Object Detection and Recognition in electrical Circuit

Master Thesis report

Submitted in partial fulfilment of the requirements for the degree of

Master of Technology

by

Saurabh Jain

Roll no. 143050048

under the guidance of

Prof. Abhiram Ranade

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai

Acknowledgement

I would like to express my greatest gratitude to **Prof. Abhiram Ranade**, for giving me aspiring guidance and friendly advice, for this Project work throughout the numerous consultation. I am also thankful to him, for his timely suggestions and constant motivation which helped me a lot in the completion of this phase of Project.

Saurabh Jain

Abstract

Sketches are used to express and represent ideas in various domains such as machine drawing in mechanical engineering, circuit drawing in electrical drawing, architecture and Software design. Increased use of touch Screen devices such as smart phones and tablets has made sketches as a very popular form of human-computer Interaction. In this report we will propose system using which electrical circuits sketches can be drawn easily with the mouse without using a menu driven interface.

The main goal of this project is to create a effective sketch recognition system for electrical circuits. The current system is restricted only electrical domain knowledge.

Our recognition system is build using DrawCAD as platform which is an IIT Bombay project for engineering drawing. It is different from menu based interface. It had a primitive identification module to recognize strokes from user as a point, line or circular arc.

The report thoroughly Describe the current situation of the DrawCAD

Contents

Chapter 1

Introduction

This chapter gives brief description about the problem definition of the project. In the first section a motivation for this project is discussed. In the second section we describes the problem definition. In the last section we will discussed the work done by others in this problem Definition.

1.1 Motivation

Sketches play an important role in human visual perception and are the primary way of communicating and representing information for humans. Increased use of touch Screen devices such as smart phones and tablets has made sketches as a very popular form of human-computer Interaction. Sketches form a basis for a variety of computer vision problems such as object recognition and detection. In this report we will look we will look at one such object recognition application in electrical circuit diagrams.

1.2 Problem definition

The basic problem description of project is to develop a powerful design tool that can draw draw the 2-Dimensional Diagram by the free-hand drawing which is more intuitive and natural way of drawing. Other Drawing tools with requires typical menu-driven, tool bar-driver or command-driven interface which is non intuitive and time consuming. The following three major aspect of DrawCAD :

- Sketch-primitive Recognition
- Constraint identification and its satisfaction
- Sketch-Symbol/Sketch-Object Recognition

1.2.1 Primitive Segment Recognition

The problem is formally stated as:

Given a sequence of mouse drawn strokes, where each stroke is a sequence of co-ordinate points with timing information(x,y,t), recognizing the constraints and the primitive segment intended by the stroke and replacing the stoke by precise form of recognized primitive segment. This problem also involves editing of strokes.

In other words, A raw stroke is beautified to the corresponding primitive segments. We now define the basic input-outputs of the System.A *Stroke* is being the array of sampled points between the a mouse press(pen down) and mouse release(pen-up) event. Thus the input is sequence of sample points $< x, y, t >$, where t is the time stamp of the sample point $< x, y >$. And we have three basic primitive segments as out, these are:

- points
- lines
- circular arc

when a stroke is drawn we need to find a closest fit of the stroke to these primitives. For each type of fit a confidence value is returned. Currently, this confidence value is just the negation of the error value, where error value is the sum of Least square of pixel position of raw segment.Suppose a stroke is identified as a line, we need to decide what are its end points and middle point with respect to the stroke drawn. Similarly, if the stroke is identified as a circular arc we need to decide its end points, center of circle of which arc is a part and radius of the arc. Other higher degree poly-lines such as spline can also be fitted but presently we are restricted to only the above defined primitives. This preprocessing also involves identification of the Markers i.e. the hard constraints which user might want to explicitly defined by drawing the marker. The type of marker such as equal length constraint marker or parallel line constraint marker is recognized during this process. This is discussed in more detail in next section.

1.2.2 Constraints Identification

Wide range of constraints are supported in DrawCAD that are commonly useful while drawing sketches. Constraints are categorized into two basic categories:

- *Soft constraints:* These are the constraints that are implicitly derived from the user drawing
- *Hard Constraints:*: These are the constraints that are explicitly specified by the notation we used in the high school geometry and required to detect in manner similar to the regular stokes in the drawing.

There is another classification independent constraints and relative constraints. Independent constraint such as angle constraint of line such as horizontal or vertical. Other type of constraints that are relative constraints describe the relation between different segments. Currently following are the relative constraint in our program:

1. Relative length
2. Parallelism
3. Perpendicular
4. Intersection
5. Tangency

6. Cocentricity

7. Points collinearity

To explicitly enforce the hard constraints user need to add some kind of marker so that our system can identify the relation between the segments, and then according to these feasible constraints are enforced in the diagram. Currently supported markers are are equal angle , equal length, parallel lines and perpendicular line markers .

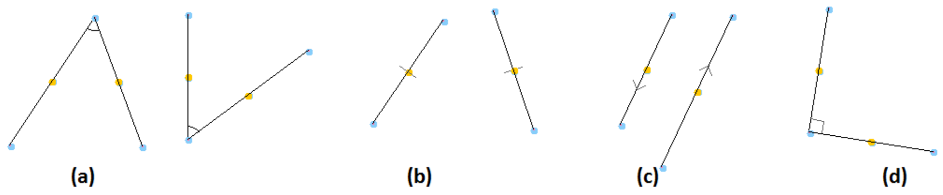


Figure 1.1: hard constraints

In fig 1.2.2 shows the various hard constraints, (a) shows equal angle constraint, (b) shows equal length constraint,(c) shows parallel line constraint (d) shows perpendicular constraints

After recognizing various constraints constrain solving is done using technique in [Shi05],which uses *Newton's Method in Multiple Dimension.*

1.2.3 Sketch-Symbol/Sketch-Object Recognition

After we have drawn a diagram suppose we want to identify different objects that have semantic meaning. This identification of the object can be in context to domain such as flow chart diagrams, electrical circuit domains, mechanical engineering drawing etc. Currently We are using Electrical Circuits as our Domain knowledge for these object or symbol identification.

The goal of this aspect is to Develop a efficient Sketch Recognition system which can identify the symbols or electrical circuits components.

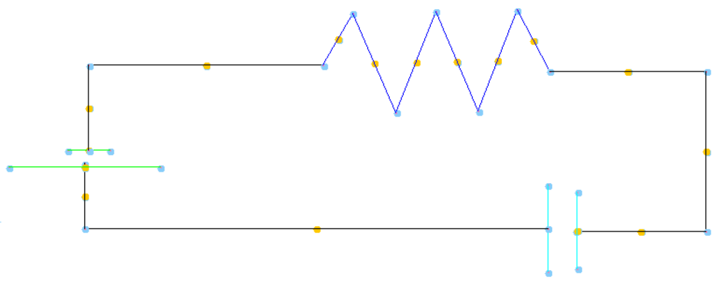


Figure 1.2: Symbol Recognition in the Electric Circuit

In figure 1.2.3 after having a drawing of the electrical circuit we want computer to recognize the various symbols in the electrical circuit drawing such as resistor, battery and capacitor in the given Diagram.

1.3 Related Work

To be updated.

1.3.1 LADDER

1.3.2 SktechREAD

1.3.3 AC-SPARC

1.3.4 Structural Method

1.3.5 Zernike Method

Chapter 2

Description of Architecture of DRAWCAD

In this Chapter we will describe the basic architecture of the current DRAWCAD version

2.1 Drawing View

2.2 Constraints View

2.3 help view

Chapter 3

Previous Work

3.1 Segmentation

It is the process of breaking sketch strokes in to the simplest building block primitive segments.These primitive segments are points, line and circular arc.

Let us consider an example we want to draw a resistance as show in Fig.3.1(c). Suppose instead of drawing 8 primitive lines of resistor by one stroke at a time, user draws a continuous stroke as shown in fig 3.1(a). After drawing this stroke can be broken down in to 8 primitive strokes to form a resistor. We need a mechanism to identify the these primitive segments i.e. where the large continuous stroke to be broken . The point at which stroke is broken are called segmentation points, these points are represented by red points in fig 3.1(b)

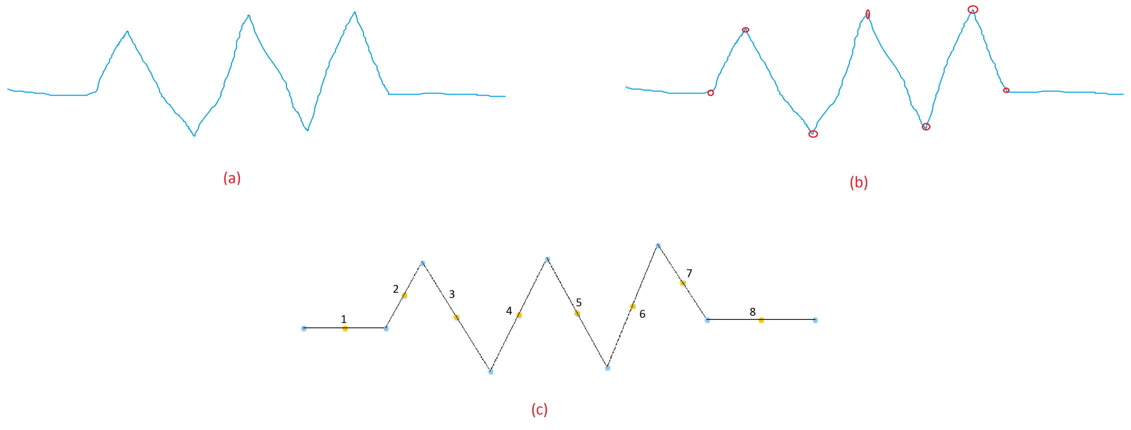


Figure 3.1: segmentation example

3.2 Segment Point Detecting Scheme

The segmentation process must have following features;

- It should ignore unwanted pen fluctuation and useful information should be kept intact.
- Able to ignore noise at the end and starting of the stroke.
- A real-time solution.
- Able to adjust with the speed of drawing of different users.

Different properties used to find segment points are described below:

3.2.1 Direction

The value of direction at any point (x_n, y_n) is given by:

$$d_n = arctan\frac{y_{n+1} - y_n}{x_{n+1} - x_n} \tag{3.1}$$

Basically this equation the user change its direction while drawing the strokes. As this information is noise sensitive as the only neighbouring 2 points are considered. We can change this method by finding the direction between end points of some neighbourhood of fixed window. However the sharp change in the direction will not be captured by this approach.

3.2.2 Curvature

Curvature can be defined as extended feature of direction. It is given by:

$$c_n = \frac{\sum_{i=n-k}^{n+k-1} |(d_{i+1} - d_i|}{D(n-k, n+k)} \tag{3.2}$$

where k = constant neighbourhood window $D(p, q)$ = length of path between the points p and q, which is summation of distance of all the points between them d_i = direction value at point i Curvature measures the change in the neighbourhood. Hence, curvature value is useful in identification of points having sharp direction change.

3.2.3 Speed

The speed value at any point in stroke is defined as:

$$s_n = \frac{D(n-1, n+1)}{t_{n+1} - t_n} \tag{3.3}$$

where t_i = time stamp of ith point,. $D(p, q)$ = path length between p and q. The motivation here is that most of the user slows down at segmentation points. So local minima of speed can be used to locate the segment points.

3.2.4 Corner

segmentation points between lines are called corner points. Supposed we have line as our only primitives the corners can be found as:

- start a point loop through the next point.
- calculate the distance and path length between the end points.
- if ratio of path length and distance becomes less than a threshold mark it as segmentation point. This can be explained by

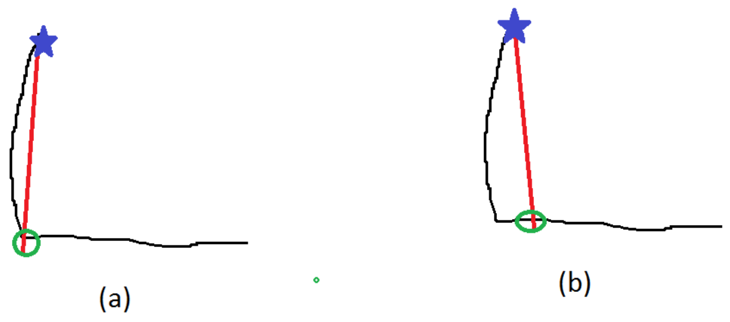


Figure 3.2: Corner points

In the figure3.1(a) the ratio of distance to path length from blue point(starting) to red point will be larger, but in the 3.2.4(b),this ratio will be less than the threshold , this this point will be considered as the corner point , and the primitive line is identified between starting and corner point. Now process is repeated on the rest of the stroke with corner point as new starting point.

3.2.5 Segmentation Method Proposed

Following two step Process for segmentation is used :

1. Finding initial segments:
2. Merging segments.

Finding initial Segments

This step find the set of segment points found by both speed and curvature method, which is actually a superset of actual segment points. As some more points can be identified as segment points because of noise. If at a certain point, if speed is less than speed mean*0.4 or curvature is greater than mean *1.5, then it is classified as initial segment point. Also the points close to the end point of the stroke is ignored because there can be possibility of noise at the end points.

Merging Segments

In the first step many other points were also identified as segments points. In this process the decision of whether a point to be taken as segment point or not is taken by merging the two segment having that common segment point. In this process two consecutive identified segment is taken if the two segment has angle less than a threshold(say 20 degrees) then two segment are merged.

3.3 Circuit-Symbol/Circuit-Components/Object Recognition

The Sketch-symbol recognition process for electrical circuits divide into two phases:

- Detection of the Candidate Symbol/Object/Component
- Classification of Detected Symbol/Object/Component

3.3.1 Detection of the Candidate Symbol/Object/Component

Before starting the process of Recognition process we need to identify the Candidate Symbols in the Diagram. A *Candidate Symbol* is a set of primitives segment that can represent an actual object from our domain. In this phase all the candidate symbol is identified which further will be used in next phase of recognition process.

An*Object/Symbol* is a Candidate Symbol which definitely represent some element from the our domain. In our Domain context these can be resistor,capacitor, gates etc. In other words it is a set of segments with semantics.

This phase take a series of the primitive segments as input and gives the list of candidates. Each candidate is a set of consecutively drawn primitive segment. Important thing to notice is that, we are considering that user will always draws a complete object at a time i.e. once user start drawing a object should finish it and then goes to next object or symbol in the diagram. In the figure 3.3.1 the input is

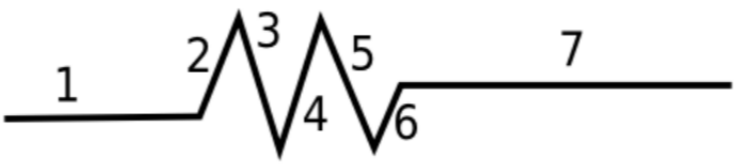


Figure 3.3: Corner points

set of segments 1,2 3,4,5,6,7. Then supposed identified Candidate symbols are 2,3,4 , 3,4,5,6,7 and 2,3,4,5,6. but 2,3,4,5,6 is our actual symbol for resistor so this set is our Object which is identified in the next phase of Recognition process.

As there will be large number of points gathered in place where there exist is a Symbol/object, using this basic idea we can observe that primitives that are close with constitute a real object. For quantification of the concentration of point a notion of *Ink Density* is defined which is as:

$$InkDensity = \frac{InkLength}{BoundingBoxArea} \tag{3.4}$$

where Ink Length represent the summation of length of all the segments and Bounding box is defined as a smallest rectangle which can cover all segment and whose sides are parallel to axes. To find the Candidate symbol two recursive procedures are used. First procedure tries to find the end point of the candidate symbol. Initially , a bound box is create d for the first two segment and ink density is calculated. Then at each step next segment is added to the and again the ink density is calculated. If the ink density drops more than 5% it will not be included in the current candidate symbol and previously added segment will be marked as possible end point of the object. Algorithm is given below.

```
input  : Series of primitive segments $segs[0 : n]$
output: List of candidate symbols
$RESULT = \{\}$
Initialize Bounding Box $B$ with $segs[0]$ and $segs[1]$
for $i \leftarrow 2$ to $n$ do
    if $i \geq 6$ then
        break
    end
    Add $segs[i]$ to $B$
    if Density drops more than 5% then
        $L1 = \text{Find\_Front}(segs[0 : i - 1])$
        $L2 = \text{Find\_End}(segs[i : n])$
        $RESULT = RESULT \cup L1 \cup L2$
    end
end
$L1 = \text{Find\_Front}(segs[0 : i - 1])$
$L2 = \text{Find\_End}(segs[i : n])$
$RESULT = RESULT \cup L1 \cup L2$
return $RESULT$
```

Figure 3.4: Forward Algorithm to detect end point of candidate symbol

In the second procedure, start point for the candidate symbol is calculated from the identified end point from the first procedure. In this procedure last two drawn segment is considered, and then bounding box and ink density is calculate. Then it will move backward and add previously drawn segment in the box and new ink density is calculated . Whenever ink density drops more than 5% it will be marked as start point. the procedure is given as:

3.3.2 Classification of Detected Symbol/Object/Component Terminologies

- **Class or Symbol Type** : A class represent any one type of electrical symbol or

```
input  : Series of primitive segments $segs[0:n]$
output: List of candidate symbols
$RESULT = \{\}$
if $n \leq 1$ then
    return $RESULT$
end
Initialize Bounding Box $B$ with $segs[n]$ and $segs[n-1]$
for $i \leftarrow n-2$ to $0$ do
    if $n - i + 1 > 6$ then
        break
    end
    Add $segs[i]$ to $B$
    if Density drops more than 5% then
        $RESULT = RESULT \cup \{ segs[i+1:n] \}$
    end
end
$RESULT = RESULT \cup \{ segs[i+1:n] \}$
return $RESULT$
```

Figure 3.5: Backward Algorithm to detect end point of candidate symbol

elements. So an Candidate Object can belong to any the defined classes such as resistor , capacitor etc in our Domain context.

- Connector: Connectors of an objects are segment which are used in sketch to connect the other object or symbols.

After the Candidate symbols Detected, in this phase we need to classify these symbol in to various Symbol Classes or connectors from the sketch . We have a list of candidate symbol as input and our output will be a list of actual symbols in the sketch.

To Classify objects in various classes we need features of objects which are uniquely defines the class of the object. Common intuitive geometry features of a sketch of the object are used these are the major feature used.

1. total number of segments, number of line segment, number of arc segments
2. Number of parallel lines, number of perpendicular lines
3. Number of end point to end point intersection segments.
4. Number of end point to mid point intersection segments.
5. Number of mid point to mid points intersection segments.
6. Ratio of average length of the segment to largest length of the segments.

We will keep some objects as our reference object and stores each feature values them in a file along a class label. When an unknown candidate object comes for classification we will calculate the above mention feature of this unknown candidate and find which reference symbol it is matched most. We have Assumed that each of these features has a normal distribution. Thus for each class we will have a mean and variance from the references .

To find the matching probability of the unknown object normal distance for each feature to every feature in a class is calculated, The normal distance for a feature f of class lets say C from the unknown object is calculated as:

$$Normal\ Distance_f = \frac{1}{\sqrt{2\pi\sigma^2_{f,C}}} exp(-\frac{(x-\mu_{f,C})^2}{\sigma^2_{f,C}}) \tag{3.5}$$

where $\mu_{f,C}$ is the mean of feature f of class C
and $\sigma^2_{f,C}$ is the variance of feature $f$of class C

After calculating the normal distance of all features of an object for a class the *Matching Value* is calculated as :

$$MatchingValue = \prod_{i=1}^{n_f}(Normal\ distance\ for\ ith\ feature) \tag{3.6}$$

where n_f is the number of total features. from above equation Matching value of every class for a candidate object is calculated and we will assign the class label to unknown object. the Algorithms defined was:

```
input  : List of candidate symbols $LST$
output: List of objects with class label

for each entry $X$ in $LST$ do
    calculate the $Matching\ Value$ for $X$ against all classes in the domain ;
    Find class $C$ for which highest $Matching\ Value$ found ;
    assign class label $C$ to $X$;
    set $Matching\ Value$ of class $C$ as parameter $MV$ for $X$ ;
    if $MV$ for $X$ < threshold then
        remove $X$ from $LST$ ;
    end
end
sort $LST$ in descending order of $MV$ ;
for each entry $X$ in $LST$ from start to end do
    $L1$ = List of segments present in $X$ ;
    $L2$ = List of segments connecting to $X$ ;
    remove all candidates other then $X$ from $LST$ which has at least one segment
    from $L1$ or $L2$ ;
end
return $LST$ ;
```

Figure 3.6: classification Algorithm

Chapter 4

Current Status of DrawCAD

4.1 segmentation Mode

In the current implementation of DrawCAD we have two main modes in which user can draw the sketches thes are :

1. One stroke per segment (OSS)
2. Multiple stroke per segment (MSS)

These modes are indicated in the toolbar panel by default the OSS mode is on(indicated in cyan color). And the MSS mode can be selected by clicking on OSS mode which toggles the mode to MSS(indicated in green color). more strokes.

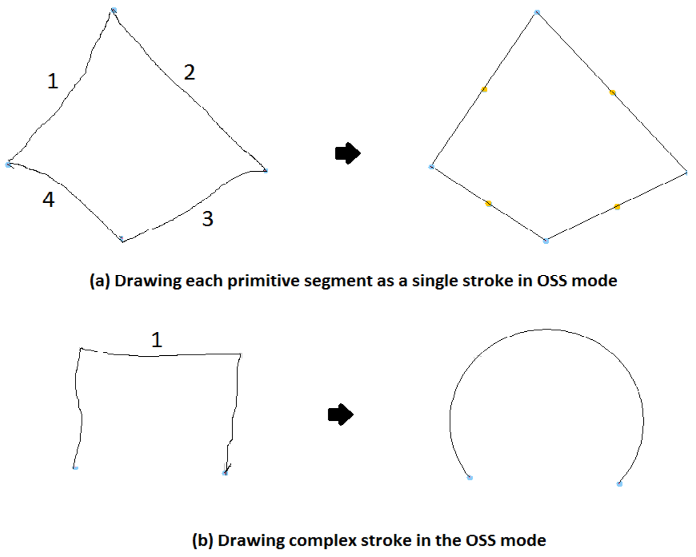

1
2
3
4
(a) Drawing each primitive segment as a single stroke in OSS mode
1
(b) Drawing complex stroke in the OSS mode

Figure 4.1: Result of Drawing various strokes in OSS mode

4.2 One stroke per segment(OSS)

In this Modes in order to draw an object, the primitive segments of the objects such as line, arcs , and circle must be drawn as single stroke per at time. i.e. single stroke must not be a complex stroke(combination of two or If the user tries to draw a complex stroke in this mode then the primitive segments fit to the stroke will be a single segment having average fit to the whole stroke.

As show in fig 4.1(a) when user tries to draw a quadrilateral by drawing stroke 1,2,3,4 one at a time in OSS mode. then quadrilateral is correctly fitted.In the fig 4.1 (b) when user tries to draw the figure using the complex stroke i.e. whole figure as single complex stroke then a arc is fitted as a primitive segment to this stroke.

4.3 Multiple stroke per segment(MSS)

In this mode complex figure is drawn using a single strokes only or in other words two or more simple strokes are drawn as a single stroke without stopping. In this strokes drawn undergo a process of segmentation where the complex stroke is broken down into simple primitive segment. This increase the ease of access for the user, as user can draw many segment in one stroke only. In the figure 4.3 when a complex stroke(left) is drawn in the

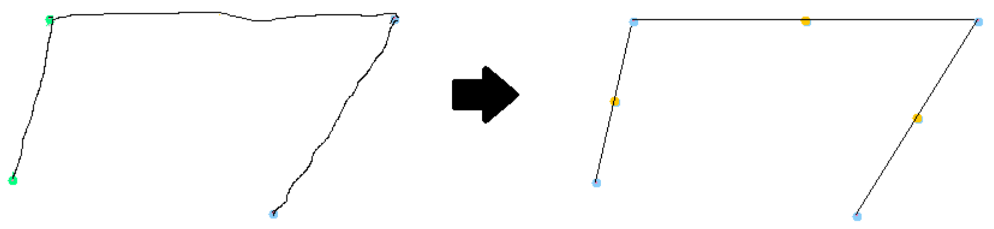


Figure 4.2: Result of Drawing various strokes in MSS mode

MSS mode then the this stroke is segmented in to the primitive segments as shown in right in figure.

4.4 Segmentation bugs

Following are the bugs in the current DrawCAD code in the MSS mode.

- hard constraints by adding marker is not recognized in this mode.
- when the complex stroke is very small or user pauses for small time at the segmentation points than sometimes the segmentation points in not detected correctly
- after segmentation process the cursor should remain at the ending point of the complex stroke but currently the cursor after completing the stroke remain at the first identified segment point.

4.5 Test Cases for Sketch-Object recognition

Previously There was no defined data set for our Program we have created about simple 120-150 test cases and 15-20 complex test cases. Simple Test cases contains only single type of symbol with no connection between them. Complex Test cases will be like complex electrical circuits having average 6-8 diagrams. to test the Recognition efficiency of the system. In some of the test cases a small amount of noisy segments(line segment instead of arc or vice-versa) is introduced . Also in some of the test-cases order of drawing of stroke is changed, These test cases will going to be failed for our cases, as we had taken the order of drawing in to account during the detection process. In some of test case symbols are aligned at different angles i.e. horizontal, vertical, diagonal to test whether our system is orientation invariants We are currently using the 11 classes of electrical components these are:

1. Resistance
2. capacitance
3. inductor
4. battery
5. buffer
6. NOT gate
7. OR gate
8. AND gate
9. NPN transistor
10. XOR gates
11. Box

following are the test-cases for sketch object recognition.

Chapter 5

Work Done/To be done

This chapter gives detailed description of what is the progress made in this semester, What are thing that were analysed, What were the important changes made in the various modules.

In the first section of this chapter, we will discuss about tolerating some noise to the sketch-symbol recognition process. In the second section, we will discuss about the modified approach to find the Detection algorithm of the candidate symbol. In the second section, a modified Recognition algorithm is proposed. In the fourth section , discuss about the some functionalities added in the DrawCAD software. In fifth section we will see if continuous recognition of symbol while drawing the circuit diagram can be useful.In sixth section we discuss about some of the segmentation bug fixes done.

5.1 Tolerating Noise in Symbol-Recognitions

In the step to of the Circuit-Object recognition process there is need to tolerate some amount of noise to the object recognition process .The motivation here the user is likely to make mistake if the user make a small mistake while drawing object, and this mistake does not change the overall semantics of the object then this small mistake can be tolerated in Object Recognition phased.

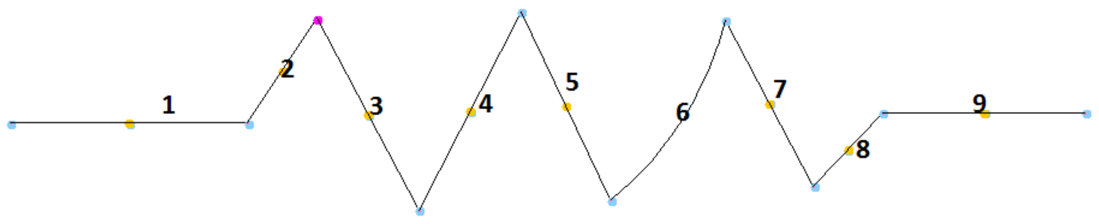
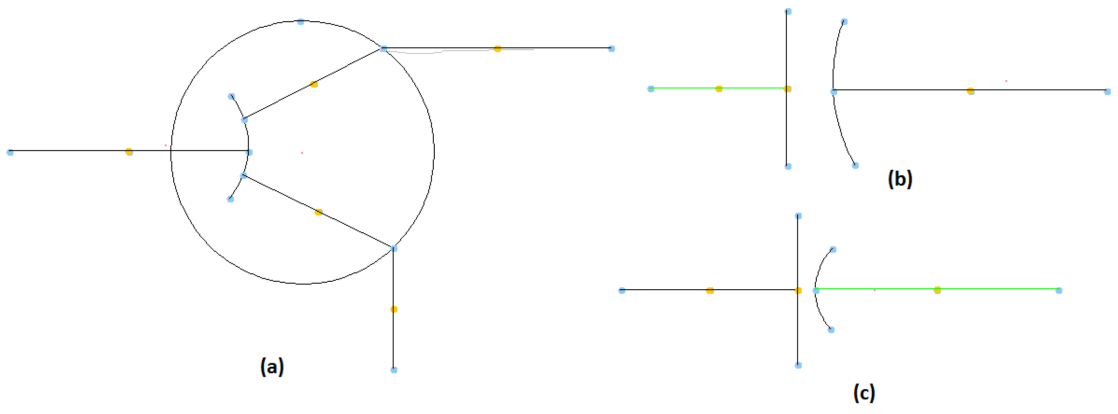


Figure 5.1: Noise while drawing a resistor

For example in fig 5.1 we can see in the process of drawing if the user made a mistake in drawing the 6th segment as a curve instead of drawing a line segment, this type of case should be capture, and our recognition algorithm should recognition it as resistance. Also in fig 5.1 some amount of noise can be tolerated, as in (a) while drawing npn transistor as a small curve line connecting base emitter and collector instead of a straight line segment does not change the semantic meaning of the symbol much, thus it can be ignored. Similarly, in while drawing capacitor(b) and battery(c) , a curve line instead of straight line can be ignored.



Also some amount of variation in length of the particular symbol is allowed , For example a resistance can have more number of crests and troughs. Similarly, and inductor can have more number of circular arcs.

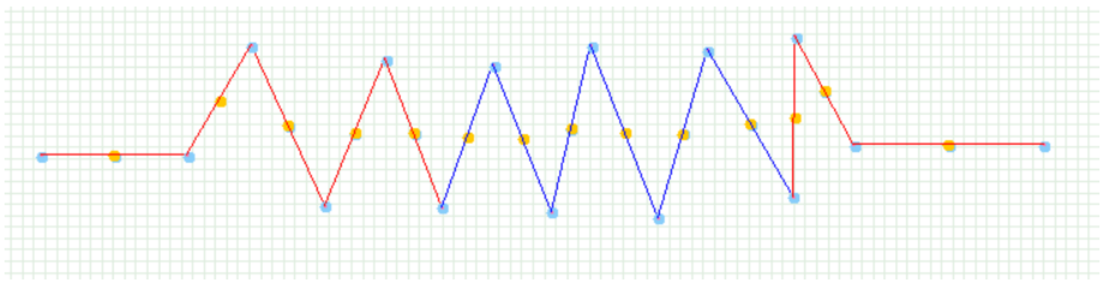


Figure 5.2: resistor can have any number of crest and troughs.

5.2 Modified Detection algorithm

To be updated.

Modification in the Detection algorithm:

- varying thresholds for the ink density
- use different sorting order instead of time based sorting left to right ordering or top to bottom ordering

5.3 Modified Recognition algorithm

To be updated.

Modification in the Detection algorithm:

- varying threshold for the probability of Matching values of Candidate object
- including more features in features set of the symbols such as :
  1. number of line that are approximate horizontal or vertical, or diagonal.
  2. number line segment or arc segment less than the average length,

   3. number of almost parallel segments or perpendicular.
   4. number of concentric arc segments
   5. number of arcs which are approximately semi arc, full circles or quarter arcs .

- use different sorting order instead of time based sorting left to right ordering or top to bottom ordering
- using bounding box on other axes instead of principle axes only, for example axes making angle 45° with principle axes.
- trying other method instead of matching value multiplication. for example Least squares or weighted summation of matching values.
- when a reference symbol is stored we can store the multiple copies of same symbol in different orientation lets say we rotate the object from 0 to 360 in bins of 30°. This will make our recognition process rotation invariant.
- When a reference symbol is added we can store the multiple copies of the same reference in which small amount of noise introduced i.e. say one or two strokes being changed from line segment to arc segment and vice versa.

5.4 Functionalities for an Object/Symbol

Following are the functionalities that has been added to the DrawCAD in this Version.

- Colour Codes for different Detected Symbols
- Select a whole Symbol/Object
- Delete a selected Symbol.
- Drag a whole Object to new location.
- increase or decrease the size of object as a whole

5.5 Immediate Recognition

This is also like user mode of recognition process, In this mode there is continuous recognition of the objects as soon as the user add a new stroke to the sketch or make some changes in the sketch like editing, scaling , translating deleting a segments. This mode can be shown in the toolbar as an indicator to which mode is on.

This type of recognition can be helpful in making user an accurate Objects as shown as user add a strokes to the current Sketch if the new stroke was not a part of an object by our algorithm then user can realize he/she had made some made some mistake while drawing this stroke our previous few stroke. So with the help of this mode user can correct the sketch on the fly easily as only previous few strokes to be changed instead of correcting the mistake after the user had drawn a whole diagram.

5.6 fixing segmentation bugs

To be updated.

Chapter 6

FUTURE WORK

6.1 Global Beautification

The future work can be aimed at a global beautification of the whole drawing. For example all the wires in the circuit are axes aligned, all the components of same size orders and aligned. Suppose user draws a figure 6.1(a) and want the figure to be globally beautified

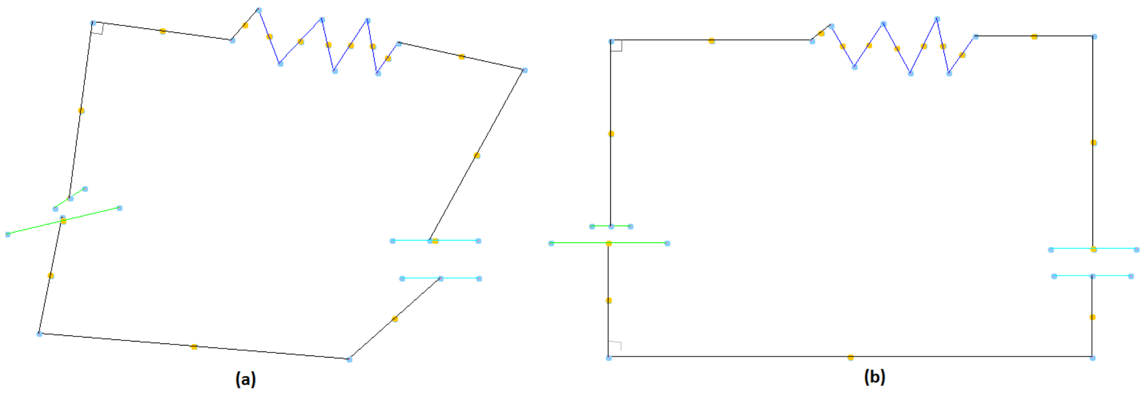


Figure 6.1: Global beautification.

where the connector are aligned to axes even the symbol are also aligned . One More layer of beautification can after recognition of symbol properly user drawn symbols can be replaced by the a standard defined symbol. hence all the symbol of same class will look same to a standard symbol.

6.2 Symbol predictor

Suppose Immediate recognition mode is on while drawing a symbol in the Diagram and if our system identify or predict what our user is trying to draw we can give an auto complete option to the user which can be more interactive for the user.

Suppose user is try to draw a resistor as fig 6.2 (a) , after few strokes suppose our system detects it is a resistor and gives an auto complete option to the user . or can give suggestion which symbol to be autocomplete capacitor or battery in (b).

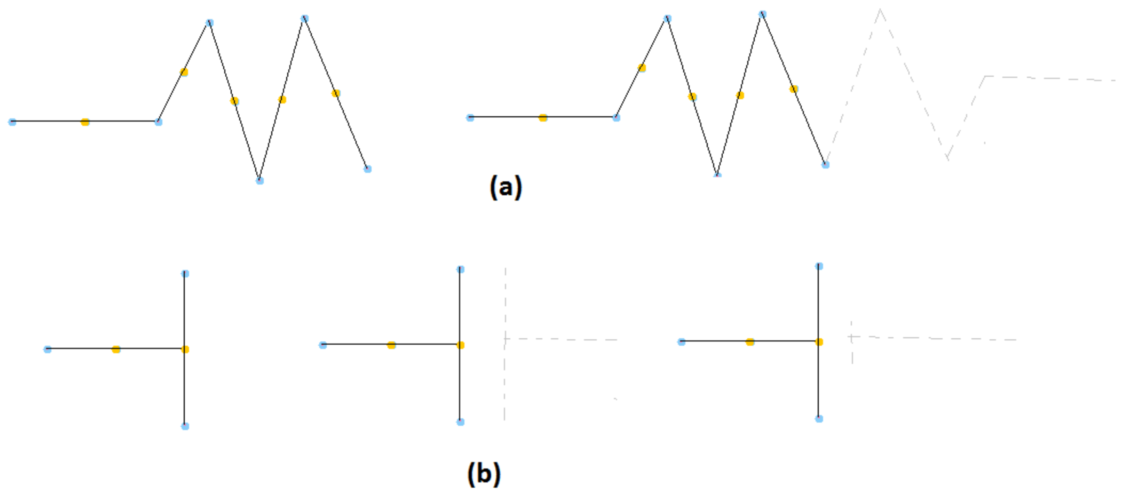


Figure 6.2: Symbol prediction

6.3 Domain Independence

Aim is to build a Domain Independent Object Recognition system. which is a generalized multi-domain sketch-object recognition system to draw class diagrams , data flow diagrams, flow charts, graphs, network configuration , stick-man figures, UML diagrams, state machines etc. The challenge will be here how to detect the candidate symbol, what should be our feature set in the process of Recognition.

6.4 Patterned Object prediction

suppose a specific In architectural design Drawing user generally draws some patterned objects. The aim is to build a system such that it can identify the pattern based on the geometrical properties of the object currently drawn and predict the next few segment which can be helpful in an interactive manner for the user.

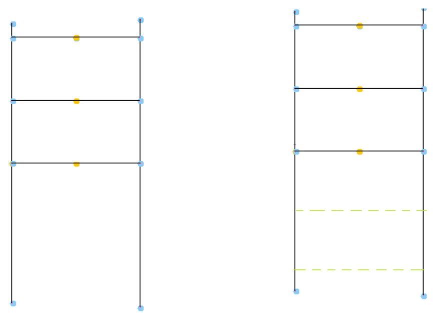


Figure 6.3: patterned object prediction

As in fig 6.4 users is trying to draw a ladder after drawing few strokes it the system recognizes what the is user drawing on the basis of geometrical properties and gives the prediction result as dotted line so that user can either choose to auto complete or can finish the object on its own.

6.5 Solver to the circuit Diagram

Suppose a Circuit diagram where the symbols are identified and there values (for exampl 40 ohms) are given. The aim is to attach a solver that can process all the information of the circuit diagram and able to calculate the current flow in the various arc of the circuits or potential drop across the resistor or capacitor.

Bibliography