RESEARCH ARTICLE OPEN ACCESS

A Hough Transform Based On a Map-Reduce Algorithm

Abdoulaye SERE*, Dario COLAZZO**, Oumarou SIE***

* *Laboratory of Mathematic and Computer Science, University of Ouagadougou, Burkina Faso (e-mail: abdoulaye.sere@univ-ouaga.bf).*

** *Department of Mathematic and Computer Science, University of Paris-Dauphine, France (e-mail: dario.colazzo@dauphine.fr).*

*** *Laboratory of Mathematic and Computer Science, University of Ouagadougou, Burkina Faso (e-mail: oumarou.sie@univ-ouaga.bf)*

ABSTRACT

This paper presents a method that proposes the composition of the Map-Reduce algorithm and the Hough Transform method to research particular features of shape in the Big Data of images. We introduce the first formal translation of the Hough Transform method into the Map-Reduce pattern. The Hough transform is applied to one image or to several images in parallel. The context of the application of this method concerns Big Data that requires Map-Reduce functions to improve the processing time and the need of object detection in noisy pictures with the Hough Transform method.

Keywords: Hough Transform, Map-Reduce, Pattern Recognition, Big Data.

I. INTRODUCTION

Hough Transform is a recognition method introduced by Paul Hough [1] in 1962. Originally the method concerns the detection of straight line in a noisy picture. Hough transform is very known in object detection in noisy pictures : the method is based on two spaces an image space and a parameter space, often called the accumulator which is simply a matrix. In the initial definition of Hough Transform, a point (x, y) in a image space indicates a line b=y-ax of the parameter space where the pairs (a,b) are a set of points of a line in the parameter space.

Many variant of Hough Transform have been proposed in [8, 9, 11]. Standard Hough Transform improves the classic Hough transform in taking a detection of vertical line into account. The standard Hough Transform associate a point (x, y) in a image space to a sinusoid curve p= x. cos (α) +y. sin (α) in a image space. SERE and others in [8, 9] introduced an extension of Standard Hough Transform, based on geometry objects for analytical straight line recognition. Some works have concerned the influence of the image digitalization on the parameter space. Several quantizations of parameter space have been established to improve detection quality. The number of votes to increase in the parameter space that depends on a fuzzy set definition has been proposed by Jayanta Basak and others in [19].

A review of Hough transform has been introduced by Henri Maître [5] to have a generalized definition the Hough transform and in 2014, a survey of Hough Transform has also been studied by P. Mukhopadhyay and others in [18].

In 2006, the generalized preimage has been proposed by Martine Dexet [7] for analytical straight hyperplane [3, 4] recognition. The Dexet‘s method does not consider the quantization of parameter space.

The Hough transform method has been generalized for others shapes detection such as circles, ellipse [16, 17]. Duda and others in [4,10] have also proposed a generalization of Hough Transform to take the recognition of arbitrary objects into account. There are also many applications such as probabilistic Hough Transform [6, 12] proposed in the OpenCV library, iris and characters recognition.

Data analysis becomes more important in scientific domains. The Map-Reduce Framework introduced in 2004, by Dean and others in [13] to process a large data distributed in a cluster of machines for Google : Google uses Map-reduce, to realize some statistical computing in Google translate, in clustering of Google news or to analyze images from satellites. Many works on Map-Reduce Framework have been proposed : Map-Reduce has been used to detect networks congestion, to translate SQL query into the form of Map-Reduce Framework.

Moreover, there are also technologies using Big Data and Map-Reduce Pattern. Hadoop is an implementation of Map-Reduce Framework, written in Java and tested in Yahoo’s cluster, introduced by Apache Software Foundation. There are many no SQL databases used with the Map-Reduce Framework such that MongoDB, Hive. There are also others implementation of Map-Reduce such as Skynet, Disco, FileMap, Themis.

Due to the number of data in scientific domains, it becomes clear to take suitable method considering the processing time. During long time, scientists have worked on the implementation of the Hough Transform method : many methods with the adequate quantization have been proposed in this sense (see a survey of Hough Transform in [18]). The image size and its resolution, the shape complexity play on parameter space size and quantization, increasing the time to perform large data. In [20] J. Illingworth and others expose a survey of efficient Hough Transform methods explaining disadvantages of the implementation of Hough Transform through a large storage and computional requirements because of the large accumulator array : For example, for the determination of q parameters, we need q dimensional space. If each parameter is to be resolved into m intervals, the accumulator array will have m^q cells. If data are constituted of many files of pictures (for instance k files) to be processed by Hough Transform, we'll have a long processing time because the cell number to process becomes k. m^q.

Our method is suitable in the processing of many data, by the Hough Transform at the same time. In classical Hough Transform, we analyze one picture. With several pictures, we go in sequential to process images one by one or to realize threads. Here, we propose an alternative to process all the data at the same time using the Map-reduce Framework. The efficiency and the scability of the Map-reduce Framework have been studied by Dean and others in [13]: In our case the picture is converted to data of pairs (a, b) stored in a file like a file of words proposed by Dean and others in [13].

This paper concerns the need of performance to process many images with the Hough transform method. Our purpose is to use the Hough Transform to detect objects in a large data of pictures and to introduce the benefit of the Map-Reduce Framework to improve the processing time. The object detected could be in any form : that allows to choose the suitable Hough Transform and then the corresponding characteristics of the parameter space.

A definition of Map-Reduce Pattern and Hough transforms are presented in the section 2, named preliminaries. In the section 3, we'll focus on the description of the combinaison (composition) of the Map-Reduce algorithm and the Hough Transform method. The section 4 and the section 5 show respectively an extension and an example of the proposed method.

II. PRELIMINARIES

This section focuses on the Map-Reduce algorithm and the Hough Transform method.

A. Map-Reduce

The Map-reduce framework is constituted of three phases : the map function, the shuffle function and the reduce function. The figure 1 illustrates the phases of Map-Reduce Framework.

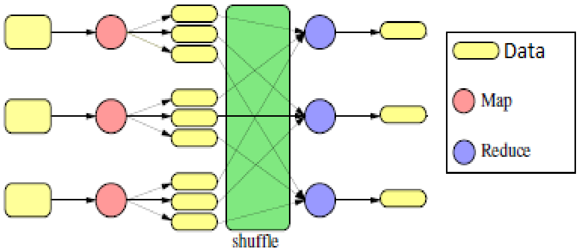

Figure 1 : The Map-Reduce Framework

In our proposed idea, the data in entry are a set of files and each file contains a set of pairs (X, Y).

1) The Map phase: the Map phase consists of a map function.

Definition (**Map function**): A mapper is a function which accepts as input a single key-value pair (k, v) and produces as output a list of key-value pairs $(k_0, v_0), (k_1, v_1), \ldots, (k_{n-1}, v_{n-1}), (k_n, v_n)$.

It takes a pair (k, v) and returns a set of pair (k_i, v_i). The map phase realizes Hough transformation. Let f be this function. We have f(k, v)={(k_i, v_i)…}.

Then, a pair (k, v) is associated to a set of pairs (k_i, v_i).

For example, a shape is a set of point (center of a pixel) (x_0, x_1), of the image. In 2D space the Hough transform is defined by $y_1 = x_1 - (x_0 \cdot y_0)$. The Map function creates for each pair (x_0, x_1), a set of (y_0, y_1) points of the parameter space.

2) The Shuffle phase : this shuffle phase begins after the map function and before the reduce function. The shuffle phase consist of grouping together the pairs (k', v') to produce the pairs that have the same key such as in the example : the pair (w_0 , v_{01}) , (w_0 , v_{02}) becomes (w_0 , <v_{01}, v_{02}>) ; the pair (w_1 , v_{11}), (w_1 , v_{12}) becomes (w_1 , <v_{11}, v_{12}>) ; the pair (w_2 , v_{21}), (w_2 , v_{22}) becomes (w_2 , <v_{21}, v_{22}>)

3) The reduce phase: the reduce phase consist of final functions of Map-Reduce Framework.

Definition (**Reduce function**): A reduce component is a function which accepts as input a key k and a list of values $\{v_0, v_1, \ldots, v_{n-1}, v_n\}$ and produces as output the same key and a new list of values $\{v'_0, v'_1, \ldots, v'_{k-1}, v'_k\}$ with $k \le n$.

If the constraint $k \le n$ is true, there will be a reduction.

Let + be an operator. The reduce function takes a pair (k', <a_0, a_1,..., a_{n-1}, a_n>) to produce a pair (k', $a_0+a_1+...+a_{n-1}+a_n$). All the resulting pairs are stored in an output file.

In the composition of the Map-Reduce algorithm and the Hough Transform method, much flows appear in the shuffle part. The Map-Reduce pattern is a parallellism of a set of map functions followed by a parallellism of a set of reduce functions.

B. Hough Transform

There are several variants of Hough Transform. We present in this section, the Hough Transform method for the detection of straight lines, circles and ellipses named parameter Hough Transform.

1) Hough Transform for straight line : A straight line is a particular case in 2 dimensional space of a straight hyperplane. There are two definitions of Hough Transform for straight line detection. The first one uses the equation : $y_1 = x_1 - (x_0 \,.\, y_0)$ where M (x_0, x_1) is a point of the straight line and the second one, named the standard Hough Transform is based on the equation :

$$\rho = x.\cos(\partial) + y.\sin(\partial) \quad (1)$$

The accumulator is quantified according to the image space dimension. If (x, y) or (x_0, x_1) is fixed data, we have in the parameter space an approximation contraint for each found parameters of the straight line definition.

For classical Hough Transform, we have :

$$(y_0)_{\min} \le y_0 \le (y_0)_{\max} \;,\; (y_1)_{\min} \le y_1 \le (y_1)_{\max} \quad (2)$$

For Standard Hough Transform, we have :

$$\rho_{\min} \le \rho \le \rho_{\max} \;,\; \partial_{\min} \le \partial \le \partial_{\max} \quad (3)$$

The generalized Hough Transform in 2 dimensional space, proposed by Martine Dexet in her thesis in 2006 transforms a point X (x_0, x_1,...,x_{n-1}, x_n) to a set of points Y (y_0, y_1,...,y_{n-1}, y_n) in the parameter space by the relation :

$$y_n = x_n - \left(\sum_{k=0}^{k=n-1} x_k \,.\, y_k\right) \quad (4)$$

In this case, we have an approximation of parameters space that leads to an object in n dimensional space :

$$(y_i)_{\min} \le y_i \le (y_i)_{\max} \quad (5)$$

2) *Hough Transform for circle detection :*

A circle is defined by the equation :

$$(x-a)^2 + (y-b)^2 = R^2 \quad (6)$$

Hence, three parameters (a, b, R) lead to 3 dimensional parameter space such that :

$$a_{\min} \le a \le a_{\max} \;,\; b_{\min} \le b \le b_{\max}$$

and

$$R_{\min} \le R \le R_{\max} \quad (7)$$

3) Hough Transform method for ellipse detection:

The ellipse definition is defined by the relation:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad (8)$$

The parameter is a pair (a, b) verifying

$$a_{\min} \le a \le a_{\max} \text{ and } b_{\min} \le b \le b_{\max} \quad (9)$$

In the classical Hough transform, given the parametric form of a curve in the image space, all possible sets of parameter values are computed in the parameter space, for each object pixel lying on the curve or the line segment. The image resolution leads to many pairs contained in the files. The parameter space quantization has an impact in the terms of the number pairs produced by the map function.

In this paper, we establish a relation between Hough Transform and Map-Reduce by applying the Hough transform in the recognition of objects, precisely straight line or circles, in the context of a lot of data in a file. In the composition of the Map-Reduce algorithm and the Hough Transform, each element plays its role: The Hough Transform in the detection of objects in a noisy picture and the Map-Reduce algorithm in the case of Big Data to improve the processing time.

III. DESCRIPTION OF THE COMPOSITION OF THE HOUGH TRANSFORM METHOD AND THE MAP-REDUCE ALGORITHM

This section shows how to make the composition of the Hough Transform method and the Map-Reduce Framework.

One of the question that appears in using the Map-Reduce framework implemented with the Hadoop is how to represent the complex type, as a class in oriented programming (not elementary type, as string or int). Because Hadoop does not propose a solution for the complex type. Here in our proposed method, the used type is a string. A pair will be represented by a String. A pair (a, b) will be equal to a pair (c, d) if a String (a, b) noticed “(a, b)” is equal to a string (c, d) noticed “(c, d)”. This relation will have an importance in the intermediate phase in order to sort the data resulting of the map function. Our method follows the phases of the figure 2.

We insert into the map function and the reduce function instructions to obtain the data of the accumulator. These instructions will be presented forward in the form of algorithms in the next paragraph.

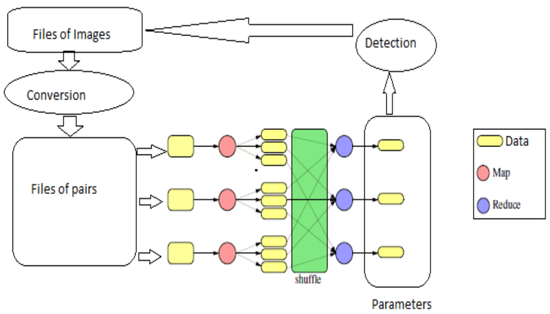

Figure 2: Hough Transform and Map-Reduce

1) Conversion to a set of pairs : At this phase, the set of files is converted to a set of pairs to create entry data for map functions. It consists of reading pictures and to retray the coordinates of its points (pixel centers) located on a contour.

Let I be an image. A shape is a set of points (x_0, x_1,...,x_{n-1}, x_n) representing the center of pixels. All the (n+1) tuples (x_0, x_1,...,x_{n-1}, x_n) point in the shape are converted to a file containing pairs.

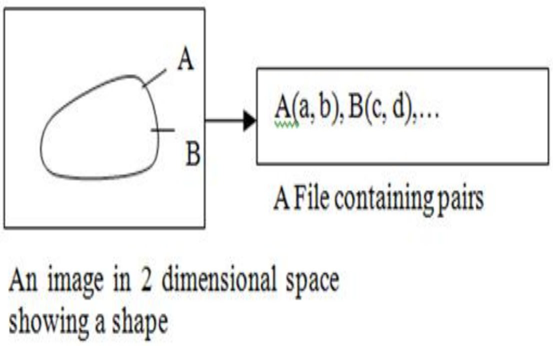

Figure 3 : the file of pairs

In the terms of programming, before setting up the execution of the proposed method, all the image files must be processed to extract contours and to create files containing the pairs (k, v). The following algorithm 1 give instructions about how to represent a point and how to create a list of pixels represented the contour formed by the shape. A contour is a variation of the intensity of the pixels showing separated groups of pixels.

Table I Algorithm of the creation of pairs

Algorithm 1 : creation of the pairs in a file

```
// this function adds a pair to a file
function addFile(string A, File F ){
 Open(F, “W”);
 Write (A, F); // this line adds a pair to a file
close(F);
}
// This function looks for the points on the shape
 // contour  and adds them to a file F.
function imageToFile(Image I, File F){
string result
Open(F, “w”)
 For ( int x=0, x<width, x=x+Δa)
     For (int y=0, y<height, y=y+Δb)
         if (I(x, y) ≥α){
             result=”(“+astring(x)+“, “+astring(y)+”)”
              addFile(result, F)
          }
close(F);
}
int main () {
File S
File I=get(“urlimage1”) ;// initial image
imageToFile(I, S) ;
return 1;
} ;
```

We see in the algorithm only one file of pairs is produced corresponding to one file of image data. Suppose that in the case of several files of image data. It appears necessary to indicate to the map function and to the reduce function the specified file which is currently processing. In this manner, the framework Map-Reduce reduces the data for the same original file because each image space corresponds to a one parameter space.

Suppose that the image data is a set of image files that is I={ I_1, I_2,..., I_{k-1}, I_k } a set of k images. What will be the definition of the map function and the reduce function?. Our method does not work directly on the set of images I. The set of images (I) corresponds to a set of files (F) that contains the data of the contours named F={ F_1, F_2,..., F_{n-1}, F_n }. The number of files noticed card (F) does not depend on the number of images (card(I)) : it is possible to have an image of I associated to a file of F or not. The figure 4 gives a relation between the set I and the set F. A set of images data can also be stored in one file of pairs.

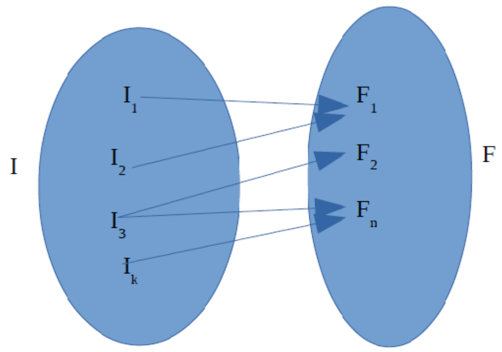

Figure 4: The relation between I and F

The set I is a set of images. The data in each file of F are pairs in the form (I_m , (x, y)) where I_m (x, y) ≥α that means the point (x, y) is on a point of the contour (shape) in the Image I_m . The map function takes a parameter (string key, string values) where key is a variable indicating the name of a file F_i and values is a variable corresponding to the data in the form (I_m , (x, y)) in F_i .

Table 1 Algorithm of the creation of pairs

Algorithm 4 creation of the pairs in the file F, in the case of several images

Let I be a set of images such that
I={ I_1 , I_2 ,…, I_{k-1} , I_k } .
Let F be a set of files of pairs such that
F={ F_1 , F_2 ,…, F_{n-1} , F_n }.
// this function adds a pair to a file
function addFile(string A, File F){
choose a file F_i of F for writting
Write (A, F_i);
close(F_i);
}
// This function looks for the points of the contour of a shape and add //them to a file F.
function imageToFile(Image I, File F){
string result ;
for each (x, y) in a opened file I_k of I for reading
if (I_k(x, y) ≥α){
result="("+"I_k"+", "+"("+astring(x)+", "+astring(y)+")"+")"
addFile(result, F)
}
close(I_k);
}
begin
I={ I_1 , I_2 ,…, I_{k-1} , I_k } .
F={ F_1 , F_2 ,…, F_{n-1} , F_n }.
imageToFile(I, F) ;
end;

2) Map phase : In the map function, we define a quantization to divide the x-axis and the y-axis. The number of cells (a, b) in the accumulator depends on the quantization of the accumulator. More the quantization is smaller more the number of data produced by the map function is important. The map function realizes the Hough Transform method.

The map function reads the pairs (I_m , (x, y)) and produces the pairs ((I_m , (a, b)), 1) that is a pair of type (string, string) where (a, b) verify b=y-ax, $a_{min} \le a \le a_{max}$ and $b_{min} \le b \le b_{max}$. a_{min}, a_{max}, b_{min}, b_{max} are the approximation value of the accumulator to represent a, b : they depend on the quantization of the accumulator. That means in some cases a≈(a_{min} | a_{max}) and b≈(b_{min} | b_{max}).

The intermediate partitioner between the map function and the reduce function analyzes the pairs ((I_m , (a, b)), sorts and groups the pairs that have the same (I_m , (a, b)). We noticed here that (I_m , (a, b)), has a type string. After the partioning and before the beginning of reduce functions, the data look like for example : ((I_1 , (a, b)), <1,1,1>) , ((I_2 , (c, d)), <1,1>) , ((I_k , (e, f)), <1,1,1, 1>)

3) Reduce phase : the reduce function computes all the peak points in the accumulator. It is possible to consider o in the instructions of reduce function, the cells where the vote number is superior to 1 or superior to a value λ (a threshold).
The reduce function will start if all the map function ends.
Before beginning reduce functions, the data are sorted and grouped.

In the ouput file, there are a set of pairs. The size of the list of values is reduced following the constraint k ≤ n (in the definition of the reduce function).

The role of reduce functions is to add values or to perform the values 1. For example, the input value ((I_1 , (a, b)), <1,1,1>) of the reduce function becomes the ouput value ((I_1 , (a, b)), <3>) . In this manner, the ouput file of reduce functions contains original image references, the references of accumulator cells and its votes numbers.

The following tables give details about the Map and Reduce functions in order to realize Hough Transform. Suppose that we have an image in 2 dimensional space, with the size Width x Height.

Table 2 The Map-Reduce Algorithm

Algorithm 2 : the map function and the reduce function in a case of one file.

```
map(String key, String value)
// key: document name
// value: document contents that the differents pairs
begin
  int a ;
  int b ;
  String result ;
for each pair (x, y) in value
  begin
    for a=0 to width with Δa as a step  do
       for b=0 to height  with Δb as a step do
            if (b≈y-ax) // this relation could change according to the // Hough transform definition
                        result  =”(  ”+AsString(a)+”  ,”+ AsString(b)+” )”
                        EmitIntermediate( result, “1”);
               endif
          endfor
    endfor
  end
end

reduce(String key, Iterator values):
// key: a word
// values: a list of counts
begin
 int peak= 0;
 String result
for each v in values
      peak = ParseInt(v)+peak;
endfor
     result =”(“+key+”,“+AsString(peak)+”)”;
     Emit(AsString(result));
End
```

Table 3 The Map-Reduce Algorithm

Algorithm 5 : the map function and the reduce function in a case of several files

```
map(String key, String value)
// key: the document name
// value: the document contents
begin
  int a ;
  int b ;
  String result ;
  String resultfinal ;
for each pair (M, N) in value
X=GETXFloatValue(N)// This function returns the float value x at the position // N[i]
Y=GETYFloatValue(N)// This function returns the float value y at the position // N[j]
    for a=0 to width with Δa as a step do //width is the accumulator width
       for b=0 to height with Δb as a step do // height is the accumulator height
            if (b≈Y-aX) // this relation could change according to the Hough  //transform definition
                        result=”(  ”+AsString(a)+”  ,”+ AsString(b)+” )”
                        resultfinal  =”(  ”+M+” ,”+result+” )”
                        EmitIntermediate( resultfinal , “1”);
            endif
       endfor
  endfor
endfor
end

Reduce(String key, Iterator values):
// key: a word
// values: a list of counts

begin
 int peak= 0;
 String result
for each v in values:
   peak += ParseInt(v);
endfor
result =”( ”+key+”,“+AsString(peak) +” )”;
Emit(AsString(result));
End
```

We notice that Δa and Δb define the quantization of the parameter space (accumulator) : Δa and Δb are respectively the length of the two sides of a rectangular cell of the accumulator.

4) Recognition phase : the remaining phase consist of the detection of pairs in the ouput file. The next algorithm gives instructions to detect the pairs that have a high vote or a number of vote superior to a threshold.

Table 4 The Detection Of Peak Points

Algorithm 3 : the detection of pairs that have maximal vote numbers in the output file.

```
Detection ( File ouputFile )
Variables
          Pair (k, v)
          Set F // this set contains the pairs of parameters
Begin
        Open( ouputFile, “R”)
        While (non EOF(ouputFile)
```

```
            (k, v)<-read(ouputFile);
          if (v>=threshold)
               add(F,(k,v));
      endWhile
     close (outputFile)
end
```

The proposed algorithms focus on the shapes based on two parameter definitions.

IV. EXTENSION OF PROPOSED METHOD

Suppose that the definition of an object takes a generalized parameter in the form $(b_0, b_1,\ldots,b_{n-1}, b_n)$ with k+1 parameters. Let A be the point $(x_0, x_1,\ldots,x_{n-1}, x_n)$, the center of an hypervoxel, in n+1 dimensional image space. The conversion of the point A of the image I in the type string will give the data $(I, (x_0, x_1,\ldots,x_{n-1}, x_n))$. The map function generates the set of data $((I, (b_0, b_1,\ldots,b_{n-1}, b_n)), 1)$. The shuffle phase between the map function and the reduce function groups the data : for example the data $((I, (b_0, b_1,\ldots,b_{n-1}, b_n)), 1)$and $((I, (b_0, b_1,\ldots,b_{n-1}, b_n)), 1)$ produces the data $((I, (b_0, b_1,\ldots,b_{n-1}, b_n)), 1)$,<1, 1>). That is computed by the reduce function obtain the result $((I, (b_0, b_1,\ldots,b_{n-1}, b_n))$, <2>).

The peak points are produced progressively by the reduce functions : we don't need to wait the end of all the reduce functions to start the object recognition in the image space.

V. ILLUSTRATION BY AN EXAMPLE

Suppose that we have three files of image data, having the same size Witdh x Height, named I_0, I_1, I_2. The file I_0 contains the data (entry data) : (1, 1), (2, 2), (3, 3), (4, 4), (5, 5) ; the file I_1 contains the data : (1, 1), (2, 1), (3, 1), (4, 1), (5, 1) and the file I_2 : (1, 1), (1, 2), (1, 3), (1, 4), (1, 5). In each file I_0, I_1, I_2 , the entry data representation gives a straight line in the image space. In parameter space, we consider the equation p=x cos (α)+ y sin (α). That gives the following representation respectively for I_0, I_1, I_2 :

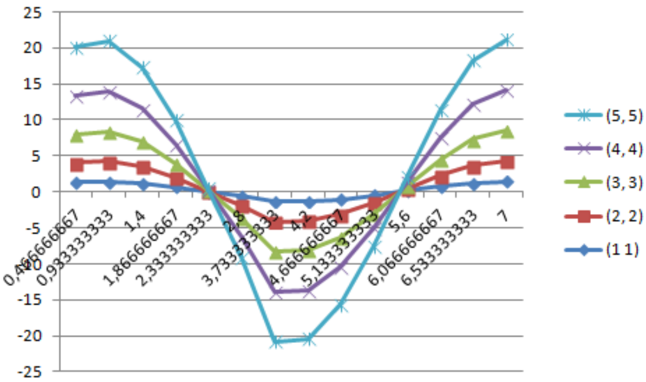

Figure 5 : the points generated by the map function for the image I_0

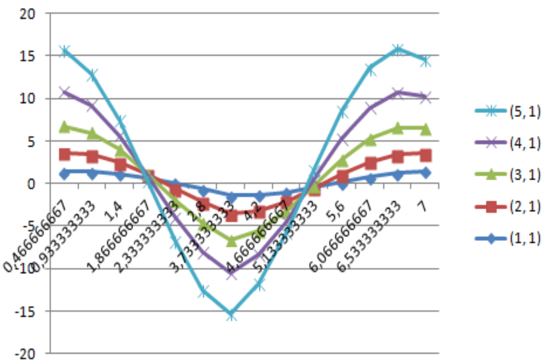

Figure 6 : the points generated by the map function for the image I_1

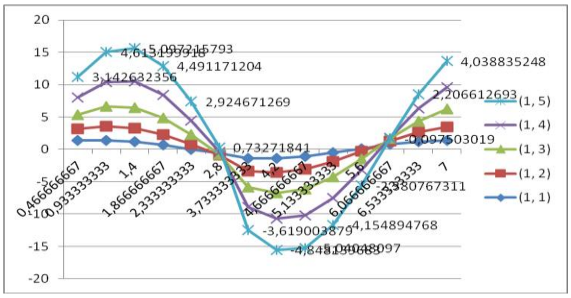

Figure 7 : the points generated by the map function for the image I_2

Each image is read by one map function : with the three images, we are going to use three map functions.

The data of these graphics are summarized in the below table :

	Δα		1	2	3	4	5	6	8	9	10	11	12	13	14	15
	0,466667		α													
	x	y	0,46666667	0,93333333	1,4	1,86666667	2,33333333	2,8	3,73333333	4,2	4,66666667	5,13333333	5,6	6,06666667	6,53333333	7
	1	1	1,34298483	1,39876686	1,15541687	0,66497625	0,03232774	-0,60723419	-1,38777756	-1,36183659	-1,0446613	-0,50408091	0,14429924	0,76182041	1,21642317	1,41088885
	2	2	2,68596967	2,79753373	2,31083375	1,32995251	0,06465548	-1,21446838	-2,77555511	-2,72367319	-2,0893226	-1,00816183	0,28859848	1,52364082	2,43284634	2,82177771
	3	3	4,0289545	4,19630059	3,46625062	1,99492876	0,09698323	-1,82170257	-4,16333267	-4,08550978	-3,13398391	-1,51224274	0,43289772	2,28546124	3,64926951	4,23266656
	4	4	5,37193934	5,59506745	4,62166749	2,65990502	0,12931097	-2,42893676	-5,55111022	-5,44734638	-4,17864521	-2,01632366	0,57719696	3,04728165	4,86569268	5,64355541
I_0	5	5	6,71492417	6,99383432	5,77708436	3,32488127	0,16163871	-3,03617095	-6,93888778	-6,80918297	-5,22330651	-2,52040457	0,7214962	3,80910206	6,08211585	7,05444427
	1	1	1,34298483	1,39876686	1,15541687	0,66497625	0,03232774	-0,60723419	-1,38777756	-1,36183659	-1,0446613	-0,50408091	0,14429924	0,76182041	1,21642317	1,41088885
	2	1	2,23605779	1,99392546	1,32538402	0,37340377	-0,6584304	-1,54945653	-2,21774853	-1,85209742	-1,09036769	-0,09545836	0,91986512	1,73847168	2,18529896	2,16479111
	3	1	3,12913074	2,58908406	1,49535116	0,08183129	-1,34918854	-2,49167887	-3,0477195	-2,34235824	-1,13607407	0,31316418	1,695431	2,71512295	3,15417475	2,91869336
	4	1	4,02220369	3,18424266	1,6653183	-0,20974119	-2,03994668	-3,43390121	-3,87769048	-2,83261906	-1,18178046	0,72178673	2,47099688	3,69177422	4,12305054	3,67259562
I_1	5	1	4,91527665	3,77940126	1,83528544	-0,50131368	-2,73070482	-4,37612355	-4,70766145	-3,32287988	-1,22748684	1,13040928	3,24656275	4,66842549	5,09192633	4,42649787
	1	1	1,34298483	1,39876686	1,15541687	0,66497625	0,03232774	-0,60723419	-1,38777756	-1,36183659	-1,0446613	-0,50408091	0,14429924	0,76182041	1,21642317	1,41088885
	1	2	1,79289671	2,20237513	2,1408666	1,62152499	0,75541362	-0,27224604	-1,94558414	-2,23341237	-2,04361622	-1,41678438	-0,4869674	0,54698955	1,46397055	2,06787545
	1	3	2,24280859	3,00598339	3,12631633	2,57807373	1,47849951	0,06274211	-2,50339072	-3,10498814	-3,04257114	-2,32948784	-1,11823404	0,3321587	1,71151793	2,72486205
	1	4	2,69272048	3,80959165	4,11176606	3,53462247	2,20158539	0,39773026	-3,0611973	-3,97656391	-4,04152605	-3,2421913	-1,74950067	0,11732784	1,95906531	3,38184865
I_2	1	5	3,14263236	4,61319992	5,09721579	4,4911712	2,92467127	0,73271841	-3,61900388	-4,84813968	-5,04048097	-4,15489477	-2,38076731	-0,09750302	2,20661269	4,03883525

Figure 8 : the output data of the map function

The quantization of parameter space plays on the number of data produced by the map functions.

Moreover, if we change the approximation of the value p, the curves in parameter space will change as follows for example.

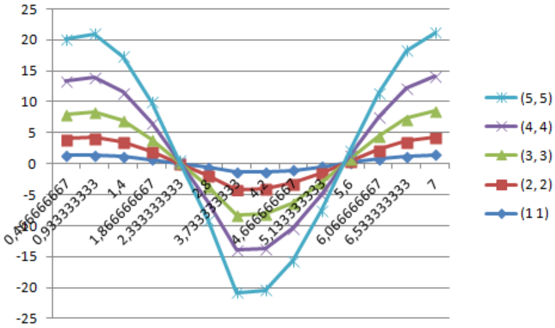

Figure 9 : the points generated by the map function for the image I_0

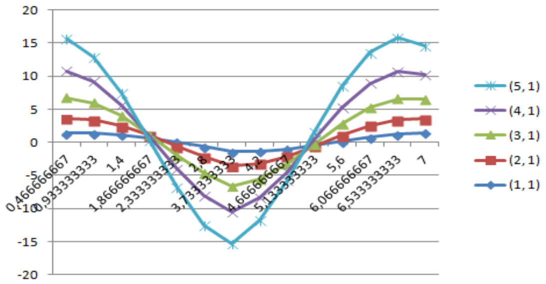

Figure 10 : the points generated by the map function for the image I_1

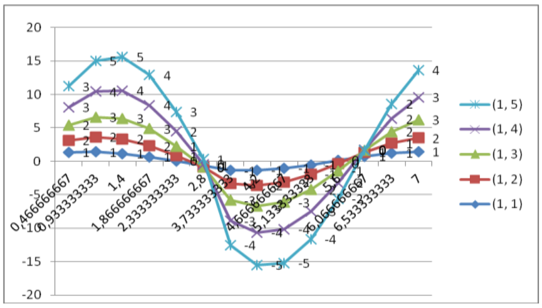

Figure 11 : the points generated by the map function for the image I_2

A quantization of the parameter p creates several groups of values p. The figure 12 gives the values α (we notice that the curves in parameter space are periodic) in taking the approximation of the values p into account, the pairs that will be computed by the reduce functions to obtain peak points in parameter space : suppose that the threshold is 3, for I_0, the reduce function will produce ((2.333, 0) 5), for I_1 we will have ((1.4, 1),3) and ((1.8666,0)3) and for $I_{2,}$ the pair ((2.8,0) 3).

	Δα		1	2	3	4	5	6	8	9	10	11	12	13	14	15
	0,466667		α													
	x	y	0,46666667	0,93333333	1,4	1,86666667	2,33333333	2,8	3,73333333	4,2	4,66666667	5,13333333	5,6	6,06666667	6,53333333	7
	1	1	1	1	1	1	0	-1	-1	-1	-1	-1	0	1	1	1
	2	2	3	3	2	1	0	-1	-3	-3	-2	-1	0	2	2	3
	3	3	4	4	3	2	0	-2	-4	-4	-3	-2	0	2	4	4
	4	4	5	6	5	3	0	-2	-6	-5	-4	-2	1	3	5	6
I_0	5	5	7	7	6	3	0	-3	-7	-7	-5	-3	1	4	6	7
	1	1	1	1	1	1	0	-1	-1	-1	-1	-1	0	1	1	1
	2	1	2	2	1	0	-1	-2	-2	-2	-1	0	1	2	2	2
	3	1	3	3	1	0	-1	-2	-3	-2	-1	0	2	3	3	3
	4	1	4	3	2	0	-2	-3	-4	-3	-1	1	2	4	4	4
I_1	5	1	5	4	2	-1	-3	-4	-5	-3	-1	1	3	5	5	4
	1	1	1	1	1	1	0	-1	-1	-1	-1	-1	0	1	1	1
	1	2	2	2	2	2	1	0	-2	-2	-2	-1	0	1	1	2
	1	3	2	3	3	3	1	0	-3	-3	-3	-2	-1	0	2	3
	1	4	3	4	4	4	2	0	-3	-4	-4	-3	-2	0	2	3
I_2	1	5	3	5	5	4	3	1	-4	-5	-5	-4	-2	0	2	4

Figure 12 : the parameter p values (in green color) detected by the reduce functions in each image .

The time complexity of computing pictures to a file of pairs is $O(n^2)$. The recognition phase is realized in O(n) according of the number of peak points. Finaly, the time complexity depends on the number of pixels verifying the condition $I(x, y) \geq \alpha$, the quantization of parameter space, included in the time complexity of Map-Reduce algorithm. The same analysis might be generalized to others Hough Transform following the Map-Reduce Framework.

VI. CONCLUSION

The definition of the Hough Transform method based on a Map-reduce algorithm, to detect straight lines in multiple noisy pictures has been presented in this paper.

In perspective, the Hadoop implementation of the Hough transform method for the detection of circles and ellipses and the generalized Hough Transform still remain to do, following the proposed Map-reduce algorithm, in the case of large data.

A study of Hough Transform in relation with the fuzzy set or the gröbner ‘s basis might give interesting results.

REFERENCES

[1]. P. V. C Hough, *Method and means for recognizing complex patterns,* In U. S. Patent 3069654, 1962.
[2]. E. Andres, *Discrete linear objects in dimension n : the standard mode,* Graphical Models, 2003, pages 92–111,
[3]. E. Andres, R. Acharya, C. Sibata, *Discrete analytical hyperplanes*, Graphical Model and Image processing, Volume 59, Issue 5, 1997, Pages 302–309.
[4]. D.H.Ballard, *Generalizing the hough transform to detect arbitrary shapes,* Pattern Recognition, Vol. 13, No. 2, 1981, pp. 111–112.
[5]. H. Maître, *Un panorama de la transformée de hough - a review on hough transform*, In traitement du signal, volume 2, issue 4, 1985, pages 305–317.
[6]. M Bruckstein, N.Kiryati, Y.Eldar. A probabilistic hough transform. Pattern Recognition, 24(4), 1991, Pages 303–316,.
[7]. M.Dexet, *Architecture d’un modéleur géometrique à base topologique d’objets discrets et methodes de reconstruction en dimensions 2 et 3,* Thèse de Doctorat, Université de Poitiers, France, 2006.
[8]. A. Sere, O. Sie, E. Andres, *Extended Standard Hough Transform for Analytical Line Recognition*, International Journal of Advanced Computer Science and Applications, Vol. 4, No. 3, 2013, Pages 256-266.
[9]. A. Sere, O. Sie, S. Traore, *Extension of Standard Hough Transform based on object dual*, Journal of Emerging Trends in Computing and Information Sciences, Vol. 6, No. 1, 2015, Pages 20-24
[10]. D.Ballard, *Hierarchical generalized Hough transform and line segment based generalized Hough transforms*, In pattern recognition, volume 15, 1982, pages 277–285.
[11]. J. Cha , R. H. Cofer , S. P. Kozaitis, *Extended Hough transform for linear feature detection*, In Pattern Recognition, v.39 n.6, 2006, p.1034-1043
[12]. J. Matas, C. Galambosy and J. Kittler. *Progressive probabilistic hough transform, Transform*, Volume: 24, n.4, 1991, Pages: 303-316
[13]. Jeffrey Dean and Sanjay Ghemawat, *MapReduce: Simplified Data Processing on Large Clusters*, Symposium on Operating System Design and Implementation (OSDI), 2004, Pages 137– 150
[14]. Xiaoyu Sun, *an enhanced self-adaptive mapreduce scheduling algorithm*, master's thesis
[15]. Bingsheng He, Wenbin Fang, Naga K. Govindaraju ,Qiong Luo, Tuyong WangMars, *A MapReduce Framework on Graphics Processor*
[16]. Robert A. Mclaughlin, Randomized *Hough Transform : Improved Ellipse Detection with Comparison*. In Pattern Recognition Letters, volume 19, issues 3–4, 1998, pages 299–305.
[17]. Ioannou D., Huda W., Laine A. *Circle recognition through a 2d hough transform and radius histogramming.* In Image and Vision Computing, volume 17, issue 1, 1999, pages 15–26.
[18]. P. Mukhopadhyay, B.B. Chaudhuri, *A survey of Hough Transform*, Pattern Recognition (2014), http://dx.doi. org/10.1016/j.patcog.2014.08.027
[19]. Jayanta Basak, Sankar K. Pal, *Theoretical quantification of shape distortion in fuzzy Hough transform*, In Fuzzy Sets and Systems 154 (2005) 227 – 250
[20]. J.Illingworth, J. Killer, *A survey of efficient Hough Transform methods,1987,* AVC 1987 doi:105244/C.1.43