Method for computing a path between a starting point and an end point on a map representative of a terrain

The present invention relates to a method for computing a path between a starting point and an end point on a map representative of a terrain.

In general, the invention is situated in the domain of navigation assistance systems for agents on a terrain.

A number of navigation assistance systems are known, in particular designed to help vehicles traveling on roads, from roadmaps modeled in vectorial form whereof the objects are associated with a passage cost. In free environments (virtual worlds, autonomous robots), the maps are shown in the form of digital images made up of elementary units that are characterized as "crossable" or "not crossable" or as having an associated crossing cost.

Many algorithms have been developed for seeking an optimized path between a starting point and an end point on such maps.

The algorithm for seeking an optimal path on a graph extracted from a terrain map, called A* algorithm, provides an optimal path in terms of a predetermined cost function. This algorithm has a high computation cost and memory cost, these costs increasing with the number of elementary units processed.

Simplifications to the optimal algorithm A* have been proposed, so as to lighten the processing cost. Examples include the algorithm proposed in the article "Near Optimal Hierarchical Path Finding" by A. Botea et al., published in the Journal of Game Development, no. 1, 2004, pages 7-28, known as the HPA* algorithm, in which the elementary terrain units are grouped together in blocks of elementary units of equal size, and possible passage points are defined between the blocks, the passage points depending on whether the underlying elementary units are crossable. The path is then optimized taking only the predefined passage points into account, which makes it possible to lighten the associated memory and computation burdens. However, this algorithm is only applicable with a situation in which the elementary terrain units are either "crossable" or "not crossable."

However, for certain applications, it is useful to take into account the cost of off-road travel rather than its possibility, when the agent in motion is suited for such off-road travel. This is for example the case for all-terrain vehicles, in particular military or agricultural vehicles, robots, or hikers and video game agents.

In these various scenarios, the elementary units representative of the terrain to be crossed are no longer simply "crossable" or "not crossable," but have an associated crossing or practicability cost, such a cost typically being able to depend on the slope when the terrain is steep, or the depth of a river for example. The cost of crossing an elementary terrain unit by an agent may depend on the type of agent (all-terrain vehicle type, for example) for which the system is intended.

Searching for an optimal path under these conditions quickly becomes unmanageable due to the cost in terms of memory and computation time. The problem must be simplified.

The methods for simplifying the path search algorithm known from the state of the art are not suited for the scenario where the elementary terrain units have a variable associated crossing cost. In fact, the HPA* algorithm mentioned above is only adapted for crossable or not crossable elementary units, which amounts to associating a binary crossing cost with each elementary unit equal to one or zero and a method based on local means (for example such as the D* algorithm) erases the specificities (watch locations, roads, etc.).

There is a need to propose a method for computing a path suited to cases where the elementary terrain units have an associated non-binary crossing cost and the map is large.

To that end, the invention proposes a method for computing a path between a starting point and an end point on a map representative of a terrain, said map being shown in the form of a digital image made up of a matrix of elementary units, each elementary unit having an associated crossing cost value. The method includes the following steps:

- dividing said digital image into blocks of varying sizes according to a breakdown criterion depending on the crossing cost values of the elementary units belonging to one said block,

- associating a crossing cost value with each block,

- obtaining a path between the starting point and the end point, made up of at least two linear segments, optimizing an overall crossing cost equal to a sum of the local crossing costs, each linear segment having an associated local crossing cost, wherein one linear segment connects a first point of a first block and a second point of a second block adjacent to the first block, the second point of the second block being selected according to a distance criterion relative to the first point of the first block.

Advantageously, the method according to the invention includes simplifying the map by breaking it into blocks of varying sizes, which makes it possible to keep significant details and simplify the map only in the areas that are homogenous within the meaning of the breakdown criterion used. Thus, the significant details are preserved, which is not the case for the systematic simplification methods through blocks of fixed size of the state of the art. Advantageously, the path according to the invention is built dynamically, the linear segments making up the path being formed dynamically over time according to a distance criterion, which makes it possible to avoid introducing significant errors into the computed path, despite the variable size of the blocks of the division.

The method according to the invention can have one or more of the features below, independently or in combination:

- the second point of the second block is the point of the second block closest to the first point of the first block according to a predetermined distance;

- the local crossing cost associated with a linear segment is a function of the length of said linear segment and the crossing cost value associated with the first block;

- the method includes, before the step for obtaining a path, a step for extracting a graph showing, for each block of the division, a set of blocks adjacent to said block of the division according to a predetermined neighborhood;

- the method also includes, after the step for obtaining a path, a step for refining the obtained path to obtain a refined path, including the following sub-steps:

  - selecting elementary units that are part of the blocks of varying size crossed by the obtained path,

  - computing a refined path between the starting point and the end point, optimizing an overall crossing cost, said refined path being made up of refined linear segments, one refined linear segment connecting a first point of a first elementary unit and a second point of a second elementary unit adjacent to said first elementary unit, among said selected elementary units;

- the method includes, before the step for dividing into blocks of varying sizes, a step for simplifying said digital image representative of said map consisting of associating each elementary unit with a category from a predetermined set of categories, according to the crossing cost value associated with said elementary unit;

- said simplification step implements computing a histogram of said digital image, and partitioning the histogram into a predetermined number of categories;

- said step for dividing into blocks of varying sizes consists of recursively dividing each block of said digital image, starting from a block equal to the digital image, into four blocks of equal size, until a stop-division criterion has been verified;

- said recursive division is stopped for a given block when an entropy value is below a predetermined threshold value, said entropy value being computed as a function of categories associated with the elementary units making up said given block;

- the association of a crossing cost value with each block obtained by dividing into blocks of varying sizes comprises, for a given block, determining the majority category of the elementary units making up said given block, and the association with said given block of a mean crossing cost value computed for said majority category.

Other features and advantages of the invention will emerge from the description provided below, for information and non-limitingly, in reference to the appended figures, among which:

- figure 1 is a block diagram showing a programmable device capable of implementing the invention;

- figure 2 illustrates an example of a map gridded into regular elementary units and simplified into blocks of varying sized;

- figure 3 illustrates a path computed between a starting point and an end point obtained using the method for computing a path according to the invention;

- figure 4 is a flowchart of one embodiment of a method for computing a path according to the invention;

- figure 5 is an example of a histogram divided into four categories according to a logarithmic scale;

- figure 6 is an example of a division into blocks of varying sizes according to one embodiment of the invention;

- figure 7 is an example of blocks considered to be blocks adjacent to a current block in one embodiment of the invention, and

- figure 8 is a table illustrating the performance of the invention relative to a method of the state of the art.

Figure 1 diagrammatically illustrates the operational blocks of a device for implementing a method for computing a path according to the invention.

The device 1 is for example an onboard computer in a specific portable housing or an onboard vehicle computer that has all-terrain travel capacities.

Such a device has viewing means 2, for example such as a screen, capable of displaying data, for example terrain maps, of the path segments. The device 1 also

includes interaction means 3, which are for example a keyboard or a touchscreen incorporated into the screen 2, allowing a user to provide parameters and set desired starting and end points.

The device 1 also includes a processor or central processing unit 4, able to run control program instructions when the device 1 is powered on. The device 1 also includes means for storing information 5, for example registries, able to store executable code instructions making it possible to run programs able to carry out the method for computing a path according to the invention. The various operational blocks of the device 1 described above are connected via a communication bus 6.

Figure 2 illustrates a simplified example of a terrain map 10 made up of elementary units 12, which in this example are square blocks corresponding to a terrain surface of N x P square meters, with N and P for example comprised in a range [20, 10000].

The map 10 is stored in memory 5 in a device 1 that can implement the invention in the form of a two-dimensional matrix, each elementary unit corresponding to a pixel of that matrix.

In figure 2, the elementary units 12 can have an associated cost value from among three possible values, each possible value being represented by an associated cross-hatching.

The map 14 shown in figure 2 illustrates the result of a step for dividing the map 10 into blocks of varying sizes 16. As will be explained in more detail hereafter, the blocks 16 are obtained by dividing into quadrants or "quad-tree" division. The blocks 16 are of varying sizes, in this example going from a block in the bottom left the size of one quarter of the map 10 to a block with a size equal to the elementary unit. As illustrated in figure 2, each block 16 has an associated cost value, which is shown graphically by cross-hatching, and which is for example equal to the cost value of the majority of the elementary units 12 making up the block 16.

Figure 3 diagrammatically illustrates a path 20 between a starting point 22 and an end point 24, obtained by applying a computation method according to the invention from the simplified terrain map 14 and using the crossing cost values associated with the blocks 16 of the map 14.

In order to better explain the particularity of the invention, figure 3 also illustrates a graph 25 connecting the centers of the adjacent blocks. The graph 25 is the graph that would be used by the traditional path computation methods known in the state of the art. According to the known state of the art, any computable "optimized" path will necessarily

pass through the arcs of the graph 25. One can therefore see that any path would significantly distance the selected path 20, considering that the blocks filled in with dotted lines correspond to elementary units with a maximum crossing cost, i.e. that are uncrossable.

Advantageously, the method according to the invention makes it possible to compute the path 20, which connects the points 22 and 24 by passing through the points 26, 28, 30 and 32, the path 20 being built dynamically.

As can be seen in figure 3, the point 26 is the point of the block 36 that is closest according to the Euclidian distance from the starting point 24 of the block 34, and so forth. Advantageously, the path computed by the method according to the invention is not extended by significant gaps that would necessarily be added by a passage imposed by the centers of the blocks of the division.

Figure 4 shows the main steps for carrying out a method for computing a path according to one embodiment of the invention, typically implemented by a processor 4 of a device 1 capable of implementing the invention.

As input, one has an original map Co, in the form of a digital image as explained above, made up of elementary units u_e each having an associated crossing cost value $V(u_e)$. The digital image representative of Co is made up of a matrix of pixels, each pixel having an associated value V. For example, the value of each pixel is represented on a byte, and can therefore assume whole numerical values between 0 and 255.

In this case, each crossing cost value $c(u_e)$, which is initially supplied between real values C_{min} and C_{max}, is transposed to be shown in the digital image:

$$V(u_e) = \frac{255}{(C_{max} - C_{min})}(c(u_e) - C_{min})$$

Of course, other similar representations, in particular with a higher number of bytes per pixel, can be considered.

The first step 40 implemented is a step for simplifying the representation of the map Co by using a predetermined number of categories Nc. The number of categories is a parameter of the algorithm, which can be predetermined, chosen by an operator or computed dynamically by an analysis of the digital image of the map Co to be processed, as will be briefly explained hereafter.

In fact, it is clear that for example, if the crossing cost value of an elementary unit depends on the slope of the terrain corresponding to the elementary unit, several spatially close elementary units can have similar, but not necessarily identical, associated crossing cost values.

One possible method for obtaining the simplification into a number of categories Nc is the histogram method, consisting of computing a histogram of the image Co, and then dividing the histogram into Nc parts, according to a logarithmic scale, as illustrated in figure 5.

Figure 5 diagrammatically shows an example of a histogram H of a map Co, including the possible cost values on the x-axis, comprised between 0 and Vmax=255 in the above example, and the number of elementary units assuming those values on the y-axis. In this example, the histogram is divided into four categories on a logarithmic scale. In a model in which low values correspond to a low crossing cost and therefore to a relatively easy crossing, while the high values correspond to a high crossing cost and therefore to a nearly impossible crossing, the use of a logarithmic scale has the advantage of allowing a better representation granularity for the areas allowing easy crossing. Thus, all of the elementary units having an associated crossing cost between 0 and V_1 are classified in category C_1, all of the elementary units having an associated crossing cost between V_1 and V_2 are classified in category C_2, and so forth. In general, all of the elementary units having an associated crossing cost between V_{i-1} and V_i are classified in category C_i.

A simplified cost value is associated with each elementary unit according to the category to which it belongs. For each category C_i, it is possible to choose any value of the interval $[V_{i-1}, V_i]$ defining the category C_i, for example the lower bound V_{i-1} or the middle of the interval $V(Ci)=(V_i+V_{i-1})/2$.

Once the map Co is simplified into a number of predetermined categories, a step 42 is applied for dividing the map into blocks of varying sizes, as a function of the categories C_i previously determined.

In the preferred embodiment, the division is carried out using an algorithm for dividing into quadrants, or quad-tree division.

This is a recursive division in which each block is subdivided into four blocks of equal size as long as a criterion for stopping the division is not met. In the first step, the root block is initialized as being equal to the complete image itself.

Each division block is defined by its size and the position in the complete image Co, for example in the upper left corner of said complete image.

A result of a quad-tree division is illustrated in figure 6. The complete initial image denoted Q_0 is processed as the initial block.

This block Q_0 is divided into four blocks or quadrants Q_1 to Q_4. For the division of this example, the stop criterion is verified on the blocks Q_1, Q_3 and Q_4.

The block Q_2 is again divided into four blocks or quadrants Q_{21}, Q_{22}, Q_{23}, Q_{24}.

Once again, the quadrant Q_{22} is subdivided in the following step and so forth, up to a minimum size, here the size of the elementary unit, is reached.

The stop criterion is typically a homogeneity criterion of the processed block, the underlying principle being that it is not useful to divide a homogenous block.

In the preferred embodiment, this stop criterion is based on the comparison of the entropy of the considered block to a predetermined threshold S.

When a current block Bc is processed that is made up of a matrix of elementary units u_e, the computation of the entropy is the following:

$$H(B_c) = \sum_{i \subseteq N_c} H(C_i)$$

Where H(Bc) is the entropy of Bc comprised between 0 and 1 and H(Ci) is computed as follows:

$$H(C_i) = -\sum_{u_e \in B_c} p(u_e|C_i) \times \ln(p(u_e|C_i))$$

The higher the value of a block, the greater the disparity is between the categories of elementary units contained in the block. Conversely, when the entropy is low, the block Bc corresponds to a homogenous terrain area.

The threshold value is chosen for example to be comprised between 0.3 and 0.7, making it possible both to simplify the map and preserve the important details.

Alternatively, it is possible to consider varying the threshold according to the number of categories shown in a block to be processed or the distance between the categories present. Thus, it is considered to increase the threshold when elementary units belonging to different categories that have disparate associated cost values are present in a block.

According to another alternative, other stop criteria based on the cost values associated with the elementary units are considered.

The step for dividing into blocks of varying sizes 42 is followed by a step 44 for extracting a graph G representative of the simplified map.

To build this graph, in this embodiment, for each block of the division, stored as a node of the graph, links are built toward the blocks positioned in the bottom right, as diagrammatically shown in figure 7.

As illustrated in figure 7, each block Bc of the division is linked to a variable number of adjacent blocks Bv according to a predetermined vicinity, which touch Bc either by a side or by a corner. In the example of figure 7, the current block Bc has eight adjacent blocks.

The step 44 for building the graph is followed by an optional step 46 for adding additional information from a road graph.

It is possible to incorporate a road graph into the system by seeking the intersections between the blocks and the roads and using them to assess a passage cost between boxes connected by a road graph.

It is possible to incorporate a road graph into the system by seeking the intersections between the blocks and the roads and using them to assess a passage cost between the boxes connected by a road.

This complete graph corresponding to a model of the terrain in blocks of varying size can then be used to compute a path Path(A,B) between a starting point A and an end point B in step 48.

The starting and end points are typically provided by a user or a user application.

In the preferred embodiment, as briefly explained above in reference to figure 3, the path is computed dynamically.

The path is made up of linear segments, each segment connecting a first point P1 of a current block that is a first block to a second point P2 of a second block, which is one of the blocks adjacent to the current block.

The initial starting point is the point P1=A, which is situated in an initial block of the division. For example, returning to figure 3, the initial starting point is point 22 of the block 34.

All of the adjacent blocks Bv of the current block Bc, as shown in the graph G previously extracted, are considered, and for each adjacent block Bv, the point P2(Bv) that is the point belonging to the adjacent block Bv and closest to the point P1 of the current block is determined, and the length of the segment S=[P1, P2(Bv)], denoted l(S), is computed.

In practice, the point P2(Bv) is situated on the border of Bv closest to the current block Bc. The determination of such a point P2(Bv) is done by a known algorithm for minimizing the Euclidian distance. In a known manner, the shortest path from a point to a straight line follows the straight line that is perpendicular to the straight line and passes through that point.

In the preferred embodiment, each segment thus determined is associated with a local crossing cost $c_{local}(S)$ that is a function of the length of the segment S and the crossing cost V(Bc) associated with the current block Bc.

For example, a proportional computation is used:

$$c_{local}(S) = l(S) \times V(B_c)$$

Alternatively, other functions for computing the local crossing cost associating the length of the linear segment and the crossing cost associated with the block Bc are considered.

To determine the optimal path within the meaning of an overall crossing cost equal to the sum of local costs associated with the linear segments that make up the path, the traditional A* algorithm is used on a simplified graph that is built over time.

Advantageously, the considered linear segments do not necessarily pass through the centers of the blocks of variable size, which avoids introducing significant deviations when the block size is very large. Thus, any error introduced due to simplification of the map into blocks of variable size is reduced.

The step for obtaining a path 48 is followed by an optional refining step 50 for refining the obtained path.

In fact, the path Path(A,B) obtained in step 48 from the simplified graph is obtained quickly, but depending on the level of simplification of the map, this path may not be precise, in particular over the portions approximated by the large blocks.

In one optimized embodiment, it is therefore possible to consider refining the obtained path.

In this embodiment, the refining consists of reloading, in the memory 5 of a device 1 implementing the invention, a set of elementary units E made up of the set of elementary units forming the blocks of the division crossed by the path Path(A,B) previously computed while adding blocks situated on each side of the diagonal lines, so as to allow the refining to find alternative refined segments to the segment passing through a corner connecting two diagonal blocks. Then, an algorithm to search for an optimal path is applied on that set E of elementary units to obtain a refined path $Path_{aff}(A,B)$.

A refined path made up of refined linear segments is then computed between the starting point A and the end point B, optimizing an overall crossing cost. A refined linear segment connects a first point of a first elementary unit and a second point of a second elementary unit adjacent to said first elementary unit, among said elementary units of the set E of selected elementary units. The granularity is finer for such a refined path, since each refined linear segment connects two elementary units.

For example, a traditional algorithm searching for an optimal path is applied, like the A* algorithm, given that the number of elementary units loaded in the memory to be processed is reduced, and the computation time is also low in light of the reduced number of possibilities to be tested.

Advantageously, practical tests have shown the effectiveness of the method proposed by the invention. Thus, the table of figure 8 illustrates the performance of the method proposed by the invention in the "simplified" column relative to the performance of a search for an optimal path in a complete graph of type A* ("complete" column), for a map of 900x700 elementary units.

One can thus see that the number of nodes to be stored in memory to depict the graph is decreased by a factor of 7, and the number of links is decreased by a factor of 8. The time to load the graph in memory is 0.735 seconds for the method according to the invention versus more than 5 seconds for the method of the state of the art. The computation time is decreased by a factor of 8. Thus, one can see that the path obtained by the method according to the invention is 25% away from the optimal path, but the gains in terms of computation time and memory are considerable.

CLAIMS

1.- A method for computing a path between a starting point (A) and an end point (B) on a map representative of a terrain, said map being shown in the form of a digital image made up of a matrix of elementary units, each elementary unit having an associated crossing cost value, characterized in that it includes the following steps:

- dividing (42) said digital image into blocks of varying sizes according to a breakdown criterion depending on the crossing cost values of the elementary units belonging to one said block,

- associating (42) a crossing cost value with each block,

- obtaining (48) a path between the starting point and the end point, made up of at least two linear segments (S), optimizing an overall crossing cost equal to a sum of the local crossing costs, each linear segment (S) having an associated local crossing cost ($C_{local}(S)$), wherein one linear segment (S) connects a first point of a first block (Bc) and a second point of a second block (Bv) adjacent to the first block (Bc), the second point of the second block (Bv) being selected according to a distance criterion relative to the first point of the first block (Bc).

2.- The method according to claim 1, characterized in that the second point of the second block (Bv) is the point of the second block (Bv) closest to the first point of the first block (Bc) according to a predetermined distance.

3.- The method according to any one of claims 1 or 2, wherein the local crossing cost ($C_{local}(S)$) associated with a linear segment (S) is a function of the length of said linear segment and the crossing cost value associated with the first block (Bc).

4.- The method according to any one of the preceding claims, characterized in that it includes, before the step (48) for obtaining a path, a step (44) for extracting a graph showing, for each block (Bc) of the division, a set of blocks (Bv) adjacent to said block of the division according to a predetermined neighborhood.

5.- The method according to any one of the preceding claims, characterized in that it also includes, after the step (48) for obtaining a path, a step (50) for refining the obtained path to obtain a refined path, including the following sub-steps:

- selecting elementary units that are part of the blocks of varying size crossed by the obtained path,

- computing a refined path between the starting point and the end point, optimizing an overall crossing cost, said refined path being made up of refined linear segments, one refined linear segment connecting a first point of a first elementary unit and a second point of a second elementary unit adjacent to said first elementary unit, among said selected elementary units.

6.- The method according to any one of the preceding claims, characterized in that it includes, before the step (42) for dividing into blocks of varying sizes, a step (40) for simplifying said digital image representative of said map consisting of associating each elementary unit with a category from a predetermined set of categories, according to the crossing cost value associated with said elementary unit.

7.- The method according to claim 6, characterized in that said simplification step implements computing a histogram of said digital image, and partitioning the histogram into a predetermined number of categories.

8.- The method according to one of claims 6 or 7, characterized in that said step (42) for dividing into blocks of varying sizes consists of recursively dividing each block of said digital image, starting from a block equal to the digital image, into four blocks of equal size, until a stop-division criterion has been verified.

9.- The method according to claim 8, characterized in that said recursive division is stopped for a given block when an entropy value is below a predetermined threshold value, said entropy value being computed as a function of categories associated with the elementary units making up said given block.

10.- The method according to any one of claims 6 to 9, characterized in that the association of a crossing cost value with each block obtained by dividing into blocks of varying sizes comprises, for a given block, determining the majority category of the elementary units making up said given block, and the association with said given block of a mean crossing cost value computed for said majority category.