MAHI: Machine And Human Interface

Bhupesh Kumar Singh kumar.bhupesh04@gmail.com
Department of Computer Science & Engineering
Lingaya's University,
Faridabad: 121002, India

G. Sahoo drgsahoo@yahoo.com
Department of Computer Science & Engineering
Birla Institute of Technology, Mesra,
Ranchi: 835 215, India

B.L.Raina rbushan@rediffmail.com
Department of Information & Technology
Lingaya's University,
Faridabad: 121002, India

ABSTRACT

Sketch recognition systems used for the development of many domains are time consuming. This is because it involves intricacies in handling the data with greater care for each domain. Present authors introduced *A new approach to sketch recognition using Heuristic* [1]. It compares very well with the statistical approach used in Bayesian networks[4] In this paper, we have introduced MAHI: *(Machine and Human Interface)* as a sketching language based on geometrical rules associated to the recognition of basic elements in its domain as the better option discussed in MAHII (Machine And Interactive Interface [2]. The recognition engine interprets the sketches based on the information obtained from the description of domains associated with the input from the user.

Keywords: MAHI, MAHII, LADDER, Domain-Description, Geometrical Constraints, Multi-connected lines.

1. INTRODUCTION

Sketch recognition is an important part of sketch-based system and it needs domain information to classify and recognize the shapes. Over the years, sketch recognition systems on pen-based input devices and hand-drawn diagrammatic domains are discuses in details. Stiny and Gips,[5] have given the grammars for shape description language. Jacobi et. al. [6] gave software module that lacked improvement to code the domain-dependent recognition system. Bimber et. al [7] introduced BNF grammar with the restriction to only shape information and provided no other helpful information. Mahoney [8], Caetano et. al. [9] use languages (including Fuzzy related grammar) were again subject to some limitation.

Recently T.Hammond & R.Davis [10][11][12][14][16] have discussed LADDER as a first sketching language. The author's claim that the language was designed and developed successfully while describe its domain classes. Unfortunately the application of language is subject to a few limitations of describing shapes such as a fixed graphical grammar, shapes that have a lot of regularity and not too much of the details. The language can describe domains for a few curves and there is difficulty in specifying curves control points.

To reduce the limitations of language a new approach to sketch recognition has been proposed in [1].Recently we have introduced its interface MAHII in [2]. In this paper, we describe a language that provides the description of basic shapes of sketch in terms of its geometry, defining the domains that implicitly inherit the geometrical based shapes. Since the dependence of domain sketch recognition system is the main disadvantage for all the systems, therefore to provide a natural and more attractive interface [2], MAHI places minimum constraints on the users in view of the system being independent of the domain.

MAHI is the language describing the shapes drawn, displayed, and edited subject to the domain. It has been successfully tested to multi-domain recognition system along with a code generator that parses MAHII interface based on domain description and generates the codes pertaining to MATLAB to give complete recognition to the system.

MAHI identifies a few basic shapes through which recognition engine can identify its corresponding domains and then it connects the particular domain to itself through an interfacing. This way, the system provides the developer to define the domains independently which describe what the domains shapes look like, and how they should be edited and displayed after they are recognized. The recognition engine is followed by heuristic engine that finally recognizes the input sketch completely.

We discuss components of MAHI (language), contents of the sketch grammar in MAHI, Testing, and System- Implementation alongwith conclusion in the subsequent sections.

2. MAHI LANGUAGE

MAHI domain descriptions include defined shapes and other information related to the recognition process, such as stroke order and stroke gradient. It contains the defined geometrical shapes with minimum constraints and takes into account its editing behavior, display methods and syntax for specifying a domain description. We create a domain description reusing defined shapes to build the desired shape hierarchically.

2.1 ELEMENTS OF MAHI

In MAHI (figure 1) we define the geometric elements in terms of the entities given: The basic core entity (BCE) unit defines a *point,* a collection of BCE forms a core entity (CE) defined by a line or an arc and the collection of BCE and CE forms a derived entities (DE) as open or closed shaped.

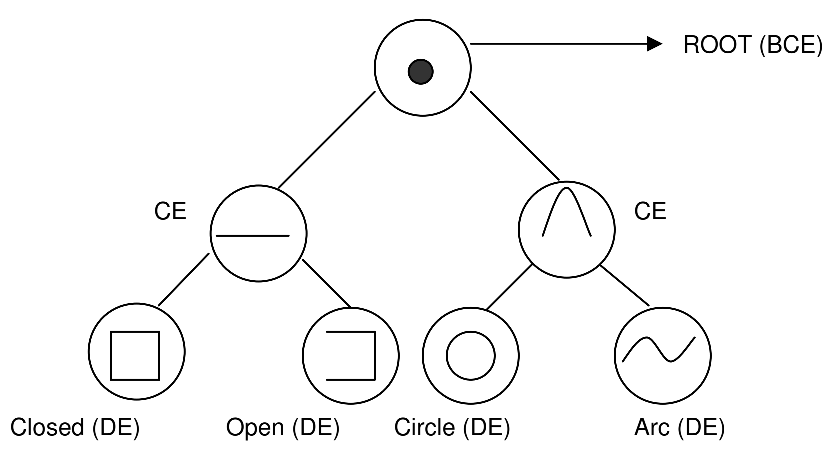


FIGURE 1: Basic Elements

2.2 GEOMETRICAL ANALYSIS

In MAHI, we are able to draw any of the shapes based on entity BCE, CE, and DE as defined by *point, arc, line, rectangle, square, circle, ellipse, polygon, curves, surfaces, volume etc*. Here a shape is a geometrical structure with basic input as geometrical rules associated to recognition process such as the stroke. Using the shape properties, an arbitrary shape is extended or reduced to the desired shape. An application designer can redefine the properties as and when he needs. The language has proven to be very powerful and highly interactive for multi domains. The language enables more accurate and fast sketch recognition by using bottom-up as well as top-down recognition in view of mathematical definition of line being defined as locus of point and point being defined as limiting case of a line. Thus, a point is recognized and is used to identify and be identified as a line or arc since the line or arc is extended to form any other extended shape(s) or extended shape(s) merged to a line or curve.

2.3 SHAPE-GEOMETRY

Generally, we define a point as a precise location or place on a plane, usually represented by a dot. We remark here that geometrically it is appropriate to define a point in terms of line segment as such; we first define a line segment as given by the following definition:
We define a line segment as the shortest distance between any two given points, whereas a point is a line segment of join of two points whose distance in one dimensions or gradient in two dimensions tend to zero. Further, if the distance between two consecutive points is not minimum then it will define an arc and consequently piece-wise continuous arcs will give rise to a curve.

- Shapes in 2 & 3 Dimensions

  Shape of point in 2 Dimensions: Given two points $P(x_1,y_1)$ & $Q(x_2,y_2)$ we associate metric d joining P & Q such that $d(\overrightarrow{PQ})\to 0$, giving $x_1\to x_2, y_1\to y_2$. i.e. $P(x_1,y_1)\to Q(x_2,y_2)$ implies that the limiting case of a line defines a point. Similarly $P(x_1,y_1,z_1,)\to Q(x_2,y_2,z_2)$ gives a point in three dimensions.

- Arc in 2 & 3 Dimensions

  If A & B are any two points in a plane with regard to point O as the origin, then as per mathematical definition we define length of Arc AB given by: $\widehat{AB} = |OA| \quad \angle\theta$ corresponding to the equation of the curve r=f(θ) where r= $|OA|$ & $\angle\theta = \angle AOB$ .Same results hold true in 3-dimensional space.

Using entities defined by BCE, CE or DE given by figure 1 we obtain any shape (linear or curve linear). Further, these entities can give rise to any of the shapes in different domains such as states in finite state machine, formation of pulley in mechanical engineering, circuit design in electronic engineering, etc.

2.4 SHAPES WITH CONSTRAINTS

A number of geometrical shapes can be defined by the algebraic equations subject to the given constraints. For example ax+by+c=0 subject to the condition $d \le x \le e$ represents a number of lines including family of parallel, perpendicular lines etc. in a plane of two dimensions for different values of a, b, c, d and e. In other words if the sketch grammar consisting of algebraic equation subject to the constraints as given above is represented geometrically, then we find that it is far superior to express constraints in terms of mathematical language than otherwise given by traditional constraints such as rotatable, angle, horizontal, vertical, etc. as defined in [9][15] .

The authors [10][11][15] presented a language containing a library of pre-defined shapes with pre-defined constraints such as *horizontal, posSlope, above left* etc including *IsRotable, angle L, vertical constraints* to define the orientation subject to the relative co-ordinate system, which seems cumbersome to the user. However, MAHI ignores all these complicated expression and having inbuilt mathematical system, the shape takes the form with ease than in Ladder. In MAHI gradient for any line in two dimensions is given by m_i= -(coefficient(x))/(Coefficient(y)), i=1,2 as such a user can draw any line following the language which naturally is one of these: Perpendicular, parallel, collinear, coincident, same side, opposite side meet, intersect, tangent, centre below, centre above, positive slope, negative slope, etc. The components and properties of these lines can be used hierarchically bottom-up or top-down in the shape descriptions. The other related new shapes are drawn by extending the properties of a line, rectangle, parallelogram, circle, to any of the shapes. Thus, in general, a new shape defined by the components, geometrical constraints, alasis, etc. as given in figure 2:

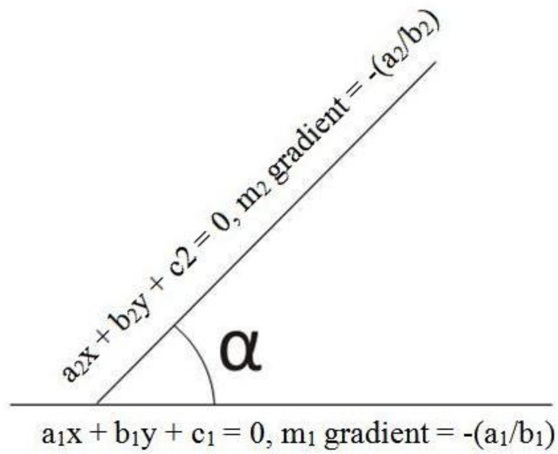


FIGURE 2: Geometrical Constraints

2.5 DEFINED-EDITING BEHAVIOR

MAHI knows when and how to recognize, edit and display the shapes. Editing and displaying are the important components of sketch interface that vary as per different domains. The display of the object helps the sketcher to make the recognition not only possible-shape but makes it beautiful by removing the clutters followed by editing gestures consisting of event for each shape. The advantage in MAHI is that a user is encouraged to standardize different domains by including some defined editing behavior. In fact, one can define one's own editing behaviors for each domain. *For example: Using the same gesture such as drawing a point inside a circle may be intended as a check in one domain(check box) and a full stop in the other domain (text box) or as an editing command .* We alternatively define the above definition in section I

```
Edit ( <number of edges or segments 'n'> and <information about each segment>)

Loop ( for i = 1 to n )
(
        trigger DoubleClickHoldDrag side 'i'
        action
        translate this
        setCursor DRAG
        showHandle MOVE side 'i'
        // where 'i'  is the side number

        trigger holdDrag side 'i'
        action
        translate this
        setCursor DRAG
        showHandle MOVE side'i' … side'n'
)
```

FIGURE 3: Editing Behavior

2.6 CURSER BEHAVIOR

Use of Editing (curser) in MAHI is dependent on its mode that, one can use pen based editing that is when the curser is in pen mode (sketching) in which case the user can draw sketches and when it is in curser mode it means the editing mode. Thus, sketching and editing use distinct pen motions. One of the editing behaviors in MAHI is if one clicks and holds the pen on the surface of the rectangle and drags the pen, the entire rectangle will translate along with the movement of the rectangle. In this way the rectangle along with the vector is translated, scaled, and rotated as one whole shape. All of editing behaviors also change the pen cursor as display to the sketcher to know of his performing an editing command. MAHI editing behavior of triggers include *click, double click, hold, hold drag, encircle, etc*. along with their subsequent application.

2.7 DISPLAY METHODS

Controlling is a vital part of sketching interface when the user recognizes the shape after he has drawn that sketch. MAHI defines recognition layer that has the power of the recognition of the sketch. It gets the input from editing layer as defined above and uses heuristic engine to perform semantic checking for the relevant details from the system database. It also includes the concept of domain independence with reference to multi-domain. It recognizes the output and displays the results to the input/output layer of MHAII interface [2].

2.8 VECTOR

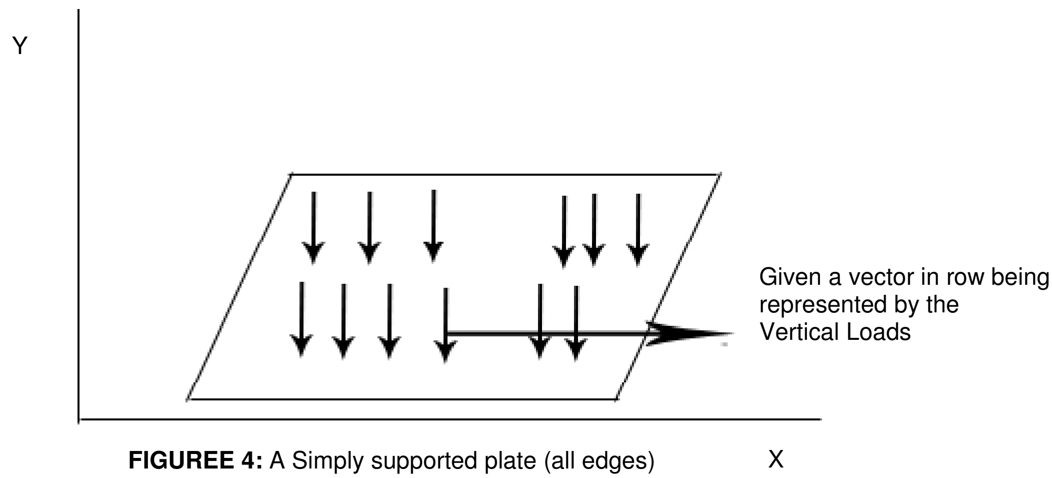
It is defined as a quantity having magnitude and orientation. The magnitude of vector is denoted by $|AB|$ where AB is the length of line segment joining points A&B and orientation is given by the arrowhead $\rightarrow$. Thus, a vector is defined by $\overrightarrow{AB}$. Our discussion with regard to the rectangle given by ABCD is such that $\overrightarrow{AB} = \overrightarrow{CD}$, $\overrightarrow{AD} \perp \overrightarrow{AB}$ as shown in figure 5.

2.9 SCIENTIFIC & ENGINEERING APPLICATION OF SHAPE

We use multi- domain shapes to provide context so that we may more effectively recognize shapes in various domains. A user may specify that when we move a rectangular plate along with the arrow of heads acting on this plate, the arrow should move with the rectangle as shown by the figure 4. For example, in Engineering, we are confronted with the problem of simply supported rectangular plate subject to uniform load governed by the differential equation given by:

$$\frac{Eh^3}{12(1-v^2)}\left(\frac{\partial^4}{\partial x^4}+2\frac{\partial^4}{\partial x^2\partial y^2}+\frac{\partial^4}{\partial y^4}\right)w-q_o=0 \text{ ----------------------------------}(1)$$

Where E, h, v, q_o *and w have the usual meanings.*



FIGUREE 4: A Simply supported plate (all edges)

We further remark here that in reference to editing behavior as defined above, the display of vertical arrow as given in figure 4 and 5 have different meanings such as in figure 4 it represents the load acting on the plate in one domain (Three dimensions) and vertical force in the other domain(Two dimensions).

3. DOMAIN DESCRIPTION OF MAHI

We define domain description as an aggregate of elements. Here elements are referred as the list of domain shapes and group shapes.

3.1 MAHI SHAPE-DEFINITION

A shape is defined if it is associated to a domain belonging to domain description. In particular, geometrical shapes are defined as the blocks belonging to the domain shapes of multi-domain. Now using the above definition alongwith the basic entity as defined by BCE, CE, DE in MAHI given in figure1, we can form new desired shape as:

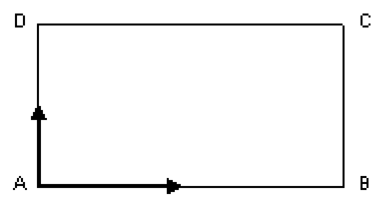


FIGURE 5: Rectangle

Formation of figures 5 is obtained by any of the following two rules:

Rule 1: Description of rectangle ABCD as locus of a point.

The locus of a point A to B horizontally & B to C vertically & C to D horizontally & D to A is vertically is a rectangle ABCD.

Rule 2: Description as inputs of rectangle based on line segment, geometrical constraints, aliases etc.

- In this case, we list the ***components such as AB,AD,DC&CB*** as the side of a rectangle ABCD.
  Equations: of two parallel lines AB and DC are given by
  $a_ix+b_iy+c_i^k=0;i=1,k=1,2$ ……………………………..(2) (by section 2.4)
  Equations of other two parallel lines AD and BC are given by
  $b_ix-a_iy+c_i^{k'}=0;i=2, k'=1,2$………………………………..(3), $k,k' \in R$, since $\overrightarrow{AD} \perp \overrightarrow{AB}$
- The components of the above rectangle are drawn subject to ***geometrical constraints:*** given, as gradient of AB is equal to negative reciprocal of the gradient of line AD. i.e. (gradient of AB)*(gradient of line AD)= -1, here gradient of a line (2 or 3) = -(coefficient(x)) /(coefficient(y))
- Alternative to the above expression of perpendicularity of the lines are also be defined as a set of ***aliases (orientation)*** *for* drawing of a rectangle. In this case, the head and tail have been added as aliases in the arrow definition given by $\overrightarrow{AB} \perp \overrightarrow{AD}$ as shown in figure 5 using specifying editing behaviors.
- Hierarchical shape definition: Four lines with two each of the vectors with a pair of parallel lines for rectangle ABCD of Figure 5. as given by $\overrightarrow{AB} = \overrightarrow{DC}$ and $\overrightarrow{AD} \perp \overrightarrow{AB}$
- MAHI defines the editing layer, which edits the sketch input. This layer includes {Pre-processing of sketch [3] and provides the input to the recognition layer. the functioning of image filtering (Gaussian FIR filter [3]), image segmentation (Canny Method [13] [15]) and image editing [3]. It provides in input to the recognition layer.

- Editing gestures permit us to recognize the system to differentiate between sketching and editing. MAHII language includes a number of predefined editing behaviors using the algebraic constraints associated to the geometry of the figures such as a point, line, arc, etc. The possible editing action include *wait, select, deselect, delete, translate, scale, IsRotate* etc.
- Editing behaviors specifies the gesture of dragging the component in any direction. In this case, the actions of these editing commands specify that the object should follow the pen at the description of the user including translating and rotating. Any arbitrary quadrilateral can be changed to a rectangle as per the action of the user, resulting in the change of coefficient of x and coefficient of y in the domain.
- Display methods indicate what to display when the object is recognized. The sketching shape follows the original component followed by the constraints associated to the other related component to give rise to the extended shape such as line, circle, rectangle, etc. to give the final shape.

The MAHI (Sketching Language) based on the domain description is translated into shape recognizer such as (geometrical) components & constraints. MAHI has the exhibitors and editors representing the display and edit sections respectively to be used in conjunction with the recognition systems to obtain a drawn sketch.

```
define shape Rectangle
description "A quadrilateral with four sides and all angles 90 degree”

Components
Line side1
Line side2
Line side3
Line side4

Clauses          // Constraints
coincident side1.p1 side2.p1
coincident side2.p2 side3.p2
coincident side3.p3 side4.p3
coincident side4.p4 side1.p4

parallel side1 side3
parallel side2 side4
perpendicular side1 side4
perpendicular side1 side2
perpendicular side2 side3
perpendicular side3 side4

equalLength side1 side3
equalLength side2 side4

Aliases

Point side1 side2.p1
Point side2 side3.p2
Point side3 side4.p3
Point side4 side1.p4

Editing

Edit ( side1 side2 side3 side4)
display original-strokes
```

FIGURE 6: DOMAIN-DESCRIPTION OF RECTANGLE (MAHI)

4. TESTING

In domain description, we have defined MAHI as the language being described by a number of symbols belonging to the various domains. MAHI has been successfully applied to around 50 shapes from flow chart and several other domains of engineering or non-engineering.

4.1 MULTI-CONNECTED LINES

Polyhedron is defined in three or higher dimensions, and polygon is defined as a plane shape in 2-dimensions. A polygon is defined as the connected number of line segments and these are formed as the extension of two lines meeting at a point being defined as co-initial bi-lines. Bi-lines form the basis for the extension of polyLines. However, we can draw piece-wise poly-linear or arcular segments as shown in figure 7.

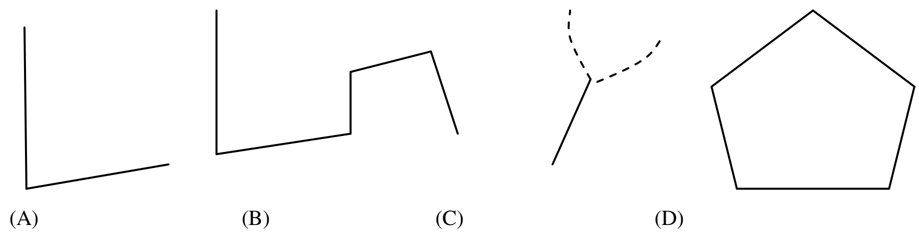


FIGURE 7 : (A) Bi-lines (B) Multi Open Lines (C) Union of lines Segment and Piece – wise arcs, (D) Pentagon

```
define shape MultiLine
description "A set of intersecting lines”

Components
Line side1
Line side2
Line side3
…
Line siden

Clauses          // Constraints
coincident side1.p1 side2.p1
coincident side2.p2 side3.p2
coincident side3.p3 side4.p3
…
coincident side(n-1).p4 siden.p4

Aliases

Point side1 side2.p1
Point side2 side3.p2
Point side3 side4.p3
…
Point side(n-1) side(n).p(n-1)

Editing

Edit (side1 side2 side3 … side (n))

display original-strokes
```

FIGURE 8: Description of MultiLine

```
define shape Bi-Line
description "A pair of intersecting lines"

Components
Line side1
Line side2

Clauses          // Constraints
coincident side1.p1 side2.p1
coincident side2.p2 side3.p2

Aliases

Point side1 side2.p1
Point side2 side3.p2

Editing

Edit ( side1 side2)

display original-strokes
```

FIGURE 9: Description of Bi-Lines

```
define shape Pentagon
description "A plane figure with five straight sides and five angles"

Components
Line side1
Line side2
Line side3
Line side4
Line side5

Clauses          // Constraints
coincident side1.p1 side2.p1
coincident side2.p2 side3.p2
coincident side3.p3 side4.p3
coincident side4.p4 side5.p4
coincident side5.p5 side1.p5

Aliases
Point side1 side2.p1
Point side2 side3.p2
Point side3 side4.p3
Point side4 side5.p4
Point side5 side1.p5

Editing

Edit (side1 side2 side3 side4 side5)

display original-strokes
```

FIGURE 10: Description of Pentagon

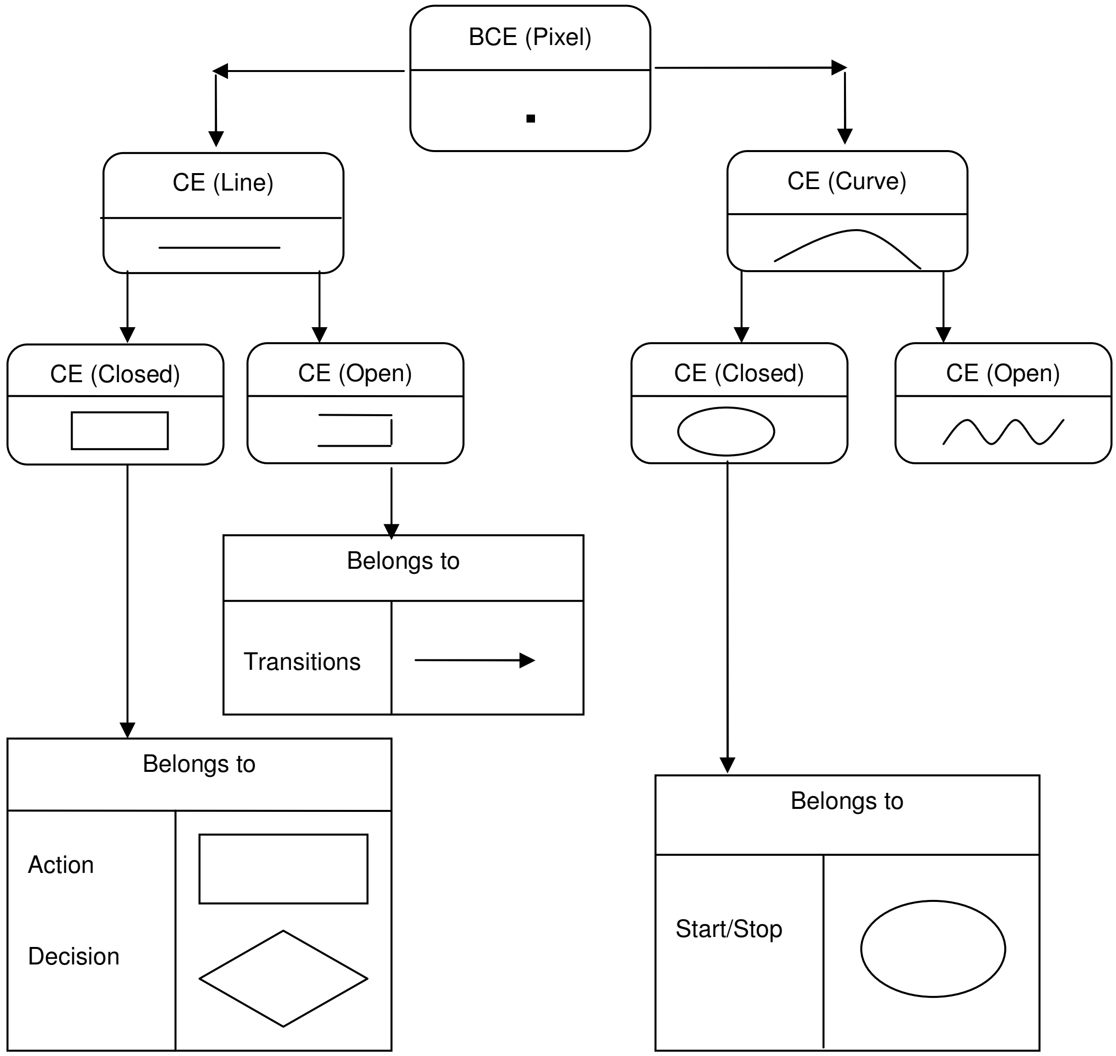


FIGURE 11: Inheritance diagram of Flow Chart diagram

4.2 SYSTEM IMPLEMENTATION

We have drawn the sketches and tested the same to confirm whether these were recognized by our domain-independent recognition.

We built a simple domain-independent recognition system to test whether sketches is recognized from our domain descriptions. The system parses a domain description into MATLAB code and Jess (a rule-based system that interface Java) [Friedman-Hill, 1995] rules, and uses them to recognize sketches. Using the domain description of MAHI, the system successfully recognized hand-drawn sketches as rectangle for any domains (Engineering & Non-Engineering).

Thus any hand drawn sketch can be recognized by parsing to various domains of MAHI.

4.3 PARSING

The domain description MAHI is parsed to produce recognition code, generating one Jess rule (recognition information), and one MATLAB file (shape description), for each shape description. The system uses the Jess rules to recognize sketches. For example figure 5 is shown in figure 12.

```
(defrule RectangleCheck
;;Get the parts of the rectangle
?f0 <- (Subshapes Sides ?s $?s_list)
(Coincident ?s ?s_side1 ?s_side2 ?s_side3 ?s_side4)
(Line ?s_side1 ?side1_p1 ?side1_p2)
(Line ?s_side2 ?side2_p1 ?side2_p2)
(Line ?s_side3 ?side3_p1 ?side3_p2)
(Line ?s_side4 ?side4_p1 ?side4_p2)
;; test for that the coincident edges
(test (coincident ?s_side1_p1 ?s_side2_p1))
(test (coincident ?s_side1_p1 ?s_side2_p2))
(test (coincident ?s_side2_p1 ?s_side3_p1))
(test (coincident ?s_side2_p1 ?s_side3_p2))
(test (coincident ?s_side3_p1 ?s_side4_p1))
(test (coincident ?s_side3_p1 ?s_side4_p2))
(test (coincident ?s_side4_p1 ?s_side1_p1))
(test (coincident ?s_side4_p1 ?s_side1_p2))
;; test for the parallel edges
(test (parallel ?s_side1 ?s_side3))
(test (parallel ?s_side2 ?s_side4))
;; test for the perpendicular edges
(test (perpendicular ?s_side1 ?s_side2))
(test (perpendicular ?s_side2 ?s_side3))
(test (perpendicular ?s_side3 ?s_side4))
(test (perpendicular ?s_side4 ?s_side2))
;; test for the equal edges
(test (equal ?s_side1 ?s_side3))
(test (equal ?s_side2 ?s_side4))

=>
;; Rectangle found successfully
;; Set the aliases
(bind ?side1 ?oa_side1) (bind ?side2 ?oa_side2)
(bind ?side3 ?oa_side3) (bind ?side4 ?oa_side4)

;; Notify recognition system that a Rectangle is recognized
(bind ?nextnum (addshape Rectangle ?s $?s_list ?side1? side2? side3? side4?))
;; Tell the Jess system that a Rectangle is recognized
(assert (Rectangle ?nextnum ?s ?side1 ?side2 ?side3 ?side4 ?))
;; Rectangle is a domain shape. Assert it.
;; Conflicts will be resolved elsewhere.
(assert (DomainShape Rectangle ?nextnum (time)))
```

FIGURE 12: Automatically generated Jess Rule for the Rectangle.

5. CONCLUSION

MAHI language is primarily based on shape and the domain description and can include any type of information that could be helpful to the recognition process. It consist of pre-defined basic shapes alongwith the constraints & editing behaviors for specifying the domain description. Interestingly its mathematical description, apart from describing how sketch diagrams in a domain are drawn, displayed and edited allows the shapes to obtain new shapes hierarchically. Its shape group describes how domain shapes interact and provide information in top-down as well as bottom–up recognition. We are also able to build a simple domain-independent mathematical-rule based sketch recognition system, that tests if the recognition is viable.

ACKNOWLEDGEMENT :

We owe our sincere thanks to Prof. G.V.K. Sinha (Gadde), Chancellor and Dr. Y.S. Goel Director/Principal, Lingaya's University , Faridabad for their keen interest in our work.

6. REFERENCES

[1] G.Sahoo & Bhupesh Kumar Singh, "*A New Approach to Sketch Recognition using Heuristic"*, International Journal of Computer Science and Network Security, Vol. 8 No. 2, 2008., PP. 102-108.,

[2] G.Sahoo & Bhupesh Kumar Singh, *"MAHII :Machine And Human Interactive Interface"* International Journal of Image Processing, Volume (2), Issue (3), 2008 PP. 1-10.,

[3] G.Sahoo & Bhupesh Kumar Singh *"A Human Detector and Identification System"* International Journal of Computer Science, Systems Engineering and Information Technology volume Vol.1(1), June 2008. PP. 39-44

[4] Chien C.F., *"Modifying the inconsistency of Bayesian networks and a comparison study for fault location on electricity distribution feeders"* , International Journal Operational Research, Vol. 1, Nos. ½, pp. 188-202, 2005.

[5] Stiny & Gips, *".Shape grammars and the generative specification of painting and sculpture"* Information Processing, pages 1460–1465, 1972.

[6] Jacob *et al.*, & S Morrison, *"A software model and specification language for non-WIMP user interfaces"* ACM Transactions on Computer-Human Interaction, 6(1):1–46, 1999.

[7] Bimber *et al.*, LM Encarnao, & A Stork, *"A multi-layered architecture for sketch-based interaction within virtual Environments"*. Computer and Graphics, 2000

[8] J.V. Mahoney & M.P.J. Fromherz, *"Three main concerns in sketch recognition and an approach to addressing Them"* AAAI Spring Symposium on Sketch Understanding, PP. 105–112, March 25-27 2002.

[9] A Caetano, N Goulart, M Fonseca, & J Jorge. *"Javasketchit: Issues in sketching the look of user interfaces"* AAAI Spring Symposium on Sketch Understanding, 2002.

[10] Tracy Hammond & Randall Davis. *"LADDER: A language to describe drawing, display, and editing in sketch recognition"* In Proceedings of the 2003 International Joint Conference on Artificial Intelligence (IJCAI-03), Acapulco, Mexico, 2003.

[11] Tracy Hammond and Randall Davis. LADDER, a sketching language for user interface developers. Elsevier, Computers and Graphics, 28:518–532, 2005.

[12] Tracy Hammond & Randall Davis, "LADDER: A Perceptually-based Language to Simplify Sketch Recognition User Interface Development" MASSACHUSETTS INSTITUTE OF TECHNOLOGY January 2007

[13] Jain, A.K., "*Fundamental of Digital Image Processing*", Prentice Hall. *University of California-Davis.,* 1989

[14] S.J.B. Shum, A. MacLean, V.M.E. Bellotti, and N.V. Hammond. Graphical argumentation and design cognition. *Human-Computer Interaction,* 12(3):267-300, 1996.

[15] Michael Shilman, Hanna Pasula, Stuart Russell, and Richard Newton. Statistical visual language models for ink parsing. In *Sketch Understanding, Papers from the 2002 AAAI Spring Symposium,* pages 126-132.,

[16] Tevfik Metin Sezgin, Thomas Stahovich, and Randall Davis. Sketch based interfaces: Early processing for sketch understanding. In *The Proceedings of 2001 Perceptive User Interfaces Workshop (PUI'01),* Orlando, FL, November 2001.