summarises these values.

The nodes are equipped with 802.11b wireless interfaces. Based on this standard, we computed the communication range or maximal interference distance (MID) as 121m after the following formulae (extracted from the Mobility Framework implementation):

$$MID = \left( \frac{\lambda^2 \cdot pMax}{16 \cdot \pi^2 \cdot 10^{sat/10}} \right)^{1/\alpha}$$

where  $\lambda$  is the wave length,  $pMax$  is the maximal transmission power (mW),  $sat$  is the minimal signal attenuation threshold (dBm) and  $\alpha$  is the minimal path loss coefficient. We computed the  $MID$  for the following values: 2.4 GHz for carrier frequency (CF), 2 mW for the  $pMax$ , -110 dBm for the  $sat$  and 3.5 for the  $\alpha$  coefficient. The factor  $\lambda$  is computed as  $c/CF$  being  $c$  the speed of light. Broadcasts all occur at the same channel frequency.

Based on this generic scenario, we defined 10 specific scenarios with varying number of nodes: from 20 to 200 nodes, increasing the number of nodes by 20 each time. We have performed 20 runs for each scenario. One run lasts 3600 seconds of simulation time. All the results presented here are the average of the 20 runs for each scenario, and the errors bars represent a 95% confidence interval. All the simulations ran on a Pentium 1.7 GHz processor under Linux Mandriva OS.

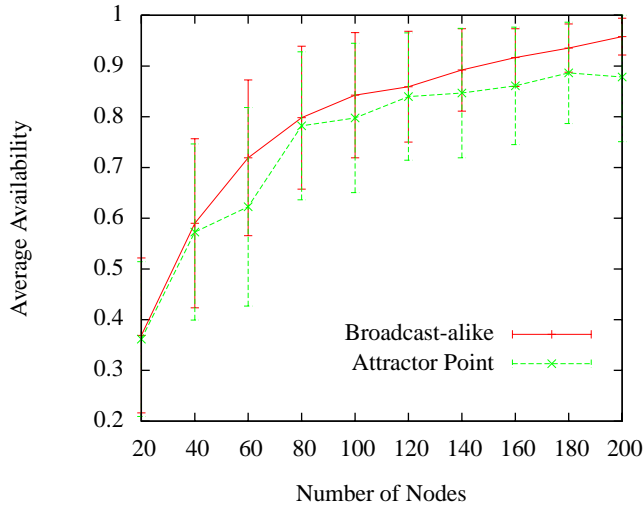
## 5.2 Results

Figure 8 shows, for each of the 10 scenarios, the average of the availability performance over the 20 runs (after one hour of simulation time). As expected, the Broadcast

Blackboard	500mx500m
Mobility Model	Random Way Point
Nodes speed	1m/s to 10 m/s
Communication range (MID)	121m
Replication time ( $T_R$ )	20s
Cleaning time ( $T_C$ )	60s
Anchor radius ( $r_{anchor}$ )	50m
Min risk radius ( $r_{safe}$ )	30m
Max risk radius ( $r_{risk}$ )	70m
Relevant radius ( $r_{rel}$ )	200m

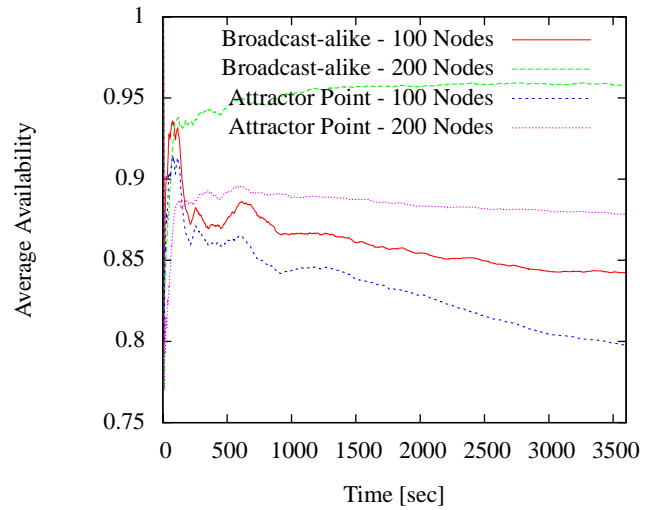
**Table 1. Simulation settings**

algorithm outperforms the Attractor Point algorithm. The results also indicate that the performances of the Attractor Point algorithm although lower are quite similar to those of the Broadcast algorithm. For both algorithms, we observe that an 80% of availability can be expected as soon as the number of mobile nodes in the environment reaches 120 nodes. This represents a density of 1.2 nodes per anchor area. The maximum availability value, nearly 95%, is reached by the Broadcast algorithm when the population of mobile nodes is 200, while the Attractor Point reaches 88% of availability for 180 nodes and above.



**Figure 8. Average availability after 1 hour of simulation**

Figure 9 depicts how the average availability of hovering information evolves during different periods of time. Two scenarios only have been chosen here: 100 and 200 nodes. In the case of the Broadcast algorithm and for the scenario with 200 nodes, we observe that the average availability reaches 95% and remains stable after nearly 1200 seconds of simulation time. However, for the same algorithm ap-



**Figure 9. Average availability through time**

plied to 100 nodes, and for the Attractor Point algorithm in both cases (100 and 200 nodes), the average availability first reaches a peak, but then keeps going down.

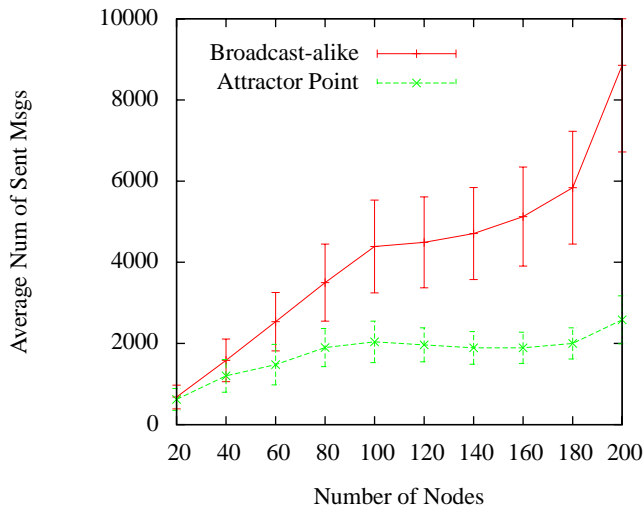
There are several explanations for this phenomenon. First, there is an initial moment of instability during the first 10min (600s) where the hovering information spreads itself around the anchor location. The average number of replicas then starts stabilising. For the specific case of the Attractor Point and 100 nodes, the average number of replicas still decreases after one hour. This may be due to the fact that replicas disappear (cleaning algorithm), thus the average number of replicas (from 0 to time  $t$ ) gradually continues going down.

For this particular case, we are planning to pursue further simulations, such as determining the instantaneous availability over time instead of the average.

Figure 10 shows the average number of messages sent for each of the 10 different scenarios. As expected, the Broadcast algorithm sends a higher number of messages when compared to the Attractor Point algorithm. This phenomenon is amplified when the number of nodes increases. In the worst case (200 nodes), the number of sent messages, in average, by the Broadcast algorithm is four times higher than the number of messages sent when the Attractor Point algorithm is used. In the other cases, it is 2.5 times higher (100 to 180 nodes)

Figure 11 shows in average the maximum number of replicas of a single piece of hovering information created during the simulations for each different scenario.

Again, we observe that the Broadcast algorithm creates more replicas than the Attractor Point. The curves for Space Complexity are very similar to those for Message Complexity (see Figure 10). This is explained as the number of sent



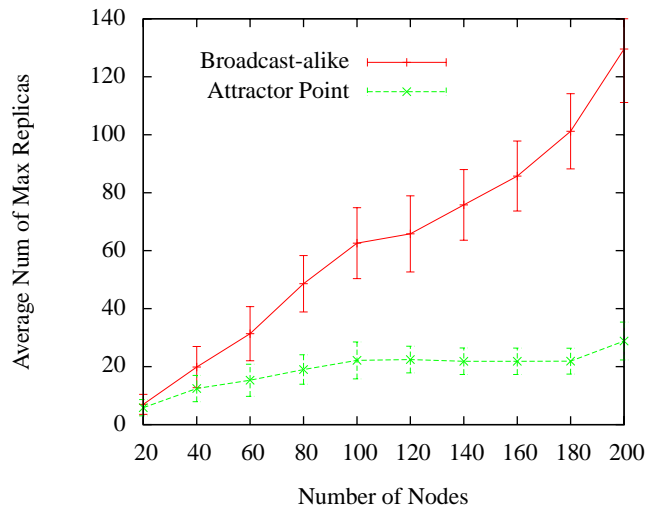
**Figure 10. Messages Complexity after 1 hour of simulation**

messages is directly proportional to the number of existing replicas; since each replica can potentially send messages (replicate itself again).

Figure 12 shows the concentration factor. We observe that the Attractor Point algorithm concentrates more replicas in the anchor area than the Broadcast algorithm. The concentration rate is above 14% for the Attractor Point algorithm when the number of mobile nodes is 80 or more. A maximal concentration rate of 8% is reached by the Broadcast algorithm. The Attractor Point concentrates 2 to 3 times more replicas than the Broadcast algorithm (depending on the number of nodes considered).

## 6 Related Works

The Virtual Infrastructure project [4, 5, 6, 7] defines virtual (fixed) nodes implemented on top of a MANET. This project proposes first the notion of an *atomic memory*, implemented on top of a MANET, using the notion of *quorums* or focal points where a reasonable amount of mobiles nodes intersect. Quorums work as atomic memory cells and ensure their persistency by replicating their state in neighbouring mobile devices. This notion has been extended to the idea of *virtual mobile nodes* which are state machines having a fixed location or a well-defined trajectory and whose content is also replicated among the nearby mobile devices. Finally, this project provided the notion of a *timed I/O automaton mobile node* where virtual mobile nodes access a clock in order to perform real-time operations. The motivation behind this project is the development of a virtual infrastructure on top of which it will be easier to define or

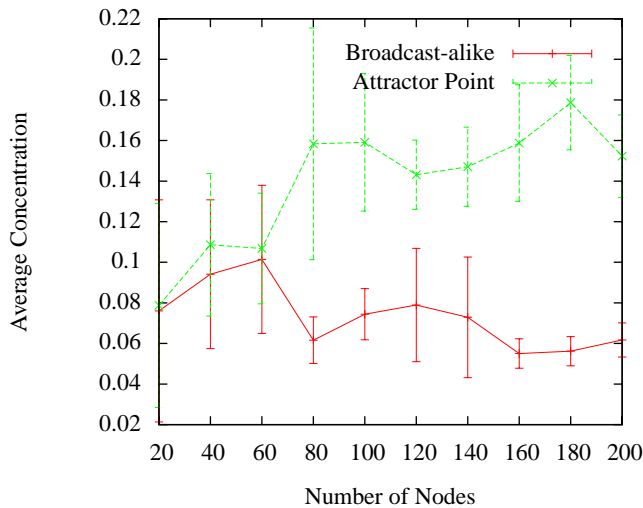


**Figure 11. Space Complexity**

adapt distributed algorithms such as routing, leader election, atomic memory, motion coordination, etc. Hovering information shares similar characteristics, it tries to benefit from the mobility of the underlying nodes, but the goal is different. We intend to provide a hovering information service on top of which applications using self-organising user-defined pieces of information can be built.

GeOpps [10] proposes a geographical opportunistic routing algorithm over VANETs (Vehicular Ad Hoc Networks). The algorithm selects appropriate cars for routing some information from a point A to a point B. The choice of the next hop (i.e. the next car) is based on the distance between that car's trajectory and the final destination of the information to route. The planned trajectory generated by a navigation system of the car is used when estimating the relevance of a car to route some information. This work focuses on routing information to some geographical location, it does not consider the issue of keeping this information alive at the destination, while this is the main characteristics of hovering information.

The work proposed by [11] aims to disseminate traffic information in a network composed by infostations and cars. The system follows the publish/subscribe paradigm. Once a publisher creates some information, a replica is created and propagated all around where the information is relevant. Cars having a replica periodically poll their neighbouring cars, using a broadcast message, to know whether they are interested or not in the replica's information. If some cars reply in an affirmative way the information is sent to them. Based on these periodic pollings, clusters are composed and replicas are removed or propagated to clusters where more subscribers and interested cars are situated. Replicas are also propagated to a randomly chosen car part of the clus-



**Figure 12. Concentration**

ter driving in the opposite direction to that of the current host in order to try to keep the information in its relevant area. Cars reply polling with their interests and also their direction. While the idea is quite similar to that of hovering information, keeping information alive in its relevant area, this study does not consider the problem of having a limited amount of memory to be shared by many pieces of information or the problem of fragmentation of information. It also takes the view of the cars as the main active entities, and not the opposite view, where it is the information that decides where to go.

The Ad-Loc project [1] proposes an annotation location-aware infrastructure-free system. Notes stick to an area of relevance which can grow depending on the location of interested nodes. Notes are kept in their relevance area by periodically broadcasting location-aware information to neighbouring nodes. This work also proposes to use this annotation system as a cache for Internet files in order to spare bandwidth. In this case, URLs are used as note identifiers. Similarly to the previous work, nodes are the active entities. In addition, in this case the size of the area of relevance grows as necessary in order to accommodate the needs of users potentially far from the central location. The information then becomes eventually available everywhere.

The ColBack system [9, 2] is part of the MoSAIC project and intends to set up a collaborative backup system for mobile devices. The two main issues the authors investigate are: fault- and intrusion-tolerant collaborative backup; and the self-carried reputation and rewards for collaboration. The environment consists of sporadically interconnected and mutually suspicious peer devices having no fixed infrastructure and access to trusted third parties. This system does not focus on geo-localized information but repli-

cation strategies and replica scheduling and dissemination techniques could be used as inspiration for hovering information replication algorithms.

PeopleNet [13] describes a mobile wireless virtual social network which mimics the way people seek information via social networking. It uses the infrastructure to propagate queries of a given type to users in specific geographical locations called bazaars. Within each bazaar the query is further propagated between neighbouring nodes via peer-to-peer connectivity until it finds a matching query. The proposed queries propagation inside bazaar techniques could be a source of inspiration when we will develop query to retrieve specific hovering information.

## 7 Conclusion

In this paper we have defined the notion of hovering information in a formal way and we have defined and simulated the Attractor Point algorithm which intends to keep the information alive and available in its anchor area. This algorithm multicasts hovering information replicas to the nodes that are closer to the anchor location of the information. The performances of this algorithm have been compared to those of a broadcast version which broadcast replicas regardless of the proximity or not to the anchor location.

The results show that the Broadcast algorithm outperforms the Attractor Point algorithm in terms of availability but only from a very small factor. The proposed Attractor Point algorithm is much less bandwidth and memory greedy than the Broadcast algorithm and achieves higher levels of concentration of data in the anchor area.

Our algorithms currently do not use information such as the size of individual pieces of hovering information, speed, direction and memory availability of mobile nodes. The preliminary results discussed here show that the concept is viable. More simulations, experiments and measures are necessary in order to derive more sophisticated algorithms and further implement them into real devices.

### 7.1 Discussion

Hovering information is self-organising user-defined information having its own autonomous behaviour once created. The information is responsible for replicating itself or for hovering from one to another in order to stay alive, available and accessible. A physical implementation of this mechanism would require a chip-like structure embedded into each node and containing a piece of software managing and supporting hovering information.

We have identified related domains such as routing over mobile ad hoc networks, mobile communications, annotation systems, information dissemination, distributed algo-

rithms, queries propagation, etc. as fields to be studied in connection with the concept of hovering information.

## 7.2 Future works

**Limited memory** The algorithms will be adapted to an environment where nodes have limited memory. Different pieces of hovering information will then compete for getting a place in nodes' memory and eventually some hovering information will disappear because of lack of space. The Attractor Point algorithm will be modified in order to cope with this problem and new caching and cleaning policies will be defined. These policies could be based on the priority of the information, its age, relevance to the current area, its history, the kind of node, etc.

**Real mobility patterns** We have tested the Attractor Point algorithm under a Random Way Point mobility model but in real world people do not behave in this way. People have routines and tend to follow the same trajectories daily while introducing some uncertainty as well. Therefore, the Attractor Point algorithm has to be applied to scenarios following real mobility patterns, for instance crowd mobility patterns in a shopping mall or traffic mobility patterns in a city.

**Real wireless conditions** The simulations performed have been done in an environment where there were neither wireless channel interferences nor physical obstacles. In real world scenarios, these two factors are inherent to wireless communications. It is thus very important to apply the Attractor Point algorithm in a more realistic environment taking in consideration these factors in order to measure the negative drawbacks on the availability performances.

**Fragmentation and recombination (swarm)** In this paper we have considered atomic information only, but depending on the size of the hovering information (e.g. an image or even a video) it could be fragmented into smaller pieces to fit in multiple nodes' memory. A query of this information will require a recombination mechanism that recovers the different pieces and gets them reassembled to form the original information. We are currently considering this recombination process as a swarm of self-assembling information particles.

**Movement speed and direction** The current attractor point algorithm takes in consideration the position of the neighbouring nodes only. A significant improvement will be achieved by taking into consideration the speed and direction of the nodes when choosing the nodes that will host replicas.

**Coordinates precision** The Attractor Point algorithm and the simulations performed do not consider issues related to the precision of geo-localisation service. Precision is an important factor in real world scenario, since it deeply affects the ideal communication range. In order to cope with reality, it is thus important to evaluate and adapt the algorithm in order to take into account problems related to precision

**Other simulation environments** Results vary when simulation environments change. In order to further validate and compare the results obtained using OM-Net++, we will use other simulation environments such as Swarm<sup>2</sup> or attraction fields [12].

## References

- [1] D. J. Corbet and D. Cutting. Ad loc: Location-based infrastructure-free annotation. In *ICMU 2006*, London, England, Oct. 2006.
- [2] L. Courts, M.-O. Killijian, D. Powell, and M. Roy. Sauvegarde cooperative entre pairs pour dispositifs mobiles. In *UbiMob '05: Proceedings of the 2nd French-speaking conference on Mobility and ubiquity computing*, pages 97–104, New York, NY, USA, 2005. ACM Press.
- [3] G. Di Marzo Serugendo, A. Villalba, and D. Konstantas. Dependable requirements for hovering information. In *Supplemental Volume - The 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)*, pages 36–39, 2007.
- [4] S. Dolev, S. Gilbert, L. Lahiani, N. A. Lynch, and T. Nolte. Timed virtual stationary automata for mobile networks. In *OPODIS*, pages 130–145, 2005.
- [5] S. Dolev, S. Gilbert, N. A. Lynch, E. Schiller, A. A. Shvartsman, and J. L. Welch. Virtual mobile nodes for mobile ad hoc networks. In *DISC*, 2004.
- [6] S. Dolev, S. Gilbert, N. A. Lynch, A. A. Shvartsman, and J. Welch. Geoquorums: Implementing atomic memory in mobile ad hoc networks. In *DISC*, 2003.
- [7] S. Dolev, S. Gilbert, E. Schiller, A. A. Shvartsman, and J. Welch. Autonomous virtual mobile nodes. In *DIALM-POMC '05: Proceedings of the 2005 joint workshop on Foundations of mobile computing*, pages 62–69, New York, NY, USA, 2005. ACM Press.
- [8] P. Galanter. What is Generative Art? Complexity Theory as a Context for Art Theory. In *Generative Art Conference*, 2003.

---

<sup>2</sup><http://www.swarm.org>

- [9] M.-O. Killijian, D. Powell, M. Banâtre, P. Couderc, and Y. Roudier. Collaborative backup for dependable mobile applications. In *MPAC '04: Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing*, pages 146–149, New York, NY, USA, 2004. ACM Press.
- [10] I. Leontiadis and C. Mascolo. Geopps: Opportunistic geographical routing for vehicular networks. In *Proceedings of the IEEE Workshop on Autonomic and Opportunistic Communications. (Colocated with WOW-MOM07)*, Helsinki, Finland, June 2007. IEEE Press.
- [11] I. Leontiadis and C. Mascolo. Opportunistic spatio-temporal dissemination system for vehicular networks. In *MobiOpp '07: Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 39–46, New York, NY, USA, 2007. ACM Press.
- [12] M. Mamei and F. Zambonelli. *Field-Based Coordination for Pervasive Multiagent Systems*. Springer Series on Agent Technology. Springer Verlag, Berlin, 2005.
- [13] M. Motani, V. Srinivasan, and P. S. Nuggehalli. Peoplenet: engineering a wireless virtual social network. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 243–257, New York, NY, USA, 2005. ACM Press.
- [14] A. Villalba and D. Konstantas. Towards hovering information. In *Proceedings of the First European Conference on Smart Sensing and Context (EuroSSC 2006)*, pages 161–166, 2006.