A Search Method using Jumping Behavior

Paola Neri-Ortiz[1] and Rubén de la Mora-Basáñez[2]

[1]Soluciones Informáticas y Diseño SID7.0
Francisco del Paso y Troncoso 19 D -7, Col. Jardín Balbuena, 15900, D.F., Mexico
[2]Facultad de Física e Inteligencia Artificial,
Sebastián Camacho No.5 Col. Centro, 91000, Xalapa, Ver., Mexico
paolaneriortiz@yahoo.com, carlosvub@hotmail.com

Abstract. Jumping behavior consists of an interleaving sequence of conducts of living organisms, such as moving or sniffing. Jumping behavior is used in many activities vital to subsistence, such as: food foraging, search for prey, avoid predators, etc. There are several interpretations for this behavior from the ethological viewpoint. Nevertheless, there is no clear biological model that provides comprehension of this conduct. We have developed a kinetic model to simulate this behavior as one that maximizes the probability to find an object in an open infinite space. The advantage of this model lies in its ability to find a local max number of goals in shorter time. Jumping algorithm uses few resources, as compared to ant algorithm which needs several agents to finding goals. Using the kinetic model, this jumping behavior (proper of biological entities) will be used as a search method for patterns in a two dimensional space, carried out by synthetic agents. Kinetic factors to consider are: moving time, local searching time, size of step, and perception capabilities. The purpose of this work is to study the behavior of the model of the jumping conduct, considered as a search method of patterns in closed 2-D spaces, through an incremental methodology. For this purpose a simulator was designed, in it, an agent looks for 2-D patterns using the leaping kinetic model. Results obtained in this work can be used in Robotics to approximate the region that a group of robots can sweep; or as an exploration method of non-unstructured spaces, such as it occurs in internet search and images search.

Keywords: Jumping Behavior, Search Methods, 2-D Pattern.

1 Introduction

1.1 Jumping Behavior

The biological phenomenon of intermittent locomotion describes the manner in that animals move, during the execution of their daily chores, such as feeding, having intercourse, prey searching, etc. [1]. An organism that shows intermittent locomotion typically alternates movement with phases of rest. Such behavior is seen, for instance, in squirrels exploring a park, birds that glide looking for prey, dogs sniffing for buried bones, etc.

We also would like to introduce some terms:

Time test: Cycles number than an agent executes the search of goals in a selected space.

Advance: Space agent travels in the same direction, measured in pixels.

Sniffing: Space in which the agent performs a local search, measured in pixels.

Size of the pattern: Pattern width and height measured in pixels.

Density pattern: Pattern density measured in pixels.

Space dimension: Width and height of the space, measured in pixels.

Space density: Space density measured in pixels.

1.2 Kinetic Model

Although there is no clear biological model that explains such behavior, a model that describes it exists [2]. The kinetic model of jumping behavior consists in an alternating sequence of advance (move) behavior and sniffing (or search) behavior, which maximizes the probability of finding objects in an open infinite space.

In the advance (move) phase, the organism moves in some random direction during a random time with average of τ_{move}. In the sniffing phase (local search), the organism pauses its movement in order to perceive or smell for food or prey, or in order to explore the unknown space surrounding it. The time spent on this activity is a variable random with a mean value $\tau_{sniffing}$. Next we explain the process through an algorithm consisting of a search method for 2-D patterns.

1.3 Jumping Behavior as Search Method

We apply the jumping behavior metaphor to a search algorithm that an agent uses in order to find 2-D patterns. Each life cycle of the agent consists of moving periods alternating with searching or "sniffing" periods. The agent moves through the search space using random values that are generated, corresponding to moving times and searching times (local search or sniffing). In the advance stage, the agent moves in a given direction, randomly chosen, during a random time generated with a mean τ_1 and a standard deviation σ_1. In the sniffing stage, the agent uses a random variable with mean τ_2 and standard deviation σ_2, to delimit a window where the local search for the sought pattern is performed.

The agent uses moving times to explore the search space, and sniffing times to make a local search. According to the used metaphor, the algorithm looks for a relation between moving frequency and sniff frequency that maximizes the probability of finding goals, at the same time, uses the lowest amount of resources, where moving and sniffing are supposed as restrictions.

Another algorithm based on the animal behavior is the ant algorithm. The ant algorithm is a heuristic technique used to solve problems where the search space is not delimited and non-structured [5]. In nature the ants go randomly in searching of food and when returning to the colony with the reward they release pheromones to guide the others by a successful way. With time, some ways will be but clear than others. This type search requires that several ants return to the departure point before visualizing a route clearly. Therefore the time of route for each ant is double the time it spends in finding the objective. In the algorithm of jump described in this work, the agent gets the objective and it marks it of a different color, this avoids that it finds the same objective twice (and wastes time).

Below we explain the theory of agents used as computational model.

2 Search Algorithm with Jumping Behavior

2.1 Behavior of the Agent during the Search

The agent used in this work searches 2-D patterns using the algorithm that is explained below, in an isolated form during the test phase and in collaborative fashion during the second stage. This agent that searches goals exhibits a jumping behavior, and alternates two conducts; a moving or advance conduct, and a sniffing or local search conduct.

The algorithm is as follows:

1) The agent starts at an initial position.
2) It selects a direction α, a distance factor T_{move} and size factor T_{search}, all random (mean values in T_{move} and T_{search}, standard deviation at 20%). The mean values are selected by user.
3) It moves in the chosen direction with a step of length L_{move} (selected by user), with

   $x = T_{move} \cos \alpha$ y $y = T_{move} L_{move} \operatorname{sen} \alpha$

4) The agent uses T_{search} (size factor) and L_{search} (selected by user) to performs a local search in a window of size

   $x = y = \sqrt{T_{search} L_{search} \pi}$

   If the goal is found during this stage, a counter is incremented.
5) The iterations number is reduced, if this is larger than cero go to step 2. Otherwise, end the program.

Below some steps of the algorithm are explained in detail:

1) At the start, the agent is positioned in the upper left corner 0, 0.
2) Then, an angle α is selected at random, used to change the direction of advance of the agent. A random value T_{move} is generated (with mean in

T_{move} and standard deviation at 20%), the agent will use it like a distance factor.

3) During the advance behavior, the agent uses T_{move} and a length of the advance step L_{move} in order to move in the direction α, in this form:

$$x = T_{move} \cos \alpha \quad \text{y} \quad y = T_{move} L_{move} \operatorname{sen} \alpha$$

4) During the sniffing behavior, the agent uses a random value T_{search} with average T_{search} and standard deviation at 20%, and a step length L_{search} to search the goal in local form in a window of size.

$$x = y = \sqrt{T_{search} L_{search} \pi}$$

This relation was obtained by adjusting the local search area, as the area of a square that contains a cycle of radius L/2, as

$$Surface = L^2 = T_{search} L_{search} \pi$$

Where T_{search} is the sniffing (or local search) time, L_{search} is the step length and π is the constant relation that exists between the perimeter and the diameter of a circle.

2.2 Experimental Setting

To carry out the tests of the behavior of a search agent looking for 2-D patterns with the described behavior, an environment with variable density and size is required.

The environment was implemented in Java, as an interface that the search agent uses to show its behavior in graphical form. The interface has the ability to create a space and a 2-D pattern of variable size and density, and to insert that pattern in the space with another variable density. The system interprets the space and the pattern as a matrix of dots and stores it in a text file that can be reused in further executions. Fig. 1 shows the interface of the system.

The text boxes “Space Dimension and Space Density” modify the size in pixels and the space density. The values of pattern dimension and pattern density modify the size in pixels and the density of the pattern.

Pattern density value, establishes the probability with which the pattern will be inserted into the space.

The average values of moving and searching variables are used to generate random values of frequency with means in the selected points.

The number of iterations that the search agent will execute through the environment can be manipulated with the value “Cycles Number” once the test finishes, a report with the average of the moving and searching values, and the number of goals obtained, will be added to the file specified in “Save report as…”.

The “pause” button stops the execution of the search agent.

The “clear” button cleans the path of the agent, for easier identification of the patterns already found.

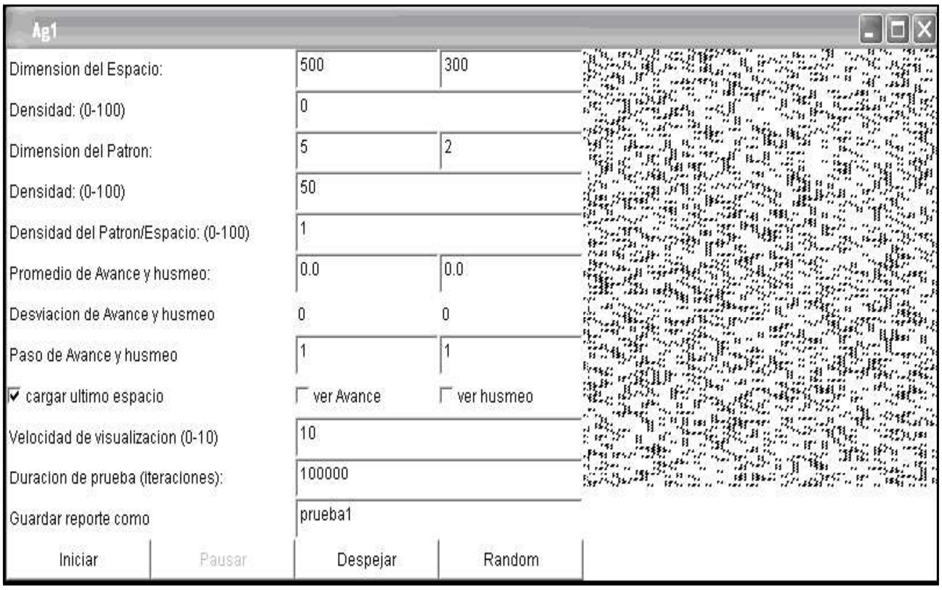

Fig. 1. Experimental environment.

The "random" button generates a pair of random numbers between 0 and 100, used as the average moving frequency and searching frequency. The standard deviation for both frequencies has been adjusted at 20%.

The "Init" button generates the space according to the established values, and draws it. If the verification box labeled as "load last space" is selected, the system will load the text files that contain the last space and pattern used previously, instead of generating a new one. Once the space has been generated or loaded, a sequence of 100 random numbers is created, having as mean value the values of moving and searching and standard deviation of 20%. These values will be used to simulate the behaviors of moving and sniffing as explained now.

Fig. 2 is an example of the agent's behavior during the execution of the program. The space with inserted patterns is shown as dots in the background. The path of the agent is shown as lines and squares. The agent marks the patterns that have been already found as dark zones. The straight lines represent the advance stages, while the squares represent the periods of sniffing or local search.

Note that with respect to the used metaphor, long straight lines show that the agent requires more resources, for instance a larger amount of energy, or, if it were a robot, then it would require a large energy source. Analogous, large searching times mean local searches in large spaces, therefore longer times.

On the other hand, smaller advance times, will cause the agent to require much time to cover the total space. Small searching times will provoke that the agent does not take advantage of the energy that employed in moving in order to search a large amount of goals.

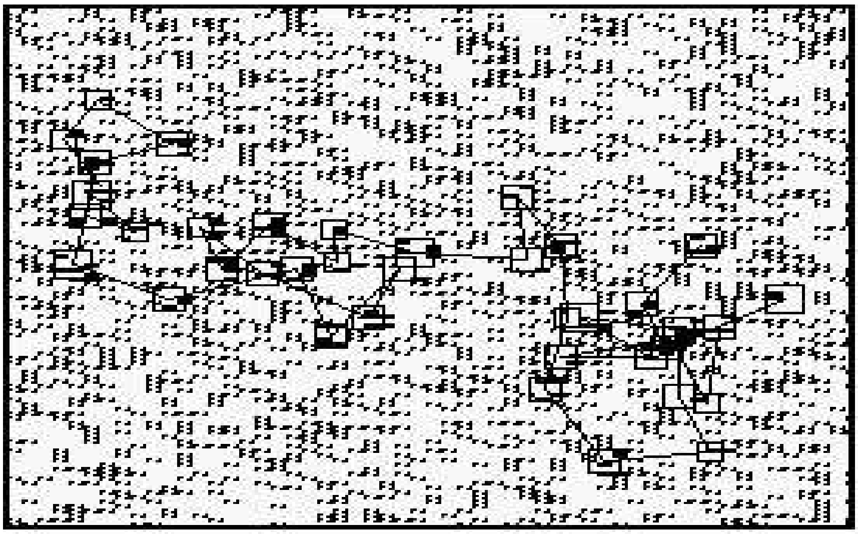

Fig. 2. Execution of the search agent.

3 Performance Test Results

So we were looking for a pair of values of moving frequency and searching frequency that uses smaller resources, and at the same time, maximizes the probability of finding goals.

The test consisted on making run to an agent in a space with 450 different combinations from moving and searching times. The purpose is to find those combinations that generate a large amount of goals.

3.1 Finding Goals in a Space of 500x300 pixels

- It was designed a pattern of 5x2pixels and inserted in a space of 500x300pixels with a density to 1%.
- In the previous space, it was executed an agent with a combination of moving and searching values chosen.
- The moving and searching values are in a range from 10 to 150 units for the advance and of 10 to 300 units for the sniff. The combinations were formed with 10 distance units, in the way (10,10), (10,20), (10,30), etc.
- For each combination of values (move, search) that form the sample, the goals were averaged found by an agent in 10 executions.
- Each execution had a duration of 100 000 iterations.
- Then a chart was built with the number average of goals found by the agent in each one of the combinations (move, search).

- In the previous chart, the maximum and minimum were identified.
- Lastly the pairs of frequency values that produced maxima were graphed in semi logarithmic scale.

The table that contains the group of obtained values of the chart is shown next (Table 1).

Table 1. Average of the goals obtained in 10 executions of 100 000 iterations, in 450 selected combinations (move, search), in a space of 500x300pixels and density of appearance of the pattern of 1%. Column S stands for *Search.*

S	10	20	30	40	50	60	70	80	90	100	110	120	130	140	150
10	78	72.5	68.7	66.2	54.1	50.5	53.3	47	45.9	25.4	21.7	20.5	18	18.7	18.2
20	76.8	94.1	100.1	99.8	97.3	99.4	95.8	96.1	88.4	46.6	44.9	40.5	35.4	33.5	30.9
30	62	85.5	98	102.4	102.3	108.2	110.2	105.1	111.8	60.2	58.8	54.6	51.6	48.6	46.3
40	51.9	78.1	89.4	99.4	103.8	106.5	107.6	108	109.4	69	71	65.7	60.5	59.3	55.4
50	40.8	67.2	80.4	91.6	95.3	105	105.1	105.3	110.2	75	73.3	72.4	70.8	68.2	63.4
60	36.9	63.2	78.5	87.7	94.1	96.9	100	107.3	104.5	81.9	86.8	75.6	71.6	72.4	66.3
70	30.9	57.8	69.6	79.2	88.6	92.5	97.6	104.5	104.9	87.1	86.2	84.3	81.2	78	74.6
80	28	50.6	67.2	80	84	90.4	95.7	98.7	99.6	94.2	89.6	87	84.9	77.5	76.9
90	25.7	48.6	62.4	72.6	80	87	92.7	95.3	96.6	96.2	92.2	92.67	85.5	83.7	83.7
100	40.64	88.45	105.45	108.91	109.64	105.45	107.73	104.09	102.55	95.36	95	92.27	92.36	88.18	87
110	39.18	83.64	108.64	108.27	110.91	106.36	107.55	101	102.82	102.27	97.36	92.73	92.45	90	86.82
120	40.27	88.55	107	107.82	111.45	108.91	109.45	104.18	101.73	99.73	98.55	98.91	95.18	96.18	84.18
130	38.27	88.18	109.91	105.55	112.91	111.36	107.82	106.18	103.45	105.91	102.09	97.64	97.73	95.82	92.45
140	38.36	78.55	104.27	111.27	113.36	114.18	108.27	108.82	103.82	105.45	102.09	98.91	99.36	95	94.55
150	37.36	75.27	98.27	110.91	112.45	112.73	108.73	107.27	108.45	105.27	103.36	97.64	102.55	95.91	97.82
160	33.22	82.22	95.56	108.22	109.11	111.67	110.78	110.78	109.00	103.67	101.78	101.20	103.30	100.50	100.10
170	36.00	71.44	97.56	111.22	112.00	113.33	107.44	108.44	112.44	107.89	102.78	105.00	102.40	101.40	99.80
180	30.56	81.00	101.89	109.00	107.22	115.11	114.11	105.44	109.00	107.89	103.67	103.90	98.60	101.20	100.00
190	31.00	71.56	91.44	109.00	111.67	115.00	110.33	108.11	109.89	106.56	108.67	105.60	105.20	101.90	96.10
200	31.33	69.89	96.33	109.56	110.78	111.00	115.33	112.22	108.78	108.67	105.44	107.30	96.90	104.10	99.80
210	26.11	74.33	92.56	107.67	108.89	110.11	111.00	108.89	108.00	112.22	108.78	109.90	104.60	104.30	104.00
220	31.67	62.56	99.78	102.33	107.78	109.56	114.89	112.56	111.11	108.78	108.56	109.10	106.30	104.20	102.70
230	26.11	70.89	91.56	105.11	111.00	115.78	112.11	112.56	107.56	110.56	110.33	108.70	108.00	104.10	103.20
240	27.78	64.67	89.00	110.67	109.67	115.33	108.67	109.56	114.56	109.22	110.67	109.10	107.30	105.10	103.00
250	20.00	60.22	87.11	108.78	114.33	112.56	111.89	112.56	113.67	111.22	111.90	109.40	108.90	105.20	99.90
260	27.1	70	86.9	108.4	110.6	112.7	111.8	111	108.8	109.3	111.8	109.3	108.2	109.3	105.9
270	29.6	67.7	91.6	99.7	111.7	109.7	111.9	115.6	111.1	110	111.7	109.8	111.2	109.2	108.9
280	27.9	59.5	98.2	101.6	111.8	111.3	110.4	110.8	109.8	110.1	110.7	110.8	113.4	103.7	105.3
290	24.7	53.6	87.7	105.6	99.2	113.3	111.1	112.4	113.8	112.8	111.5	109.1	109.9	110.1	110.9
300	27.9	69.2	92	103.1	107.2	112.4	110.3	113.6	113.1	115.5	110.7	110.2	110.2	109.2	107.8

In Table 1 two max values have been shadowed in the local search address (sniffing) for each advance value. Were selected two max values for each advance value. First max values were shadowed in dark grey, second in light grey. Max values except first and second column, maintain a large distance from the chart beginning. When advance increases they come closer each other more and more. Max values can be identified more easily in the surface graph 3.5 like two mountains in light grey.

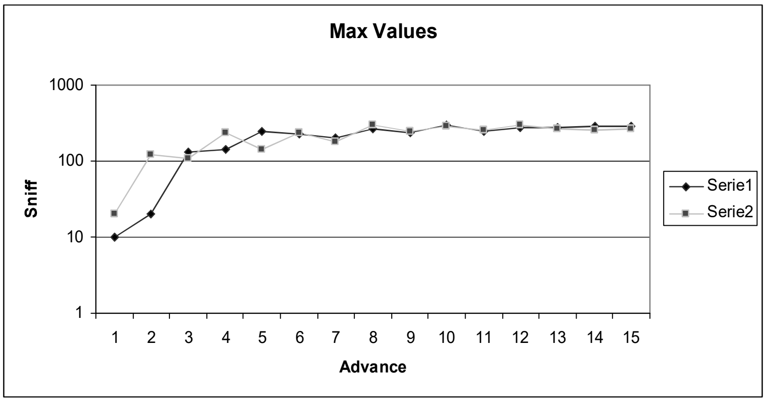

Fig. 3. Finding max values in spaces of 500x300 pixels with densities of pattern appearance to 1%.

In Fig. 4 the combinations that produced maxima have been graphed. We can observe a behavior that resemblance to an exponential one.

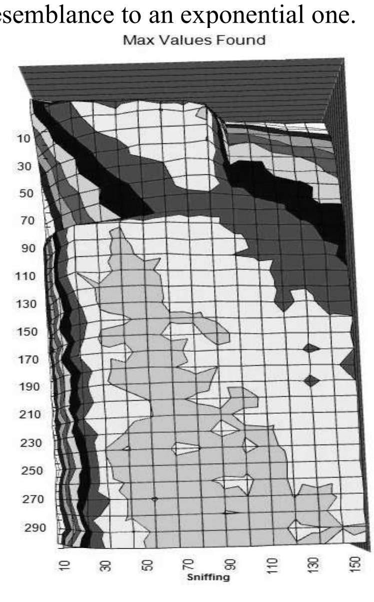

Fig. 4. Relationship of the T_{move} and T_{search} values that generated maxima in spaces of 500x300pixels with densities of pattern appearance to 1%.

In the following experiment, again it has been used a sample of 450 advance combinations and sniffing in a space of 600x400 pixels, with the purpose of assuring that the behavior observed in the previous experiment stays.

3.2 Finding Goals in a Space of 600x400 Pixels

- It was designed a pattern of 5x2pixels inserted in a 600x400pixels space with a density to 1%.
- In the previous space, it was executed an agent with a combination (moving and searching values) selected.
- The moving and searching values are in a range from 10 to 150 units for the advance and of 10 to 300 units for the sniff. The combinations were formed with 10 distance units, in the way (10,10), (10,20), (10,30), etc.
- For each combination of values (move, search) that form the sample, the goals were averaged found by an agent in 10 executions.
- Each execution had a duration of 100 000 iterations.
- Then a chart was built with the number average of goals found by the agent in each one of the combinations (move, search).
- In the previous chart, the maxima and minima are identified.
- Lastly the pairs of frequency values that produced maxima were graphed in semi logarithmic scale.

Table 2. Average of the goals obtained in 10 executions of 100 000 iterations, in 450 combinations (move, search), in a 600x400pixels space and density of pattern appearance to 1%. Column S stands for *Search.*

S	10	20	30	40	50	60	70	80	90	100	110	120	130	140	150
10	87.7	79.4	79.0	76.9	59.7	51.7	51.4	52.6	45.3	26.0	21.8	21.3	18.0	17.8	17.1
20	89.9	109.9	114.0	116.8	117.7	117.6	108.2	107.2	101.0	48.8	43.1	41.1	38.6	35.9	33.0
30	73.3	96.5	117.5	122.3	124.2	133.1	126.6	133.0	124.2	65.9	59.3	60.0	54.0	49.4	48.4
40	64.2	86.0	111.7	116.2	124.0	137.2	133.9	135.7	130.1	78.0	74.1	73.7	65.7	60.8	59.5
50	60.8	79.1	98.0	112.1	119.7	123.3	131.3	130.7	138.7	84.8	82.9	74.1	72.2	73.8	67.9
60	54.2	69.3	90.4	106.1	113.3	121.5	126.9	126.8	132.5	100.5	92.4	86.2	85.4	80.6	80.8
70	49.1	62.5	81.9	95.1	103.1	117.0	125.7	119.7	126.6	107.5	102.1	92.9	91.4	82.3	82.4
80	54.9	58.2	74.4	90.7	103.3	106.3	114.2	121.5	124.2	107.9	103.4	101.9	97.7	93.0	87.9
90	45.9	52.0	67.2	84.4	93.6	102.9	108.4	114.4	123.7	118.7	109.0	101.8	100.1	101.4	96.6
100	42.8	95.6	120.8	135	144.3	136.5	130.8	128.3	120.8	117.3	115.4	108.4	106.5	103	97.4
110	40.5	90.8	128.3	126.3	134.8	143.8	135.7	126.4	125.5	126.2	117.9	115.7	110.4	111.2	103.4
120	38.9	89.6	134.1	143	139.1	141.7	138.5	128.3	132.2	128.5	117.7	113.8	121.8	110.1	113.8
130	37.2	98.9	121	130.9	139.7	140.4	134.5	132.3	131.8	129	122.5	121.2	122.1	112.5	112.7
140	34.6	81.3	118.7	136.7	137.1	143.7	139.4	136.7	132.4	134	130	126.3	123.4	118	113.7
150	35.9	93.3	115.4	141.2	139.6	145.7	144.8	138.6	136.4	134.5	126.6	125.2	123.1	120.9	122.8
160	35.2	80.7	122.4	132.5	146.3	147.8	140.3	143.3	136.4	133.5	129.1	130.3	123.3	120.9	115.6
170	32.9	89.7	119.1	136.5	144	140.3	144.7	144	140	137.4	134.3	130.9	129.4	124	124.9
180	38.2	75.4	115.1	133.7	147.1	150.9	146.2	139.2	144.4	140.4	133.9	128.2	128.4	127.6	121.9
190	32.2	72.9	114.9	122.7	140.5	144.9	140.9	141.9	143.4	141.2	139.8	131.7	133.5	128.4	121
200	28.9	76.3	113.2	134.9	139.2	144.2	149.6	144.2	140.2	140.5	135.1	130.4	133.5	135.7	128.2
210	32.6	72.4	115.8	135.9	142.1	148.6	144.8	144	145.5	138.7	135.9	133	132.3	128	128
220	25.9	68.7	116.6	137.9	145.9	145.5	146.9	145.5	142.6	140.5	143.5	133.6	136.2	130.3	131.2

230	31.3	73	113.6	138.8	150.1	147.9	149.3	144.1	144.4	142.4	141.3	141.8	133.6	133	133.7
240	28.8	75.4	108.6	143.2	136	144.9	146.5	151.9	146	144.4	139.6	143.3	135.5	135	130.9
250	23	68	106.1	133.8	136.2	146.4	153.7	146.7	148.6	143.3	140.2	142.2	137.3	136.1	127.9
260	28.1	69.2	103.5	136.4	150.7	147.2	149.5	142.3	147.8	142.9	141.8	138.4	133.3	135.9	133.4
270	28.2	67.8	104.2	141.6	146.9	148.1	146.5	149.2	145.3	145.8	141.6	141.4	139.1	137.5	135
280	24.2	61.9	106.4	124.9	133.9	148	149.9	151.5	144.4	146.2	136.1	143.1	140.8	139.7	135
290	26.9	65.8	106.8	124.3	147.2	138	151.1	145.1	152.7	151	142.9	139.6	140.2	142	136.2
300	25.8	69.9	102.7	121.8	125.9	147.1	143.5	154.5	144.6	144.2	146.1	143.4	141.4	141.9	137.7

The same result was found in both experiments (3.1and 3.2). In Table 2 two max values have been shadowed in the local search address (sniffing) for each advance value. We selected two max values for each advance value. First max values were shadowed in dark grey, second in light grey. Max values except first and second column, maintain a large distance from the chart beginning. When advance increases they come closer each other more and more. Max values can be identified more easily in the surface Fig. 5 like two mountains in light color.

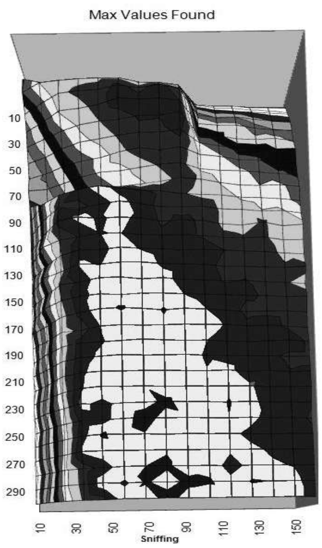

Fig. 5. Relationship of the T_{move} and T_{search} values that generated maxima in spaces of 600x400 pixels with densities of pattern appearance to 1%.

In Fig. 6 the combinations that produced maxima have been graphed. We can observe a behavior that resemblance to an exponential one.

As it is shown in Fig. 4 and Fig. 6, the moving and searching combinations that generate the maximum probability of goals, move from a similar way to an exponential assuring that the phenomenon is of a complex type.

In the following section, the effect of the space size when finding patterns is studied. For this, several spaces of different sizes were designed and in each one an agent was executed with a single moving and searching combination (20, 20), (30, 125), (50, 40) and (70, 30). The result is shown next.

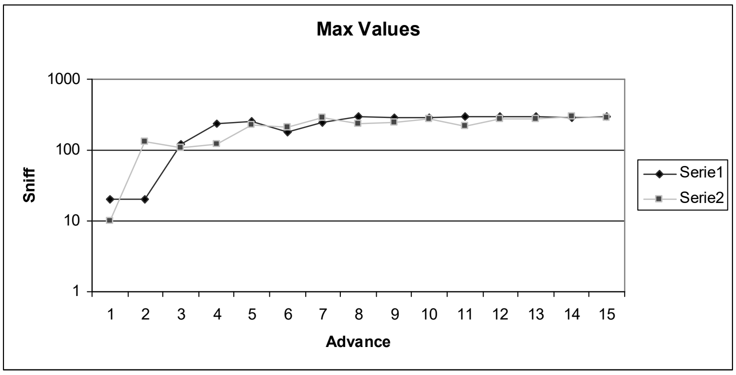

Fig. 6. Relationship of the T_{move} and T_{search} values that generated maxima, in spaces of 600x400pixels with densities of pattern appearance to 1%.

3.3 Maximum Goal Probability according to the Space Size

- Was designed a pattern of 5x2pixels inserted in spaces of different sizes with a density to 1%. The spaces were squared (same height and width).
- In the previous space an agent was executed with a combination of moving and searching values of (20, 20), (30, 125), (50, 40) and (70, 30). Those values were selected because they produced max values in previous tests. Those values appear like restrictions on resources used by the agent.
- For each space and combination 10 agents in an independent way were executed and the goals were averaged among all.
- Each execution had a duration of 100 000 iterations.
- A chart was built with averages of goals found by agent in each space.
- Knowing the total number of patterns inserted in the space, the percentage of goals/patterns was calculated with the purpose of knowing the average space that an agent sweeps with the selected combinations of advance and sniff.
- Lastly the obtained percentages were graphed.

The spaces were designed in several sizes, starting in 50 square pixels and stopping in 300 square pixels. Look that the larger space, the fewer goals found. Although the max goals number was in 50 and 75 pixels, these values were eliminated because it is easy to find goals in so little spaces. Then the value obtained like a maximum was 125pixels.

The results of the experiment are shown in table 3. The column X*X contains the size in square pixels of the space. Column x contains side size in pixels (local search window). The column #goals, contains the number of goals found by the agent. The column #patterns, contains the number of total patterns inserted in the space. Lastly, the column goals/patterns contain the percentage of the goals/patterns that was able to find the agent during the ten executions in each space.

Table 3. Average of the goals obtained in 10 executions of 100 000 iterations, with the moving and searching combination a) (20,20), b) (30, 125), c) (50, 40) and d) (70, 30) in different sizes spaces and density of pattern appearance to 1%.

a) (20, 20)

X*X	X	#goals	#patterns	goals/ patterns
2500	50	4	4	100.00
5625	75	6.5	7	92.86
10000	100	11.5	15	76.67
15625	125	14.4	18	80.00
22500	150	10.5	15	70.00
28900	170	19.3	27	71.48
30625	175	26.3	38	69.21
32400	180	25.8	38	67.89
40000	200	18.4	32	57.50
62500	250	26.2	62	42.26
90000	300	27.9	83	33.61

b) (30 , 125)

X*X	X	#goals	#patterns	goals/ patterns
2500	50	4	4	100.00
5625	75	6.6	7	94.29
10000	100	11	15	73.33
15625	125	14.1	18	78.33
22500	150	11.4	15	76.00
28900	170	19	27	70.37
30625	175	28.7	38	75.53
32400	180	28.4	38	74.74
40000	200	21	32	65.63
62500	250	33.5	62	54.03
90000	300	39.6	83	47.71

c) (50, 40)

X*X	X	#goals	#patterns	goals/ patterns
2500	50	2.85	4	71.25
5625	75	6.2	7	88.57
10000	100	11.4	15	76.00
15625	125	13.7	18	76.11
22500	150	10.8	15	72.00
28900	170	17.4	27	64.44
30625	175	24.7	38	65.00
32400	180	24	38	63.16
40000	200	18.4	32	57.50
62500	250	26.9	62	43.39 %
90000	300	27.1	83	32.65 %

d) (70, 30)

X*X	X	#goals	#patterns	goals/ patterns
2500	50	1.125	4	28.13
5625	75	4.57	7	65.29
10000	100	8.9	15	59.33
15625	125	11.2	18	62.22
22500	150	8.9	15	59.33
28900	170	15	27	55.56
30625	175	20.3	38	53.42
32400	180	18.4	38	48.42
40000	200	14.6	32	45.63
62500	250	19.2	62	30.97
90000	300	21.2	83	25.54

The combinations (20,20) and (30, 125) were chosen being maximums in the tests studied in the previous section. The values (50,40) and (70, 30) too were chosen being maximums but in tests that were not considered for this article. These tests were similar to the previous ones but with a small number of combinations (135 instead of the 450 combinations of test 3.1 and 3.2).

According to the previous result, an agent with capacities similar to the exposed ones in these tests can cover a space of 125 square pixels with the maximum goal probability.

4 Conclusions and Future Works

In this work, we tried to simulate a metaphor of jumping behavior like a search algorithm that an agent uses to find 2-D patterns. Every cycle, the agent is alternating periods of moving and searching (local search). Moving and searching times supposed as restrictions in the resources of a search agent. Therefore, the best results are those that combined maximum number of found patterns (goals) and a minimum of resources.

A large advance frequency makes that the agent goes far away distances before making a search. Big sniffing frequencies (local search), makes the agent execute local searches in large areas. Small advance frequencies will make the agent to take more time in exploring large search spaces. And finally, small sniffing frequencies will not detect the pattern, if this one is bigger than the window (that the agent explores during the sniffing time).

The results showed that the max probability of goals is when the advance frequency is similar or smaller than the sniffing frequency (when the agent has more time for local search than exploring the space). The experiments showed that exist two sniffing values for each moving frequency that generate a maximum (of goals). This agrees according to [2].

The relationship among frequencies (movement and sniff) that generates greater probabilities of finding patterns showed an exponential form, which assures that the nature of the exposed phenomenon is of a complex type.

Actually we use this metaphor to simulate the behavior of several agents looking for patterns in a selected space with and without communication. Results will be shown in next work.

References

1. Kramer, D.L., Mclaughlin. R.L: The Behavioral Ecology of Intermittent Locomotion. AMER. ZOOL., 41, pp. 137-153 (2001)
2. Benichou, O., Coppey, M., Moreau, M., Suet, P-H., Voituriez, R.: Optimal search strategies for hidden targets. Physical Review Letters 94, 198101, pp. 1-4 (2005)
3. Grver, L.: A fast quantum mechanical algorithm for database search. In: Proceedings of the 28th ACM Symposium on the Theory of Computing, pp. 212-219 (1996)
4. Boyer, M., Brassard, G., Hoyer, P., Tapp, A.: Thight bounds on quantum searching. In: Workshop of Phys. Comp. 96, pp. 36-43 (1996)
5. Lorenz, C., Mottl, P., Wattenhofer, R.: Ant Algorithm. Distributed Computing Group, Thesis, Swiss Federal Institute of Technology Zurich.