

REALTIME GESTURE RECOGNITION

BACKGROUND OF THE INVENTION

5 1. Field of the Invention

The present invention relates to the realization of real-time gesture recognition which uses a unique recognition method to match symbols and lines with database in real-time.

10

2. Description of the Related Art

15

20

User interface is becoming an essential element in the consumer electronic equipment. Many consumer electronic equipment today uses remote controls/touch screen to interact with the product, but increase of the complexity and functionality of the current consumer electronic equipment such as portable multimedia player, car audio system and other product with multiple functions (e.g., video playback, audio player, FM Radio, DTV, photo viewing, mobile phone, personal organizer and Navigation system) has made it more and more difficult for user to utilize the full advantage of the feature offer. To navigate between each feature and configuration is becoming a time-consuming menu operation.

25

However, the variety of physical layout of button for each remote control and graphical user interface layout has raised numerous problems: the difficulties in locating the required remote control and command, the confusion with the button layout, the replacement issue and so on. Recent improvements of touch screen and graphical user interface design technology have big impact in the market. The designers have made efforts to develop a more user-friendly graphical user interface but still the fundamental menu style navigation and operation have its limitation and problem.

30

35

The problem of current graphical user interface is that it requires the user to select a sequence of the menu option selection in order to perform a particular action. Another way is by displaying the icon that represents the action onto the working screen area for the user to click/press to perform specific action but this raises some question on how many icons being displayed on the working screen is efficient. The other approach is to provide some sort of user profiling by learning the frequency of the user menu use and hide the less frequent user menu. But this approach only works

with small number of people.

People frequently use speech and gesture to communicate with other people. The gesture is used by all people with different culture and language. Advance of the gesture technology has led to possibility of real live commercial application/product.

5 The current technologies use the neural network approach to learn and match the pattern of the gesture. This approach requires large amount of the processing power, learning time and storage and has a higher cost to use it in consumer electronic equipment.

Unfortunately, while there are many human gesture recognition technologies
10 being developed, there is very limited number of commercial products which use human gesture to interface with the product.

Lastly, each user is probably unique and has own needs. It will be advantageous to let the user to define his/her gesture to interact with the system. It provides a new way for user to redefine how to interface with the system. The general
15 gesture solution may not be optimum for particular user requirement.

SUMMARY OF THE INVENTION

In view of the foregoing, it is an object of the present invention to provide a
20 symbol gesture recognition solution to recognize the user symbol/gesture input using finger/pointing device.

It is another object of the invention to reduce the complexity and resource use and achieve the gesture recognition with high accuracy and response.

It is another object of the invention to achieve the user-friendly interface and
25 reduce the time for user to navigate and control task.

To solve the above problems, according to the present invention, a real-time symbol gesture recognition method is shown in Figure 2.

(1) Symbol Gesture Recognition

- a. De-hooking, removes the unintentionally created hook at the end
30 and beginning of the line.
- b. Smoothing process to reduce the digital artifact generated by the input device.
- c. Feature Extraction, which extracts the feature vector based on the orientation of each point.
- 35 d. Normalization process is to normalize the input symbol/line and make it uniform that can eliminate the problem of unpredictable

user input.

- e. To check the intersection within the line data and compare with database.
- f. Line intersection is checked to match the symbol database for matching.
- g. Match processing is done after the final point is received, based on the available data from previous process
- h. Recognizing a symbol which comprises one or more lines.

(2) Symbol and line database is created to store the reference feature vector of the line and combination of the line which is a symbol.

(3) Layout map is to layout the screen/user interface into deferent region. Each region is then assigned to UI type and gesture type with corresponding actions to perform.

According to an embodiment of the present invention, there is provided a gesture recognition method for an electronic system built with a processor, a memory, a screen and an input device for accepting coordinates of drawn lines, the method comprising: (a) accessing the input device to determine a start, end and line data points comprising coordinate position, timing and one or more start points; (b) real-time de-hooking step of removing a hook from a beginning and end of the line data points; (c) real-time smoothing step of removing irregular variations in a handwriting style; (d) real-time feature extraction step of finding an orientation between each pair of points; (e) normalizing an original feature points of the line in the memory to obtain a normalized feature points; (f) checking a connection between each pair of the original feature point; (g) checking an intersection between each pair of the original feature points; (h) analyzing a curve and segmenting a line; (i) matching process which comprises comparing a feature vector of a current line with a reference feature vector in a database; and (j) recognizing a symbol which comprises one or more lines.

The step (b) may comprise: locating abrupt changes in turning angles at the beginning and end of the line while the hook length is less than a given threshold, the threshold being a percentage of the diagonal length of the screen.

The step (c) may comprise: substituting a coordinates of the original line data points with a weighted sum or an average of the neighboring line data points, to eliminate noise and erratic handwriting styles.

The step (d) may comprise: slope estimation between each pair of points and mapping the slope to an orientation chart to obtain an orientation code of that

segment of line between two points.

The step (d) may further comprise: finding a length between the pair of points; and accumulating the length if the orientation code of successive points are same, or storing the orientation code and the point as original feature point when the
5 length is greater than the threshold.

The step (e) may comprise: restoring a problem of writing direction dependency by recalculating the original feature points and reordering the recalculated feature points to obtain the normalized feature points.

The recalculating in step (e) may comprise: re-computing the original
10 feature points, such that the distance between each pair of original feature point is constant, in accordance to Pythagoras theorem.

The reordering in step (e) may comprise: cyclically re-arranging all the recalculated feature points from a fixed reference point; and recalculating orientation codes with respect to the normalized feature points.

The step (f) may comprise: detecting a presence of a connected line segment
15 between each pair of original feature points.

The step (f) may further comprise: combining the orientation codes of the normalized feature points with the corresponding connection presence; and storing the combined result in the feature vector.

The step (g) may comprise: determining the intersection point between two
20 pairs of original feature points; if intersection exists, obtaining the index of the original feature points and appending the index into the feature vector.

The step (h) may comprise: checking for a possibility of curves among the extracted orientations and segmenting line orientations into curves or lines or a
25 combination of both.

The step (i) may comprise: matching the feature vector of the line with the reference feature vector stored in a line database; if a match exists, outputting the line index number of the line.

The step (i) may further comprise: accumulating a list from a symbol
30 database consisting of the present line index number.

The step (i) may further comprises: matching a second/or more line index number to the accumulated list and satisfying the intersection criteria between lines to ascertain the symbol index number.

The line database may consist of line index number, the orientation code
35 information, connection information and intersection information.

The symbol database may consist of symbol index number, line index

number and intersection criteria between lines.

The user interface on the screen may be mapped into layout map that comprises: regions list, input type and symbol list for each region.

5 As the nature of the symbol gesture recognition, the recognition process can start from the first input coordination. It will increase the recognition response rate by distributing the recognition process over time slide of each input.

Each user interface for particular application has particular layout and UI and operation. The layout map comprises each application layout map and action map. The gesture recognition unit then checks the layout and action map to determine
10 the user input type and action to be performed. There are two input type: gesture type and UI type. UI type input is typically used for general UI operation such as button click, slider bar vertical and horizontal scroll, double click and press and drag. The gesture type input is symbolic draw of the symbol to represent a particular action.

15 The database is to store the reference symbol and line data into two tables as shown in the Figure 7 and Figure 8. The symbol structure will store symbol unique ID and the combination of the lines. The line structure stores the feature vector of the line.

BRIEF DESCRIPTION OF THE DRAWINGS

20

Figure 1 shows the overall process of the algorithm used in the present invention for the real-time gesture recognition.

Figure 2 outlines the steps associated with preprocessing and feature extraction used in the algorithm of the present invention.

25 Figure 3 explains the Line Matching method used in the algorithm of the present invention.

Figure 4 shows the steps involved in the Symbol Matching method used in the algorithm of the present invention.

30 Figure 5 shows a screen layout, where the Gesture and UI area are not overlapping, as an example for the present invention.

Figure 6 shows a screen layout, where the Gesture and UI area are overlapping, as an example for the present invention.

Figure 7 shows the structure of the Line Database, consisting of the Header and line data, which is used in the present invention.

35 Figure 8 shows the structure of the Symbol Database, consisting of the Line Index Number 1, Line Index Number 2, Intersection and Symbol Index Code, which

is used in the present invention.

Figure 9 shows the inputted line data after the preprocessing steps of de-hooking and smoothing, as an example for the present invention.

5 Figure 10 shows the two orientation charts suitable for use by the present invention.

Figure 11 shows the inputted line data after the feature extraction and normalization, recalculation and reordering steps, as an example for the present invention.

10 Figure 12 shows an example to illustrate the necessity of checking the line intersection criteria in the present invention.

Figure 13 shows an example to illustrate the necessity of checking the symbol intersection criteria, i.e. intersection between lines, in the present invention.

Figure 14 shows an example of some simple gesture, of one or more lines, that can be recognized by the algorithm of the present invention.

15 Figure 15 shows an example to illustrate the necessity of the connection checking step of the algorithm in the present invention.

Figure 16 is a Block diagram showing the interaction between the symbol gesture recognition units with other components, for the present invention.

20 DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention of recognizing gesture's on an electronic system with a screen deals with applications like a touch panel input for a car audio system. For example, the functions for operating the video, audio and picture applications, like
25 play, pause, rewind, fast-forward, etc, can be operated by simple gesture on the touch panel, which is recognized by the algorithm in this invention. Unlike other methods, which are developed for a particular set of gesture or symbols, this invention is designed to be robust enough to allow arbitrary symbols so that each user can design their own database, based on their handwriting styles.

30 Figure 16 shows the interaction of the user and electronic system using the application 1, 2 and 3 1601-1603 in a typical embedded system with RTOS and input device.

To interface with system, user needs to use the input device via pointing device 1609, touch screens with touch sensor 1610 and other pointing input device
35 1611. As the user uses the input device, the input devices then sends interrupt to RTOS through driver 1612 and activate the symbol gesture recognition unit 1605 to

start getting coordination data from driver 1612. The symbol gesture recognition unit sends the first coordination to the UI manger 1604 to determine the input type based on the current layout map 1607. The UI manger responds to symbol gesture recognition unit 1605 with the type input. If the input type is gesture type, Symbol
5 gesture recognition unit 1605 processes all the input data until user releases the input device. The symbol gesture recognition unit 1605 generates result and sends to current active application to perform action according to layout map 1607 specification.

This invention supports two types of area: Gesture type and UI type. As
10 shown in Fig 5 and Fig 6, the screen is divided into Gesture and UI Area. To support both these areas on the same screen, each screen is associated with a layout map. A specific user interface has a specific layout map associated with it, having the layout of the gesture and UI area. To infer whether the input drawn on the screen is a Gesture or UI, the starting point coordinates are analyzed with the coordinates of
15 each area on the map. In Fig 5, the Gesture and UI area do not overlap, but in Fig 6, the Gesture and UI area overlap. Hence, the layering concept is used in specifying the coordinates of each area on the map. The area with more priority is specified first, followed by the lesser priority area, thereby, the starting point coordinate analysis is done, by satisfying the priority criteria. In Fig 6, the UI area, being the top-most layer,
20 has more priority than the Gesture area.

Most of the functions for operating the video, audio and picture applications can be controlled using Gestures. But for real-time response of the application, UI operations are necessary. This invention categorizes the input from the touch panel into Gestures, Special Gesture or UI operation and implements the algorithm
25 associated with each category.

Since this invention has the ability to recognize symbols with one or more lines, two databases are required. Line database, Fig 7, is used for storing and matching line data and symbol database, Fig 8, consists of Line index numbers. These databases can be created by the user based on the preference of symbols for
30 each operation depending on the application, consisting of one or more lines. The feature vector corresponding to each line is stored in the line database. Each feature vector consists of line index number, the orientation code information, connection information and intersection information along with the packet size. For storing line information in the line database, the line drawn on the touch panel is analyzed based
35 on the gesture recognition algorithm steps and the feature vector is stored in the database file. For storing symbol information in the symbol database, lines are first

drawn on the touch panel, which is recognized and matched by the algorithm. The line index numbers, along with the line intersection information are stored in the symbol database. Thus, the symbol and line databases are used in the matching process of gestures drawn on the touch panel.

5 This invention deals with real-time recognition of the gestures written on the touch panel, as shown in Fig 1. As soon as the first point (starting point) of the line is obtained by the algorithm from the touch panel controller, the processing of the data starts, and the matching is done immediately after the last point (end point) is obtained. Hence the algorithm processes line data without any delay, in real-time, and
10 does not wait for the last point (end point) to begin the processing.

 As shown in Fig 1, the starting point of the coordinate data obtained from the touch panel is analyzed based on the layout map. A decision is made 104 whether the point is in the UI area or the gesture area. If the starting point is found to be in the UI area 103, a check is done on the number of starting points and categorized
15 accordingly. If the number of starting points is found to be one, then if the end point is obtained immediately, without any data points, then it is determined to be a click, else if there are data points then it is a scroll. If there are two starting points, the data is analyzed to be a pinch action.

 If the starting point does not lie in the UI area, then it must lie in the Gesture
20 area. The preprocessing step is done to each point that is inputted, in real-time. A check is done for the repetition of sequence. If it is determined as a Continuous Gesture 101 then the response to the input is in real-time. As the input data point comes in, action is performed and stopped at the end point. If the input is a normal
25 Gesture 102, then as every data point is inputted, feature extraction and recalculation steps are performed. Once the end point is obtained, reordering and matching steps are performed. It is highly appropriate to say that this algorithm processed data in real-time as the preprocessing, feature extraction steps are performed for every data point as and when they are inputted without any delay and does not wait for the end point to start the analysis. This improves the response time of the system greatly.

30 Most of the recognition techniques assume that the data is given in a predetermined form, which satisfy certain requirements. The low quality of the data is basically due to irregularity generated by users having an erratic handwriting and variations in handwriting styles. To overcome these problems, Fig 2, preprocessing
35 201 is used, which involves the substitution of data. Preprocessing eliminates noise, normalizes handwriting, and reduces the amount of redundant information, in order to fix the variations of handwriting and facilitate encoding of raw data into feature

vectors.

The first preprocessing step in the algorithm is De-Hooking 203. Hooks are commonly found at the ends of the strokes. They are generated during fast writing, when pen-down and pen-up events are generated. The way hooks are detected in strokes consists of locating abrupt changes of the turning angle. To eliminate the parts of a stroke which constitute a hook, two conditions must be met. The first one is that the turning angle satisfies $\theta_i > \theta$, where θ is a given threshold. The second condition is that the arc length is less than a predetermined arc length. According to Fig 9a, the angle between 3 consecutive points i , $i+1$ and $i+2$, is found to be θ_i and the length of the line segment between i and $i+1$ is also calculated. If the angle is greater than the threshold angle and also the line segment length is greater than the threshold line segment, then the point data in the hook area are discarded, i.e. the point i and points before i , if any, are discarded and a new starting point is obtained at the end of the hook. A similar de-hooking process is conducted at the end of the line, and a new end point is obtained if hooks are detected.

The second step of preprocessing 201 consists of smoothing 204 the line data. A weighted average of the present co-ordinate point with the neighboring points is calculated to get the new co-ordinate point, which is substituted in place of the original co-ordinate point. This method of smoothing is the simplest means of filtering, which is not as complex as the Gaussian filtering used by some other algorithms, which adds to the complexity of the recognition algorithm.

Fig 9b shows an example input line drawn on the touch screen. As seen in Fig 9b, the input line has hooks both at the starting point and end point, and is very noisy and erratic. At the end of the dehooking process, the hooks near the start and end points are removed as shown in Fig 9c. The output of the dehooking process is smoothened and the result obtained is shown in Fig 9d.

The smoothened data points are used for Feature Extraction 202, which is the next step of the algorithm. The step of Orientation determination 205, involves the estimation of slope between a pair of points. The slope estimated is used in the determination of the orientation of the pair of points from the orientation chart. The orientation chart divides vector space into eight cardinal directions. Fig 10 shows the Orientation charts. The ideal Orientation chart is showed in Fig 10a. To suit the kind of symbols dealt by this algorithm, the Orientation chart is optimized suitably as shown in Fig 10b, in connection with the present invention. Thus a sequence of orientation based on the successive points is obtained. If the orientation remains constant for a predetermined threshold length, the orientation code and the starting

point are stored as the feature point. The process is repeated until the end point of the line is obtained. Fig 11a shows the feature points (corner points) and extracted orientation codes for the input line shown in Fig 9b.

5 Once all the directions are extracted and the end point is obtained, the extracted directions are pre-processed again. A threshold length is calculated based on the actual length of the line. The length of each line segment, between the extracted directions is calculated. If this length is less than the threshold length, that particular extracted direction is discarded.

10 The obtained orientation codes are highly dependent on the direction in which the line was written. For the same line, different orientation codes are obtained based on the writing direction, which in turn will increase the number of entries in the database. To overcome this problem, the line data is made independent of the writing direction by normalizing the input line. The normalizing step consists of recalculating 206 the feature points and reordering 207 the feature points.

15 Recalculating of the feature points deals with the re-calculation of the co-ordinate points stored in the feature vector, in such a way that the distance between the pair of points is maintained a constant. The x and y coordinate points are changed according to the Pythagoras theorem, to maintain the distance between the points constant. The obtained co-ordinates are stored as the recalculated feature points. Fig 11b shows the original input line in Fig 9b after recalculation. The length of each line segment is maintained the same and the angles of the lines are drawn in accordance to Fig 10a.

20 The recalculated feature points are reordered, from a fixed reference point, cyclically, to obtain the normalized feature vector. The normalized feature points are used to calculate the orientation between them. The orientation codes thus obtained is independent of the writing direction. This orientation code forms a part of the feature vector. Fig 11c shows the original input line in Fig 9b after reordering. In this example, the top left-most point is taken as the reference point. The top left-most point 111, in Fig 11b is the starting point in the normalized feature vector and the points are reordered cyclically to obtain Fig 11c.

30 The shape of the line obtained after reordering is most of the time different from the original line as can be seen from Fig 11b and Fig 11c. In some cases however, the reordered line shape and the original shape is the same, as in Fig 15c and Fig 15d. In some other cases, the reordered shape is the same but the original line shape is different. An example of this is shown in Fig 15. Although Fig 15b and Fig 15d have the same orientations after reordering, their original shape is different as

seen in Fig 15a and Fig 15c. To differentiate such lines, the connection criteria step is performed. The presence of the line segment between a pair of points in the original feature points is determined. This connection information forms a part of the feature vector.

5 To differentiate between lines having the same orientation but differ in that there is an intersection between the line segments, line intersection check is done. The original feature vector is used for this. If an intersection is determined between two pair of points, the index of these points are found and stored in the feature vector. The orientations of the lines in Fig 12a and Fig 12b are the same. The only difference
10 is that there is an intersection. Thus the intersection criterion is used to differentiate them. Similarly, the case of the shape 8 in Fig 12c is different from the shape in Fig 12d only by the intersection, although both have the same orientation.

 To save memory, the orientation code and the connection information are combined into a single byte and stored in the feature vector. Thus the header of the
15 feature vector consists of the line index number, the packet size, number of orientation codes, number of intersections and whether the start point and the end point are the same. The rest of the feature vector consists of the orientation codes combined with the connection information and the intersection index, as shown in Fig 7.

20 In order to extract and match curves, curves are grouped into 8 categories as shown in Fig 17. Each curve is associated with a possible set of orientations, which make up the curve. An example of the possible orientations for curve A is shown in Fig 18. The next step of the algorithm is Line Segmentation. Thus, before the matching process is started, the extracted directions from the input lines are analyzed
25 and grouped into arcs or lines or both.

 The next step of the algorithm deals with the matching of the input line to the lines stored in the line database, Fig 3. The first input line's feature vector is compared to the reference feature vectors in the line database. First the orientation codes and the connection check is checked. If the conditions are satisfied, then the
30 intersection criterion for the line is checked. If this condition is also satisfied, then the Line Index number of the reference line in the database which matches with the current line is stored in the memory. If there is no match, the input feature vector is deleted from memory and the system waits for new input.

 The symbol database, Fig 8, consists of the symbol index number, line index
35 number of one or more lines and the presence of intersection between lines. Referring to Fig 8, as an example, the symbol with Symbol Index number 1 consists of lines

with Line Index numbers 5 and 4. The entry in the Intersection column is 1, which indicates that the intersection criteria between the lines need to be checked. So, if the two lines have Line Index number 5 and 4 and they intersect, there is a match and the symbol index number 1 is outputted. If the intersection criterion does not have to be checked, then a 0 is entered in the intersection column.

If the first line matches with one of the lines in the line database, Fig 4, the line index number thus obtained is used to create a list 401 of all symbols having this line as one of the lines. If the list has only one entry, the associated symbol index is outputted. Else, the system waits for the second or more lines, which are present in the list, to complete the matching process. For symbols as shown in fig 13, the intersection criteria needs to be satisfied, this will be specified in the symbol database.

In this invention, the recognition of special gestures 101, continuous gestures, does not use a database. The response to these gestures is real-time, even when the end point is not obtained. The first few data points are analyzed to determine whether it is a continuous gesture and the corresponding action is performed immediately, until the end point is obtained.

In this invention, the method of storing the feature vectors in the line database is similar to the recognition process. When in the database creation mode, the line drawn on the input device undergoes the pre-processing, feature extraction and matching steps, same as the recognition algorithm. The already existing line database is analyzed to detect the presence the same line, already in the database. If the line is already present in the database, it is not stored; else the line feature vector is stored as shown in Fig. 7.

For the creation of the Symbol Database, first the individual lines are matched and the already existing symbol database is analyzed to see whether the symbol made up of these lines are present in the database or not. If such a combination of lines is not in the symbol database, then an entry is made in the symbol database as shown in Fig. 8.

What is claimed is:

1. A gesture recognition method for an electronic system built with a processor, a memory, a screen and an input device for accepting coordinates of drawn lines, the method comprising:
 - 5 (a) accessing the input device to determine a start, end and line data points comprising coordinate position, timing and one or more start points;
 - (b) real-time de-hooking step of removing a hook from a beginning and end of the line data points;
 - (c) real-time smoothing step of removing irregular variations in a
10 handwriting style;
 - (d) real-time feature extraction step of finding an orientation between each pair of points;
 - (e) normalizing an original feature points of the line in the memory to obtain a normalized feature points;
 - 15 (f) checking a connection between each pair of the original feature point;
 - (g) checking an intersection between each pair of the original feature points;
 - (h) analyzing a curve and segmenting a line;
 - (i) matching step which comprises comparing a feature vector of a current line with a reference feature vector in a database; and
 - 20 (j) recognizing a symbol which comprises one or more lines.
2. The method according to claim 1, wherein the step (b) comprises:
locating abrupt changes in turning angles at the beginning and end of the line while the hook length is less than a given threshold, the threshold being a
25 percentage of the diagonal length of the screen.
3. The method according to claim 1, wherein the step (c) comprises:
substituting a coordinate of the original line data points with a weighted sum or an average of the neighboring line data points, to eliminate noise and erratic handwriting styles.
- 30 4. The method according to claim 1, wherein the step (d) comprises
slope estimation between each pair of points and mapping the slope to an orientation chart to obtain an orientation code of that segment of line between two points.
5. The method according to claim 1, wherein the step (d) further comprises:
35 finding a length between the pair of points; and
accumulating the length if the orientation code of successive points are same,

or storing the orientation code and the point as original feature point when the length is greater than the threshold.

6. The method according to claim 1, wherein the step (e) comprises:

5 restoring a problem of writing direction dependency by recalculating the original feature points and reordering the recalculated feature points to obtain the normalized feature points.

7. The method according to claim 6, wherein the recalculating comprises:

re-computing the original feature points, such that the distance between each pair of original feature point is constant, in accordance to Pythagoras theorem.

10 8. The method according to claim 6, wherein the reordering comprises:

cyclically re-arranging all the recalculated feature points from a fixed reference point; and

recalculating orientation codes with respect to the normalized feature points.

9. The method according to claim 1, wherein the step (f) comprises:

15 detecting a presence of a connected line segment between each pair of original feature points.

10. The method according to claims 6, wherein the reordering comprises:

cyclically re-arranging all the recalculated feature points from a fixed reference point; and

20 recalculating orientation codes with respect to the normalized feature points, and

wherein the step (f) comprises:

detecting a presence of a connected line segment between each pair of original feature points;

25 combining the orientation codes of the normalized feature points with the corresponding connection presence; and

storing the combined result in the feature vector.

11. The method according to claim 1, wherein the step (g) comprises:

30 determining the intersection point between two pairs of original feature points;

if intersection exists, obtaining the index of the original feature points and appending the index into the feature vector.

12. The method according to claim 1, wherein the step (h) comprises:

35 checking for a possibility of curves among the extracted orientations and segmenting line orientations into curves or lines or a combination of both.

13. The method according to claim 1, wherein the step (i) comprises:

matching the feature vector of the line with the reference feature vector stored in a line database;

if a match exists, outputting the line index number of the line.

14. The method according to claim 13, wherein the step (i) further comprises:

5 accumulating a list from a symbol database consisting of the present line index number.

15. The method according to claim 14, wherein the step (i) further comprises:

10 matching a second/or more line index number to the accumulated list and satisfying the intersection criteria between lines to ascertain the symbol index number.

16. The method according to claim 13, wherein the line database consists of line index number, the orientation code information, connection information and intersection information.

15 17. The method according to claim 14, wherein the symbol database consists of symbol index number, line index number and intersection criteria between lines.

18. The method according to claim 1, wherein the user interface on the screen is mapped into layout map that comprises:

regions list, input type and symbol list for each region.