The Knowledge Engineering Review, Vol. 11:2, 1996, 161–189

Review of pattern matching approaches

D. MANFAAT[1,2], A. H. B. DUFFY[1] and B. S. LEE[2]

[1]*CAD Centre,* [2]*Department of Ship and Marine Technology, University of Strathclyde, Glasgow G1 1XJ, Scotland*

Abstract

This paper presents a review of pattern matching techniques. The application areas for pattern matching are extensive, ranging from CAD systems to chemical analysis and from manufacturing to image processing. Published techniques and methods are classified and assessed within the context of three key issues: pattern classes, similarity types and matching methods. It has been shown that the techniques and approaches are as diverse and varied as the applications.

1 Introduction

One of the main characteristics of modern technology is the extensive use of automation. Automation in its more advanced forms demands at least a degree of autonomous decision-making from the machine, and decision-making, if it is to be sensible at all, requires sufficiency in relevant information. We humans also use all kinds of information in our daily life to make decisions of varying degrees of import. The sources of this information are diverse, but fundamentally it is produced by processing the raw data received through our senses. It is natural, therefore, that the key enabling technologies of automation includes the sensors and the techniques to process the data obtained from them.

Visual information is an important element in a wide range of applications, such as industrial robots, land and underwater vehicle navigation and character recognition equipment. Moreover, graphic image processing takes the drudgery out of many arduous tasks in different applications. Cohen & Feigenbaum (1982) give several examples which include medical applications (e.g. screening cancer examinations by tissue image analysis), remote-sensing applications (e.g. satellite image analysis for monitoring natural resources) and industrial applications (e.g. inspecting printed circuit boards).

A key aspect of reasoning with visual information is pattern matching (Rumelhart, 1989), and it can be simply defined as the activity of matching patterns with the aim of finding similarities between them for the purpose of recognition and/or retrieval of similar patterns. One field where pattern matching is involved is pattern recognition. Fu (1976) points out that the problem of pattern recognition usually involves a discrimination or classification of a set of processes, physical objects or events. Pattern matching is used to decide how closely an input pattern "fits" a class of patterns; thus classifying the input pattern into a predetermined class. Another very important field where pattern matching can play a vital role is design, where the process is often greatly assisted by identifying, retrieving and then modifying appropriate past designs, with the implied benefit of being able to utilise relevant past experience (Maher & Zhao, 1987; Duffy & Kerr, 1993).

A number of approaches has been developed in pattern matching, and this paper attempts to classify the techniques and review them critically in order to assist the selection of techniques appropriate for specific tasks and future development. Although it is desirable to include all of the relevant work, for practical reasons only a number of key pieces of work has been selected to examine the main features of pattern matching techniques. These are then assessed within the context of three issues, viz. *pattern classes, similarity types* and *matching methods*.

2 Scope of the review

The range of application areas for pattern matching techniques is extensive. The major areas considered within this paper include mechanical CAD systems (e.g. Schneider et al., 1989; Akutsu, 1992; Peters, 1992), manufacturing (e.g. Kalvin et al., 1986; Schwartz & Sharir, 1987; Meeran et al., 1993), chemical analysis/identification (e.g. Sussenguth, 1965; McGregor, 1982; Akutsu et al., 1991; Akutsu, 1992), image processing (e.g. Cooper, 1989; Karbacher, 1990; Hall & Matias, 1993) and spatial design (e.g. Coyne & Postmus, 1990; Dave et al., 1994; Giretti et al., 1994; Gross et al., 1994; Voß et al., 1994). The selection of these main areas has been based on the diversity of the pattern matching techniques used in each area, given the same object type (pattern class). This situation leads to identification of the wide range of object representations used in each area, and understanding of how each of these diverse techniques work.

Some of the techniques developed may have similarities in certain aspects, but may differ in others. This is hardly surprising as most of them are developed with specific applications in mind. The three main aspects which characterise a pattern matching technique are:

- **Pattern Classes.** This aspect refers to the types of patterns which are used in the pattern matching processes. This will also include the method of representing the patterns to be matched.
- **Similarity Types.** This aspect indicates the degree of exactitude required for a match and the scope of the match. Thus, this aspect helps guide the pattern matching processes, in that these processes may terminate when the matching of patterns meets this aspect.
- **Matching Methods.** The underlying methodology on which the matching processes are based can also have significant effects on the characteristics of the techniques, and thus will be an important issue when considering adoption for specific applications.

3 Pattern classes

The reviewed pattern matching approaches focus on a number of different types of patterns to match. The four basic classes of patterns are: *geometric pattern, topological relations, distributed pattern* and *semantic/symbolic pattern*. Examples of the different representations of the class geometric pattern are *point sets, dimensional graphs* and *drawings*. The class topological relations is represented as a network or a graph made up of vertices, edges and faces. This network or graph represents connectivities of the elements of a pattern. Examples of the different representations of the class distributed pattern are *spatial patterns, spatial relationship patterns* and *graphical patterns*, and of the class semantic/symbolic pattern are *text* and *symbols* (such as diagrams, icons, etc.). Figure 1 gives a diagrammatic representation of this classification.

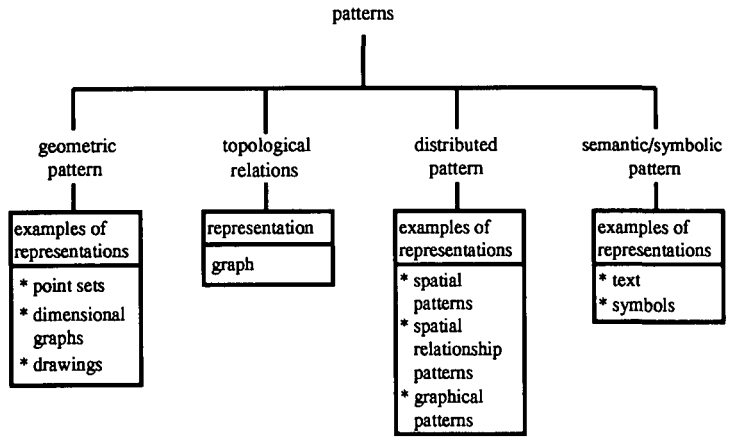


Figure 1 Four basic classes of patterns

Defining classes of patterns indicates the different basic representations of patterns which may be used in pattern matching techniques. Each class of patterns may be used in the different techniques. It is, therefore, worthwhile to describe briefly each of these classes of patterns and present the techniques based on these classes.

3.1 Geometric pattern

Patterns in this class have positions defined in a coordinate system. Thus, *point sets*, as one of the examples of the representations of the class geometric pattern, have points, each of which can be defined by a set of coordinates. Figure 2a gives an example of a point set in a two dimensional (2D) coordinate system. *Dimensional graphs* refer to the graphs whose elements (vertices, edges and faces) are defined in a coordinate system. These graphs could be two dimensional (planar graphs) or three dimensional (polyhedra). Figure 2b gives an example of a planar graph whose vertices are points in a two dimensional coordinate system. Hopcroft and Tarjan (1973) defines that a graph G is planar if there exists a mapping of the edges of G into the plane in such a way that:

(1) Each edge (v, w) is mapped into a simple curve, with v and w being mapped to the endpoints of the curve.
(2) Mappings of two distinct edges have only their common endpoints in common.

Figure 2c gives an example of a polyhedron as the representation of a three dimensional (3D) graph. The *drawings* examples of the class geometric pattern include diverse types of drawings in coordinate systems. These types of drawings may include lines, arcs, circles, polygons, curves and characters. Figure 2d gives examples of various types of drawings in a two dimensional coordinate system.

Kalvin et al. (1986), Schwartz and Sharir (1987), Schneider et al. (1989), Akutsu (1992, 1995), Peters (1992), Meeran et al. (1993), McLaren (1994) and Reihani (1994) have been identified as based on matching geometric patterns. More specifically, Table 1 gives a summary of the pattern matching techniques which use the example patterns of the class geometric pattern.

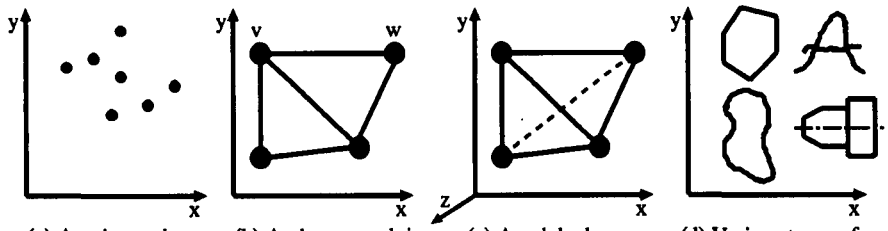


Figure 2 Illustrations of the examples of the class geometric pattern

Table 1 A summary of pattern matching techniques using the class geometric pattern

The Class Geometric Pattern		
Pattern	Two dimensional	Three dimensional
Point sets	McLaren	The first approach in Akutsu(1992) and Akutsu(1994)
Dimensional graphs	The third approach in Akutsu(1992)	The second approach in Akutsu(1992)
Drawings	Schneider et al, Peters, Kalvin et al, Schwartz & Sharir, Meeran et al and Reihani	Schwartz & Sharir

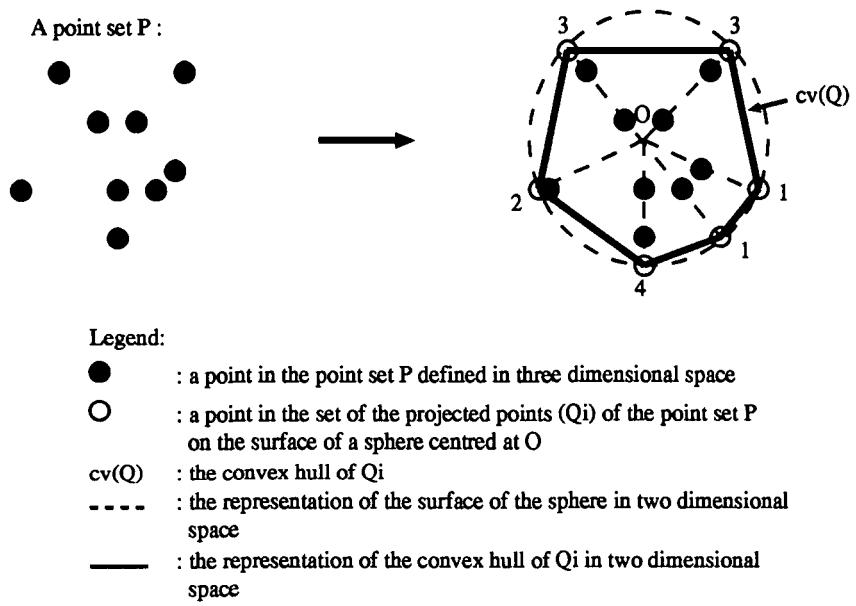


Figure 3 Construction of cv(Q) in two dimensions (adapted from Akutsu, 1992)

In the work by McLaren (1994) a novel self-contained navigation system is proposed for underwater remotely operated vehicles (ROV) operating in and around offshore installations. During the visual inspection and maintenance tasks of the offshore installation it is essential for the operator to know precisely where the ROV is at any given time. In this work, point sets in two dimensional space are used to represent the positions of the components of the offshore installation (such as the legs of the oil rig platform). The matching process in this system is carried out by matching these point set data extracted from a sector scanning sonar device with a computer model of the offshore installation. The result of this process is used to determine the position and heading of the vehicle. This work is comparable to the research in model-based object recognition which is presently providing factory robots with greater abilities to perceive and understand their surroundings.

In Akutsu (1992) there are three approaches. The first approach uses three dimensional point sets to represent mechanical parts and chemical structures. In this approach, two point sets are tested whether they are congruent (similar) or not. Before they are matched the points of each set are initially projected to the surface of the sphere directed to the centre. Then, a "convex hull" is constructed for each set of projected points (see Figure 3). The "convex hull" of a set of points S in dimensional Euclidean space E is the boundary of the smallest convex domain in E containing S (Preparata & Shamos, 1985). For a point set in three dimensional space shown in Figure 3 the convex hull of this set is a convex polyhedron illustrated in two dimensional space. The matching process is then carried out by comparing the convex hulls using Sugihara's algorithm (Akutsu, 1992). This algorithm is used to determine the congruity (similarity) of two polyhedra which is based on the algorithm for planar graph isomorphisms defined by Hopcroft and Tarjan (1973), that is, the algorithm used to determine similarities between planar graphs.

In Akutsu (1995) point sets, which form sequences of points, are used to represent tertiary (three dimensional) structures of proteins. The side chains of protein structures are ignored and only the main chains are considered, since a main chain represents an outline of the shape of a tertiary structure. This work is used for the tasks of substructure search and common substructure search of proteins. These kinds of searches involve matching the outlines of the shapes of protein structures.

The second and third approaches in Akutsu (1992) use dimensional graphs represented in three and two dimensional space, respectively. The second approach is used to test whether two graphs in three dimensional space are congruent or not. For the pattern matching process, each graph to be

matched is transformed into a polyhedron. Then, to test whether two polyhedra are congruent or not, Sugihara's algorithm (Akutsu, 1992) is used. In the third approach, two graphs in two dimensional space are tested to determine whether they are congruent or not. In this approach, each graph representation is transformed into a string representation, thus transfering the geometrical matching problems into the string matching problem.

Representations of objects for pattern matching in work by Kalvin et al (1986), Schwartz and Sharir (1987), Schneider et al. (1989), Peters (1992), Meeran et al (1993) and Reihani (1994) are in the forms of drawings in two dimensional space. In addition, the work by Schwartz and Sharir (1987) is also concerned with drawings in three dimensional space. Most of these drawing representations are for application in mechanical engineering, and drawings, in most cases, represent the models of mechanical parts, except the work by Reihani (1994), which is concerned with drawings of handwritten characters. The common idea of the pattern matching process is to match the attributes (such as the shape, size and orientation) of a stored drawing (or a part of it) with those of an input drawing (or a part of it).

3.2 Topological relations

The class topological relations represents patterns in terms of relations between the elements or parts of the patterns. As has been mentioned in the first paragraph of section 3, this class is represented as a graph made up of vertices, edges and faces. The vertices represent the elements or parts of a pattern and the edges represent the relationships between these elements or parts. A face is made up of a set of closing connected edges and, in most cases, it is used for the purpose of graph matching. This class can be considered as having a different nature depending upon a viewpoint or particular aspect being considered. It differs from the graph representation in the class geometric pattern (i.e. *dimensional graphs*) in that it does not contain geometric information (such as dimensions, shapes and positions) about a pattern, but rather it focuses on the interconnections or connectivities between the elements of the pattern.

Different examples of topological relations, which can be represented as graphs, are spatial relationships of a spatial layout, relationship between points (which are connected by a set of lines) of a geometric figure and atom connectivities of a chemical structure. The graph representation of each of these examples may be as a planar graph or in three dimensional space. In the case of pattern recognition discussed in this paper, only planar graphs are considered. For the geometric figure example, the geometric representation is transformed into a tri-connected planar graph representation, ignoring the geometry (including dimensions, shapes and positions) of the figure. A graph is tri-connected if for each quadruple of distinct vertices v, w, a and b there is a path p: $v - w$ such that neither a nor b is on the path p (Hopcroft & Tarjan, 1973). Figure 4 gives illustrations of the planar graph representation of each of the examples of this class.

Note that in Figure 4 the planar graph representation of the spatial relationships example is represented as a *directed* graph. A directed graph is a graph whose edges are ordered pairs (v, w) of vertices; v is called the *tail* of the edge and w is the *head* (Hopcroft & Tarjan, 1973). Thus, for example, for the edge linking space A and space B, A is the tail and B is the head and the relationships between the two is read as *space A is to the left of space B*. The planar graph representations of the other two examples are *undirected* graphs; the edges are unordered pairs (v, w) of vertices (Hopcroft & Tarjan, 1973). For the geometric figure example, two examples of the possible planar graph representations are given to show that these examples may equally represent the geometric figure since they are graphically similar based on a technique to measure planar graph isomorphisms, such as that defined by Hopcroft and Tarjan (1973). For the chemical structure example, the planar graph representation represents the adjacencies between the atoms.

Work by Sussenguth (1965), McGregor (1982), Alvisi and Odorico (1988), Akutsu et al. (1991), Giretti et al (1994) and Voß et al. (1994) uses planar graph representations of this class. A summary of the pattern matching techniques which use the different examples of the planar graph representations of this class is given in Table 2.

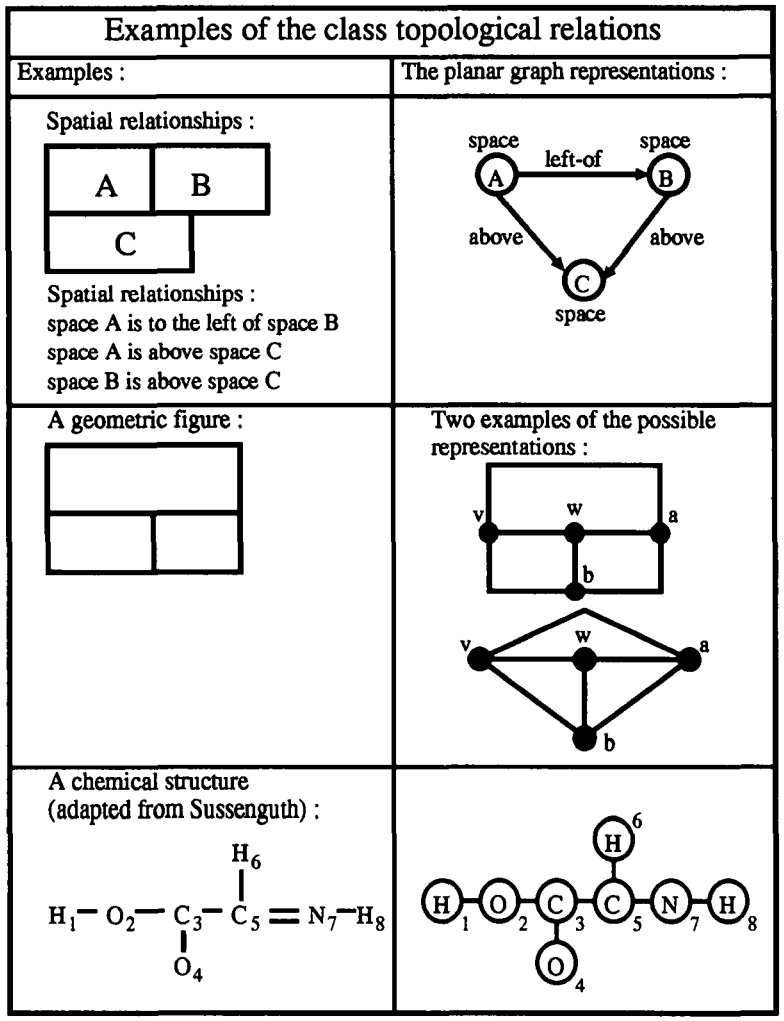


Figure 4 Illustrations of planar graph representations of the examples of the class topological relations

Work by Giretti et al (1994) and Voß et al. (1994) is for application in spatial design. Each has different spatial layout retrieval approaches. One common approach between the two is the layout retrieval using planar graphs. These graphs represent spatial relationships of spatial layouts. In the layout retrieval process, the graph representing the spatial relationships of a spatial layout case is matched with an input graph given by the designer. A technique to measure the planar graph isomorphism between the two graphs may then be used.

The approach developed by Alvisi and Odorico (1988) uses geometric figures which are represented as tri-connected planar graphs. It follows that two different geometric figures may be defined as being similar graphically if their tri-connected planar graph representations are isomorphic based on a technique to measure the planar graph isomorphisms between graphs. For

Table 2 A summary of pattern matching techniques using the class topological relations

The Class Topological Relations	
Examples :	
Spatial relationships	Giretti et al and Voβ et al
Geometric figures	Alvisi & Odorico
Chemical structures	Akutsu et al, McGregor and Sussenguth

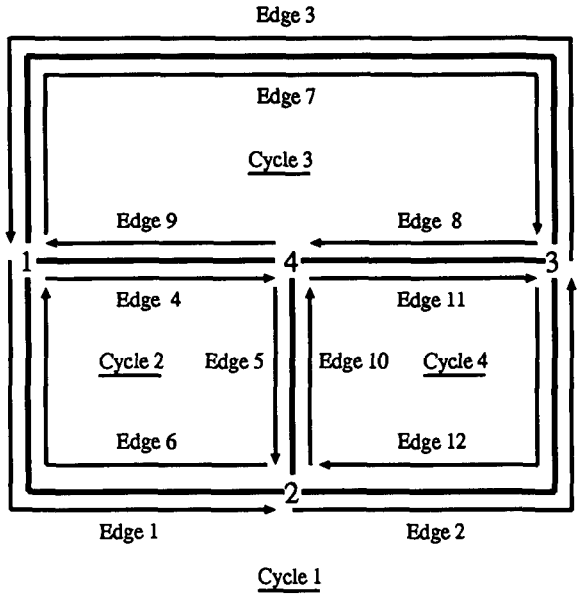


Figure 5 A tri-connected planar graph representing a geometric figure and its labelling (adapted from Alvisi & Odorico, 1988)

the purpose of pattern matching, the tri-connected planar graph of each geometric figure is represented as a directed graph (see Figure 5). A sequence of the directed edges which starts and ends in the same vertex is called a cycle of which there are two types (Alvisi & Odorico, 1988):

(1) The external cycle, made up of all the edges belonging to the external border of the graph, oriented counter-clockwise (e.g. Cycle 1).
(2) The internal cycles, oriented clockwise, of minimum area, i.e. such that no other cycle can lie within the region delimited by them (e.g. Cycles 2, 3 and 4).

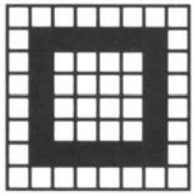
The matching between two geometric figures is carried out by exploring the correspondences between the edges of the directed graphs based on the similarity of the degree of the tail and head vertices, respectively, of the edges of the graphs being compared. The degree of a vertex is the number of edges coincident in this vertex. For example, in Figure 5 the degree of each vertex is 3.

Work by Sussenguth (1965), McGregor (1982) and Akutsu et al. (1991) uses the chemical structure example. Figure 4 also gives an example of a chemical structure and its planar graph representation. Note that in this representation each node represents an atom of the structure and each edge represents a bond or relation between two atoms. In addition, each node is labelled. This is used for the process of exploring the correspondences between the nodes of two planar graphs when attempting to find the isomorphism between these graphs.

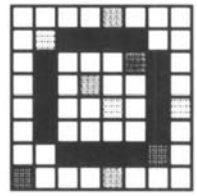
3.3 Distributed pattern

The term "distributed pattern" is coined here to refer to the pattern distributed across a matrix of grid units or of pixels in computer monitors. As has been mentioned in the first paragraph of section 3, examples of the different representations of this class are *spatial patterns, spatial relationship patterns* and *graphical patterns*. The spatial patterns refer to the patterns which form the shapes of spaces. Thus, each grid unit or pixel of a pattern represents a part of the shape of a space. Figure 6 gives different examples of spatial patterns on 8 × 8 pixel backgrounds.

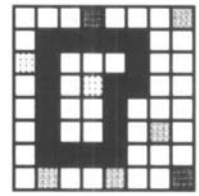
A spatial relationship pattern refers to a pattern which represents the relationships between the spaces of a spatial layout as illustrated in Figure 7. In this pattern, each grid unit or pixel represents the adjacency between two spaces. It may also represent the position of a space relative to the adjacent space.



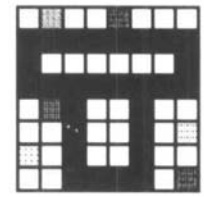
(a) A square pattern on an all white pixels background



(b) An O-shaped pattern on a noisy 8x8 pixels background

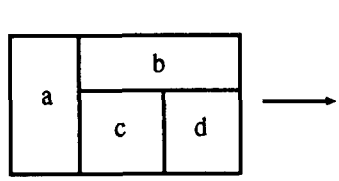


(c) An L-shaped pattern on a noisy 8x8 pixels background

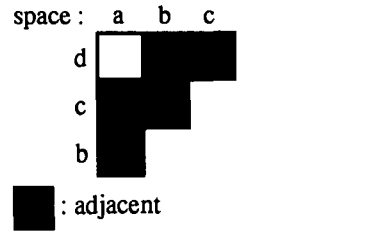


(d) A T-shaped pattern on a noisy 8x8 pixels background

Figure 6 Different examples of spatial patterns on 8 × 8 pixel backgrounds



(a) Spatial layout



(b) Pixel representation of the spatial relationship pattern of (a)

Figure 7 An example of pixel representation of spatial relationship pattern

The graphical patterns are coined here to refer to any pattern other than spatial patterns or spatial relationship patterns. The example of a graphical pattern is not included in Figure 6 as, like the spatial pattern, it basically has the representation where each grid unit or pixel represents a part of the pattern, but it does not form a spatial pattern.

Table 3 gives a summary of pattern matching techniques which use the examples of the class distributed pattern. As can be seen in this table, the first approach in the work by Coyne and Postmus (1990) and one of the different layout retrieval approaches in the work by Voß et al. (1994) use spatial patterns which represent the floor plan layouts of building designs. Matching two patterns is carried out by comparing the grid units or pixels making up the shapes of the patterns. The second approach in Coyne and Postmus (1990) uses spatial relationship patterns. This approach matches the spatial relationship patterns of two floor plan layouts by comparing the grid unit or pixel representations.

Patterns which are distributed across a matrix of pixels in computer monitors are closely related with the field of image processing and pattern analysis. In most cases, pattern matching in this field is carried out by matching images (or pictures), with a template (Cooper, 1989). Hence, this

Table 3 A summary of pattern matching techniques using the class distributed pattern

The Class Distributed Pattern	
Examples :	Coyne & Postmus and Voβ et al
Spatial patterns	
Spatial relationship patterns	Coyne & Postmus
Graphical patterns	Cooper, Hall & Matias, Karbacher, Korn and Teo & Sim

approach has often been termed "template matching". The matching process is carried out by applying the pixels of the template at different positions on the pixels of the picture, and aims to find a part of the picture which is similar to the template. Cooper (1989), Karbacher (1990), Korn (1990) and Hall and Matias (1993) investigate template matching for graphical patterns which range from simple patterns, such as characters (Cooper, 1989), to more complex patterns, such as virus images (Hall & Matias, 1993). In both template and picture, a black pixel represents 1 (i.e. part of the image), a white pixel 0 (i.e. the background) and a grey one has a value between 0 and 1 (i.e. either part of the image or noisy background). The picture and template may have high resolution (all the pixels forming them are "black") or low/noisy resolution (the pixels are the combination between "black" and "grey").

Patterns in the work by Teo and Sim (1995) are also represented in a two dimensional matrix of grid units, called an array of receptor cells. This work explores the use of matching in recognising postal codes on envelopes, either printed or, primarily, written. Since handwritten characters of post codes can be in different positions, orientations (i.e. rotated) and scales, this work, using neocognitron networks (Fukushima & Miyake, 1982; Fukushima, 1988), which is an implementation of neural networks, aims to determine a methodology of obtaining an optimised set of parameters for the network to recognise these rotated and scaled handwritten characters.

3.4 Semantic/symbolic pattern

The class semantic/symbolic pattern represents the semantic/symbolic attributes of objects, such as the names, functions, sizes, colours and other features. These attributes can be represented in text or symbols (such as diagrams, icons, etc.). Those represented in text can be associated with values which form the attribute-value pairs representing the objects. These attribute-value pairs are in most cases represented in the form of lists. Those represented in symbols can be represented in lists as well.

Work by Dave et al. (1994), Giretti et al. (1994) and Gross et al. (1994), which all are in the field of case-based design, and Voß et al. (1994), in the field of access and retrieval in design, uses this class for pattern matching as a part of the retrieval mechanism. In Giretti et al. (1994) this class is used in addition to the planar graph representation which represents the spatial relationships of a spatial layout as presented in section 3.2. In Voß et al. (1994), this class is used in addition to the planar graph representing the spatial relationships of a spatial layout and spatial patterns representing spatial layouts as presented in sections 3.2 and 3.3, respectively.

In the work by Gross et al. (1994), symbolic patterns used in this work are diagrams. This work links "a computer as a cocktail napkin" program that interprets hand-drawn sketches and diagrams with Archie III, a case-based design aid, to support case-based reminding in conceptual design (see Domeshek & Kolodner, 1991, 1992). A sketch is made up of a set of primitive elements, which are called *glyphs*. Examples of glyphs are lines, circles, boxes, triangles, arrows, numbers, letters, etc. The sketching part of the linked program is used to recognise hand-drawn diagrams, which may include different kinds of glyphs, spatial relations between the elements of a diagram and characteristic configurations of glyphs, such as a tree-diagram or a bubble diagram. This part also incorporates a catalogue of diagrams which forms templates with which an input diagram is matched.

Archie III contains a case base which includes stories, problems and responses. In addition to the illustrations of stories in this case base using text, photographs, and drawings, the stories are indexed with simple diagrams. Each story which is diagrammatically indexed points to one or more diagrams in the diagram catalogue of the sketching part. To retrieve a story the sketch program's graphical search routine is set up to search the diagram catalogue. When it finds a matching diagram, the sketch program signals Archie III to retrieve the story associated with the same diagram.

Dave et al. (1994) present a system that enables case adaptation and case combination of building design cases to generate new design solutions more efficiently. The approach taken in this

work has much in common with some of the key issues that are typically associated with case based reasoning (CBR) (Kolodner, 1993) systems in general. The system has a case base in and from which design cases can be stored and retrieved. In this case base, each case is represented as a complete three-dimensional geometric model of a design associated with ordered symbolic information. The three-dimensional geometric model is represented using a CAD modeler which acts as a graphics engine. The symbolic information is represented as association lists attached to relevant graphic entities. To select a case, given a new design problem, the symbolic information of the new site description is input and matched with the symbolic information of the case using the association list representation.

Giretti et al. (1994) propose the Architectural Symbolic Assistant (ASA) system, an interactive assistant to architectural design. This system can be used to assist architects in addressing architectural design aspects which include the domain knowledge, such as the function-structure model of a building, and different knowledge sources, such as the aesthetic aspect of the building. The ASA system integrates a graphic subsystem with a case based reasoner. The graphic subsystem supports constraints among parameters of the primitive graphic entities (a set of parametric architectural objects). The case based reasoner consists of two components related to each other: the scene base and theory base. The scene base contains a set of design solutions (scenes) which are spatial aggregates of architectural objects, such as houses or parts of a house. The class semantic/symbolic pattern in the form of text is used to represent a set of parameters which describe every component of a scene.

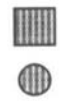
The theory base contains a set of theories that are collections of partial descriptions of scenes. These descriptions form abstractions over scenes. There are two partial descriptions of scenes in the theory base: the adjacencies between compartments of the scene, represented by an adjacency graph as has been presented in section 3.2 and the relations between the access and other compartments of the scene, represented in symbols.

The FABEL project (the work by Voß et al., 1994) developed an approach to retrieve similar CAD(-like) layout drawings, and pattern matching is used to find the most similar architectural layout drawings in the retrieval process. As mentioned in section 3.2, in this project different retrieval approaches have been proposed, some focusing on the retrieval of semantic attributes, others on that of geometric attributes of the layout drawings. The approaches in FABEL which use planar graphs and spatial patterns to represent the geometric attributes have been presented in sections 3.2 and 3.3. The other approaches use the class semantic/symbolic pattern in the form of text and symbols to represent the semantic attributes. The major common semantic attributes are the subsystem (fresh air, used air, construction, etc.) to which an object belongs, its function or morphology (connection, equipment, etc.), its resolution or degree of abstraction (zone, bounding box, concrete component) and its scale or size.

4 Similarity types

A pattern matching process involves, at one stage or another, judging whether an object is similar to another. The soundness of this judgement of similarity is crucial to the whole pattern matching process, and therefore before discussing matching methods a more precise description of the concept of similarity is necessary.

Rips (1989) and Smith and Osherson (1989) show a common idea of looking at similarity from the point of view of the role it plays in psychological processes, such as categorisation and decision making. Rips (1989) gives a simple and appealing idea about the way people decide if an object belongs to a category by pointing out that 'the object is a member of the category if it is sufficiently similar to known category members'. Smith and Osherson (1989), in contrast, 'extend certain theoretical notions about similarity—particularly notions about similarity to prototypes—to studies of decision making and choice'. A model of similarity and prototypes which describe those situations wherein a person estimates the probability that an object belongs to a class by relying on the similarity that the object bears to the prototype of the class is proposed. Another proposal, by

Type	Pairs of objects			Similarity			
	a	b	c	Attributes	a	b	c
1. Resemblances (part-similarity) --- sets of objects that are "alike". That is, there is some kind of similarity present either in part or whole.				Colour	×	√√	×
				Shape	√√	√√	√
				Size	×	√√	√
2. Overall similarity --- sets of objects similar overall. That is, more attributes are similar than dis-similar.				Colour	√	√√	√
				Shape	√√	√√	√
				Size	×	√√	√
3. Identity --- equivalence sets of identical objects. That is, all attributes are the same/identical.				Colour	√√	√√	√√
				Shape	√√	√√	√√
				Size	√√	√√	√√
4. Part-similarity --- sets of objects similar in part. That is, some attributes are similar.				Colour	×	×	√
				Shape	√√	×	√
				Size	×	√	√
5. Part-identity --- sets of objects identical in part or sharing common particular attributes. That is, some attributes are the same/identical.				Colour	√√	×	×
				Shape	×	√√	√√
				Size	×	×	√√

Key: √√ = Same √ = Similar × = Dis-similar

Figure 8 Illustrations of concrete sameness relations (adapted from Smith, 1989)

Barsalou (1989), deals with the question of instability (or flexibility) in people's conceptual representations and its implications for a theory of similarity (Vosniadou & Ortony, 1989).

More specific concepts of similarity, proposed by Smith (1989), define different types of similarity which may exist between objects. Smith uses the term "relations of sameness" to denote the similarity between objects. These relations are partitioned into two kinds: *concrete relations* between objects and *abstract relations*.

Concrete relations of sameness between objects are relations of sameness between these objects in terms of their attributes. Abstract relations of sameness are relations that organise the concrete sameness between objects. That is, an abstract relation forms a relational set which contains sets of pairs of objects, all having the same class of the concrete sameness. Smith classifies the concrete relations of sameness between objects into five classes: *resemblances, overall similarity, identity, part-similarity* and *part-identity*. Figure 8 shows these five classes of concrete relations of sameness which may exist between objects. In this figure, each class of relations is illustrated with three examples, each showing the relationship between two objects in a pair. The attributes of each object in the pair, such as the colour, shape and dimension, are used to compare the objects. These five classes of the concrete relations of sameness are discussed in the following subsections. It can be noted, however, that in Figure 8 the class *part-similarity* is defined for two objects whose particular attributes are considered similar, whilst *part-identity* is defined for two objects which are identical in their particular attributes or share common particular attributes.

It is the concrete relations which are adopted to classify the similarity types used in the pattern matching approaches into their five classes. The reason for this is that the approaches use the attributes, such as the shape, dimension, position, orientation, etc., of the patterns for pattern matching. The way the approaches use the attributes of the patterns for pattern matching is the same as the way Smith's (1989) proposal defines the classes of the concrete relations of sameness between objects, that is, it uses the attributes of the objects. Thus, it is relevant to use these classes to classify the similarity types used in the approaches. In addition, it is considered in this paper that

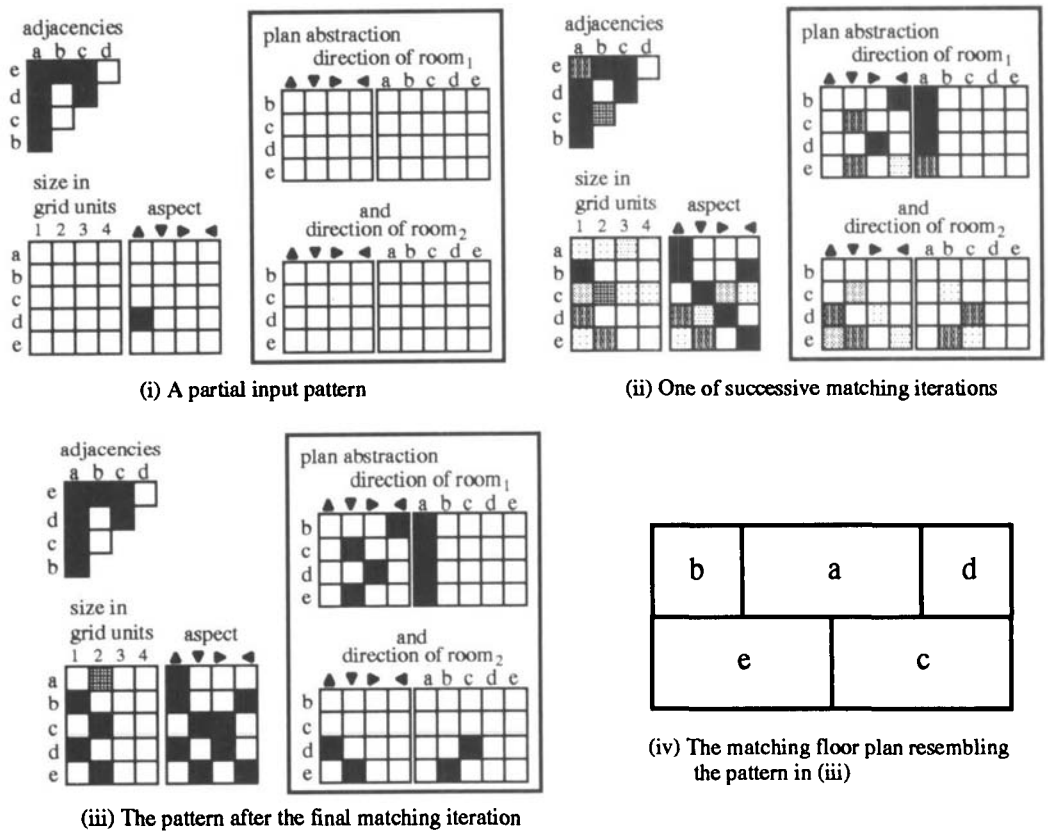


Figure 9 An illustration of matching spatial adjacencies and aspect requirements in matrices of two-dimensional grid units (adapted from Coyne & Postmus, 1990)

the terms *part-similarity* and *part-identity* also mean the similarity and identity, respectively, between the parts of objects or between a part of an object and another object.

4.1 Resemblances

This class is an all-encompassing relation that includes all the other relations of sameness and is, therefore, similarity at its most "undisciplined"—unconstrained and unspecified (Smith, 1989). More specifically, in terms of the development of the various relations of sameness, resemblance is the very early relation where the other relations may not be differentiated from each other. As the development goes on, the other relations may be defined, as Smith (1989) points out that specific kinds of similarities may gain differential status only with development. This implies that the similarity type used in the initial stage of matching two objects in each of all the reviewed pattern matching approaches is resemblances.

As an example, Coyne and Postmus (1990) describes matching spatial relationships in the domain of floor plan design. Each of a set of floor plans is represented in matrices of two-dimensional grid units in terms of its attributes: the adjacencies between spaces, the size in grid units of each space, the aspect of each space (that is, the directions faced by their clear sides) and the "plan abstraction". The plan abstraction is a representation of a sequence of actions describing the construction of the plan (that is, the relative positions of each space to the others) assuming one space is already in place. For matching spatial relationships to find the matching stored patterns of floor plans the approach uses neural networks as the matching method. Figure 9 gives an illustration of matching spatial adjacencies and aspect requirements in matrices of two-dimensional grid units. Given a partial input pattern of spatial adjacencies and the aspect of one of the spaces (i.e. space *d*) as given in (i), the neural network system, after successive iterations, produces the

pattern shown in (iii). The matching stored floor plan which resembles the pattern in (iii) is found and shown in (iv).

4.2 Overall similarity

The overall similarity is a whole-object relation that takes into account all of an object's characteristics at once, and no particular attributes are emphasized. A brown shoe and a black boot would be holistically similar; a brown shoe and a brown wagon would not. Overall similarity is the relation between objects that are (perhaps) discriminably different but also highly similar overall, for example, two chairs or two dogs (Smith, 1989).

Cooper (1989), Schneider et al. (1989), Karbacher (1990), Korn (1990), Hall and Matias (1993), Dave et al. (1994), Giretti et al. (1994), Gross et al. (1994), McLaren (1994), Reihani (1994), Voß et al. (1994) and the first approach in Akutsu (1992) use overall similarity in matching the object patterns. In addition, the work by Coyne and Postmus (1990) also uses overall similarity in matching spatial patterns, apart from resemblances in matching spatial relationships as discussed in section 4.1.

For example, McLaren (1994) attempts to match the data features, extracted from a sector scanning sonar device, to the model elements of an offshore installation using geometric constraint checks. The matching process is carried out by trying one-to-one mapping between each data feature and each model element. A binary constraint check, an instance of the geometric constraint checks, is used to ascertain whether the distance between two data features lies within the range of distances between the corresponding model elements. Thus, it can be observed that the class of sameness used in the matching process is the overall-similarity.

4.3 Identity

Like the overall similarity relation, the identity relation is also a whole-object relation that takes into account all of an object's characteristics at once. It is different from the overall similarity relation in the degree of sameness. If two chairs or two dogs are considered to be overall similar, the relation that exists between two dinner plates in a matched set of china can be considered as being an example of the whole object identity relations (Smith, 1989). This is because the shape, size and pattern (i.e. all the defined attributes) of the plates are identical.

The work by Sussenguth (1965) uses identity as the criterion in matching chemical compounds. Chemical compounds are represented by graphs and two graphs are tested to determine whether they are isomorphic. It they are, then it can be said that these graphs are identical, since the components (the number and the degree of vertices) of all the subgraphs of one graph are identical with those of another graph.

4.4 Part similarity

The part similarity relation is concerned with the constituent attributes of objects, i.e. particular aspects or attributes are emphasised in the comparison. An example of this is the relation between a red ball and a pink thread (Smith, 1989), where the two colours may be considered as having a degree of similarity with red being an attribute of one and a part attribute of another.

Kalvin et al. (1986), Schwartz and Sharir (1987), Peters (1992), Akutsu (1995) and Teo and Sim (1995) use part similarity for matching. The common idea of their work is to identify parts of an object which are similar in shape, size or position to a given input object (or a part of it). As an example, in Akutsu (1995), the search for the substructure and common substructure of proteins use part similarity in matching the structures. The attribute used to compare one structure to another is the shape of the structure. Figure 10 shows an illustration of the common substructure

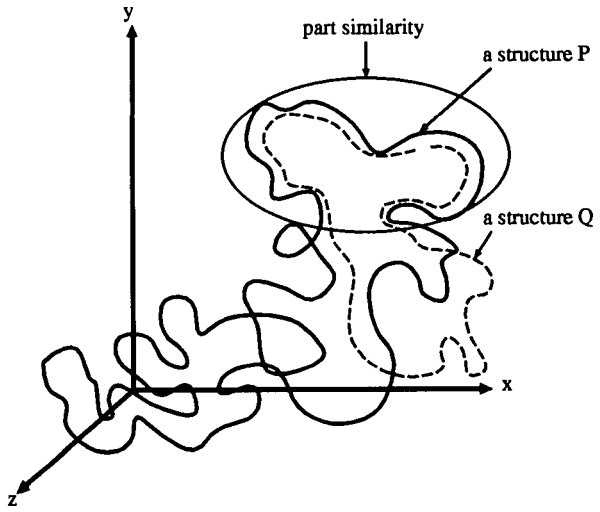


Figure 10 Common substructure search between two structures P and Q

search between two structures P and Q. Note that in this figure, the shapes of the two structures match partially; thus, they are part similar.

4.5 Part identity

The part identity relation is also concerned with the constituent attributes of objects. As with whole-object identity and overall similarity, part identity differs from part similarity in the degree of sameness. If a red ball and a pink thread are considered as being part-similar, a red ball and a red thread are part-identical (Smith, 1989). This is because in the latter object comparison the two objects share the common colour attribute, i.e. red.

Sussenguth (1965), McGregor (1982), Alvisi and Odorico (1988), Akutsu et al. (1991), Meeran et al. (1993), Giretti et al. (1994), Voß et al. (1994) and the second and third approaches in Akutsu (1992) use this criterion in matching object patterns. As an example, in Alvisi and Odorico (1988), geometric figures are represented as topological relations, i.e. as planar graphs. The measure of identity between an input figure and the parts of a geometric figure to be matched is based on the isomorphism between the graph of the input figure and the subgraphs of the matched figure. If they are isomorphic, then it can be said that the matched figure is part identical to the input figure. Figure 11 shows part identity between the input pattern (the triangular figure) and the parts of a geometric figure from the viewpoint of topological relations, i.e. planar graphs, as all the input pattern and the parts are isomorphic or graphically identical.

5 Matching methods

A matching method basically refers to a method used in the process of matching patterns with the aim of defining the degree of match between the patterns. In the process of matching, it establishes the correspondence between input patterns derived from the external world of objects and patterns previously represented in the computer. As the output of this process, it defines the degree of match or similarities between the patterns.

In relation to the previous two sections it is worthwhile to discuss the different matching methods which are used in the reviewed pattern matching approaches. This will indicate the diversity of the matching methods which may be used given a class of patterns and the different types of similarity between patterns which the method in each approach may use. The different matching methods which have been identified can be considered under the headings of *neural networks, mathematical techniques* and *symbolic pattern matching*. They are discussed in the following subsections.

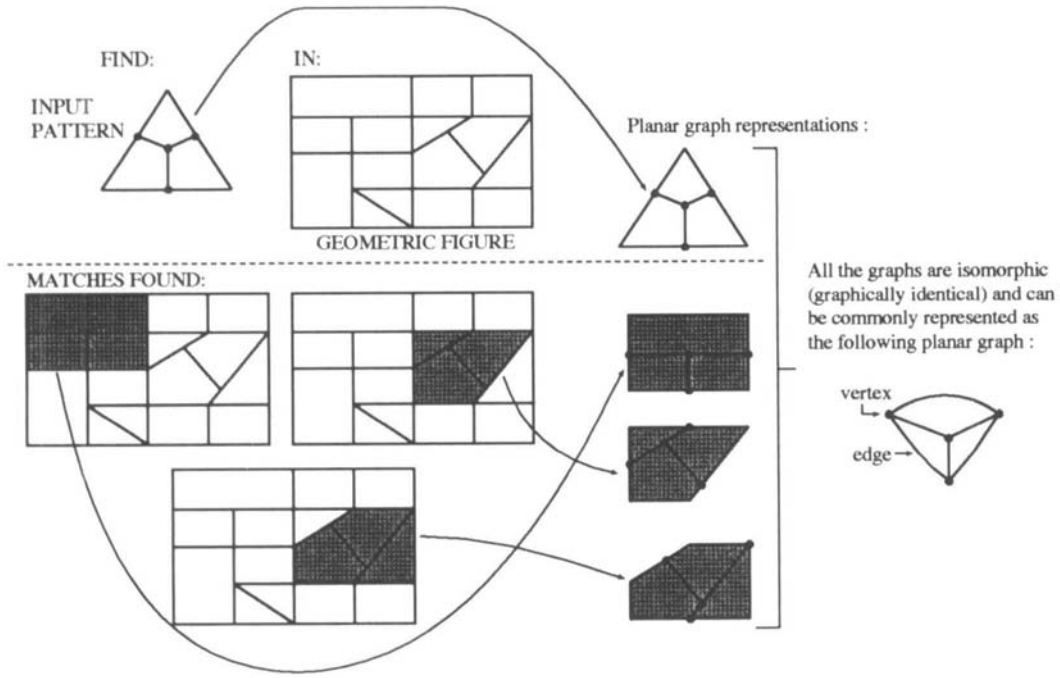


Figure 11 An illustration of part identity between a geometric figure and an input pattern from the viewpoint of topological relations (adapted from Alvisi & Odorico, 1988)

5.1 Neural networks

Neural networks are mathematical models that shadow the microstructure of the brain (Coyne & Newton, 1989). Neural network models, which are also often called *parallel distributed processing* (PDP) systems (Rumelhart, 1989), consist of very large networks of processing units (nodes) and weighted arcs connecting the nodes. The network stores knowledge of a particular application domain. In the pattern matching process, the neural network system matches an input pattern with the previously stored patterns and after successive iterations the system finds the matching pattern. The overall merit of using the neural network model is that the model resembles the functionality of the human brain. As pointed out by Rumelhart (1989), 'our ability to pattern-match is probably not something that sets humans apart from other animals but is probably the essential component to most cognitive behaviour'. As the overall drawback of using the model, Coyne and Postmus (1990) point out that 'destruction or modification to one part of the network generally results in uniform degradation of the performance of the system rather than the loss of any particular item of memory'. It follows that the model is sensitive with the locally changed pattern. That is, a local change to the input pattern may cause the model to produce a different output pattern which may not be expected.

Coyne and Newton (1989) argue that 'one of the claims being made about neural network models is that, in spite of their simplicity, they exhibit properties that could be considered essential features of human cognition: the idiosyncratic way they "learn", their reliance on "memory" and their abilities to deal with partial information'. They can also appear to simulate human pathologies, such as amnesic syndromes, interference phenomena, cross talk and blending errors (McClelland & Rumelhart, 1987; Clark, 1989). With abilities to exhibit these features, these models can be used to deal with such tasks as learning experiences, storing them in "memory", perceiving information, retrieving experience from "memory", comparing it with perceived information and concluding the correspondence between the experience and perceived information. These are all features required for pattern matching (Rumelhart, 1989; Coyne & Postmus, 1990).

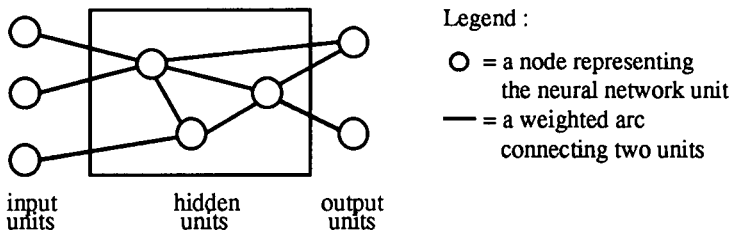


Figure 12 An example of a neural network model (adapted from Coyne & Postmus, 1990)

The way neural networks consider input information, search, retrieve and use experience, or information previously stored in "memory" to match the input information, basically resembles human reasoning processes. In relation to this, Coyne et al. (1989) define neural networks as being reasoning devices based on the uniform distribution of information across networks of relatively simple elements (weighted arcs and units). Each weighted arc connects two units. Figure 12 gives an example of a neural network model. The simple elements of the model in this diagram can be used to store knowledge of application domains.

Apart from the weights on the arcs connecting units, there are some other parameters used to represent the knowledge which is stored, such as input, output and threshold values on the units. The threshold value is the value of a unit with which the net value resulting from the connected units is compared to produce the output value of the unit. In this figure input units receive values from the "outside world", and output units transmit their values to the world, whilst hidden units only receive and transmit values to other units (Coyne & Postmus, 1990).

Coyne and Postmus (1990) point out that 'it is simplest to think of units as receiving binary inputs and producing binary outputs. The output value of a unit (1 or 0) is a function of the sum of the products of the weights of arcs directed towards the unit and the output value of the connected unit. A net value greater than the threshold generally means that the unit will "fire" (producing a value of 1). Net values less than or equal to the threshold produce a 0. This kind of operation is propagated throughout the network to produce a configuration of output values in response to a given set of inputs. Learning involves automatically adjusting the weights and thresholds to account for all input and output pairs presented to the system'.

As an example, consider a simple network in Figure 13, which consists of three input units and one output unit. It can be seen that each input unit is connected to the output unit with a directed arc of weight 1. The output unit has a threshold value (θ) of 0. The input pattern (1 0 1), which means that the values of the input units are 1, 0 and 1, produces 1 as the output value of the network. This is because the sum of the product of each input unit with each weight (to produce a net input) is 2, which is greater than the threshold value 0 of the output unit.

There are many published works and textbooks dealing with neural networks and interested readers should refer to e.g. Coyne and Newton (1989) and Coyne and Postmus (1990) for details.

For pattern matching, an input pattern is given to the system to produce an output pattern which matches the input. For this purpose, the system initially learns a set of stored patterns with which the input would be matched. Neural networks used for pattern matching are found in work by

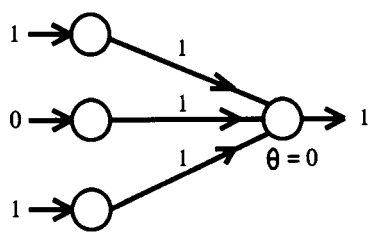


Figure 13 An example of a simple neural network (adapted from Coyne & Postmus, 1990)

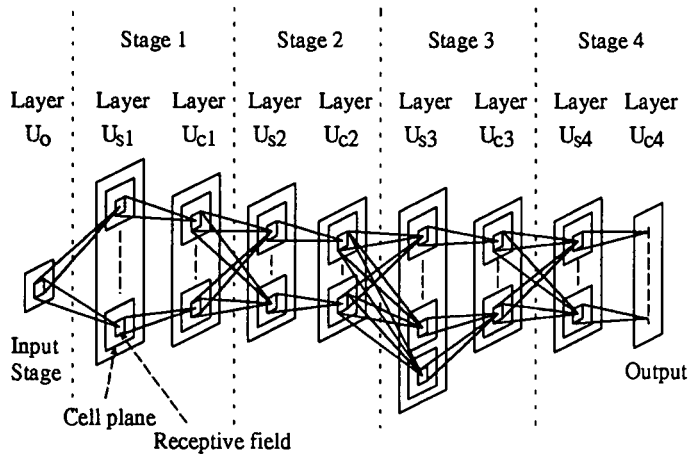


Figure 14 An Architecture of neocognitron neural network (adapted from Teo & Sim, 1995)

Coyne and Postmus (1990) and Peters (1992) in design, Korn (1990) for pattern analysis and image enhancement, and Teo and Sim (1995) in handwritten character recognition and scene analysis.

A major drawback of the standard neural network approach is that once a network has "learned" to match specified input values with particular outputs, then a change in the input would result in a mismatch of the output. Thus, patterns which are essentially the same but have been rotated or scaled would be regarded by the "trained" network as being a new pattern and would result in a mismatch.

As has been mentioned earlier, Teo and Sim (1995) use the neocognitron neural network to recognise handwritten characters of postal codes on envelopes. Neocognitron is a hierarchically structured multi-layer neural network that is recognised for its ability to tolerate scaling and rotational effects of the input patterns. It was originally proposed by Fukushima (Fukushima & Miyake, 1982; Fukushima, 1988). The architecture of the neocognitron network is illustrated in Figure 14. The network consists of neuron-like cells simulating those in the human brain. The network is made up of a number of stages (Stages 1, 2, 3 and 4), with each stage having two layers (Layers Us and Uc), except the input stage which has only one layer (Uo). The basic idea of a hierarchical network is to have every processing element on each layer of the network only receive connections from a restricted, localised subset of the previous layer's processing elements (see Fukushima, 1988). This subset of elements is known as the receptive field or receptor cell. Collections of receptor cells organised in two-dimensional arrays are called "cell-planes".

Only the input stage has one cell-plane. The cells are used to store the binary representation of the input image. This is the pattern required to be recognised or matched. Each of the succeeding stages after the input stage has a layer of "S" cells (simple cells) followed by a layer of "C" cells (complex cells). These S-cells and C-cells in each layer are grouped into cell-planes. These cell-planes containing S-cells and C-cells are thus called S-planes and C-planes, respectively.

In the network, layers of S-cells and C-cells are arranged alternately. In this arrangement, the function of the S-cells is to extract a particular feature, whilst that of the C-cells is for the tolerance of the positional shift of the feature. The processes of feature-extraction and the tolerance of the positional shift of the feature are repeated. Teo and Sim (1995) point out that 'during these processes local features extracted in a lower stage are gradually integrated into more global features. Finally, each C-cell of the highest stage integrates all the information of the input pattern, and responds only to one specific pattern. Thus, the responses of the C-cells of the highest stage show the result of pattern recognition of the network'.

The overall merits and drawbacks of using neural networks have been given in the opening paragraph of this subsection. There are further strengths and weaknesses of using this method for pattern matching applications. The strengths can be summarised as follows:

- Neural network models are capable of pattern completion which means that they can recognise a pattern in partial or noisy data and output a new pattern which completes the pattern. This capability is especially useful for image recognition although other applications, including design, computer vision and process control, can also benefit from it.
- In the design application, neural network models can be viewed as being a form of associative or analogical reasoning with a store of design examples (Coyne & Postmus, 1990). Here, analogical reasoning refers to reasoning with the expected properties of a new design from the store of design examples and their performances based on analogy between these examples and the new design.
- Neural network models can be used to generalise patterns since, given a partial pattern to be matched with a store of patterns, the output produced may represent the combinations of the stored patterns. Generalisation, here, basically means classification of a set of patterns into a class, of which the process is essential for effective pattern matching. Thus, neural network models can be used for pattern classification.
- The neocognitron network, one of the implementations of neural network models, can be used for scale and rotation invariant matching. That is, it recognises an input pattern regardless of the input's position, scale and rotation/orientation.

Coyne et al. (1989) discussed the weaknesses of using neural network models and outlined them as follows:

- 'There are considerable implementation problems in applying neural network modelling techniques. Neural networks involve the simulation of parallel processes, entropy calculations, and the optimisation of parameters within nonlinear systems. Simplifications are generally made, but the processes are still computationally expensive. Thus, these techniques require faster and cheaper computational power if they are to be practical'.
- 'The theories that lead to predictions of behaviour are not well understood. The approach to research is generally empirical, and the properties of neural networks are determined and supported by experimentation rather than proof'.
- 'Controversies surrounding neural network models generally relate to the psychological reality or otherwise of the approach. In opposition to connectionism there are the advocates of symbolically-oriented, rule-based models of "classical cognitivism" (Putnam, 1960; Fodor, 1968). These controversies are however of less concern to those interested primarily in applications'.
- 'Within the applications field, potential controversy centres on the extent to which explicit structures, such as procedures, rules, semantic networks, frames and scripts, should be abandoned in favour of neural networks, which are essentially black boxes. The knowledge they contain is not open to scrutiny with the same clarity, for example, as rules in an expert system'.

5.2 Mathematical techniques

The approaches using mathematical techniques as the matching method in most cases are applied to the category of pattern matching problems which include geometric and graph matching. There are many techniques of this category and these will be examined in detail later. However, it can be said that their essence is that some form of mathematical techniques is applied to measure the degree of match between patterns. The overall merit of using these methods is that the result of matching is accurate. That is, because of the mathematical calculation, it is likely that the minimum measure of the difference between the patterns or parts of them can be achieved. The drawback is that the overhead of running the pattern matching process is high, because the patterns often have to be represented and matched mathematically which may involve different kinds of data structures and algorithms.

A number of different matching methods based on mathematical techniques are used by various researchers for a variety of application areas. Quite a number of them, used in the geometrical

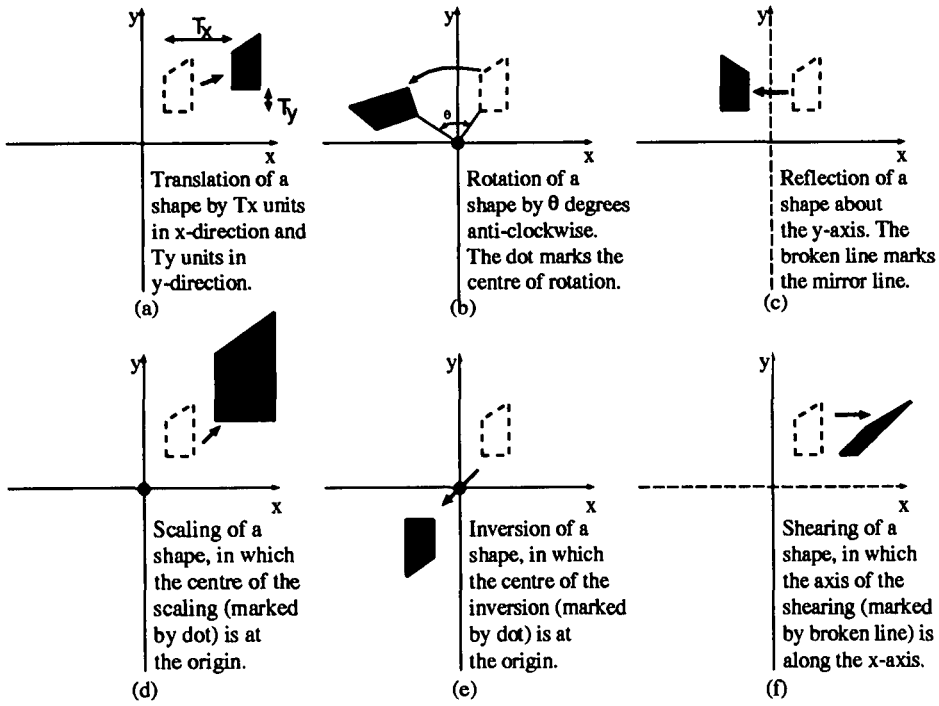


Figure 15 The six types of two-dimensional transformation (adapted from Rooney & Steadman, 1987)

matching problem, use geometrical transformations for normalising or processing the geometrical data into forms amenable to matching. Geometrical transformations refer to the transformation of the geometry of a two- or three-dimensional object from one situation (i.e. position, shape, size and orientation) into another. Whilst transformations of three dimensional objects can involve complex processes, those of two-dimensional ones are usually much simpler. Rooney and Steadman (1987) distinguish six types of two-dimensional linear transformation:

- *Translation*—a movement in the plane, from one position to another position, which preserves the shape and does not alter its orientation (see Figure 15a).
- *Rotation*—a movement in the plane, from one position and orientation to another position and orientation, which preserves the shape (see Figure 15b).
- *Reflection*—a movement in the plane, from one position and orientation to another position and orientation, which produces a congruent mirror image shape from the initial shape (see Figure 15c).
- *Scaling*—a movement in the plane which produces a (geometrically) similar shape from the initial shape but which preserves the orientation of the shape (see Figure 15d).
- *Inversion*—a movement in the plane, from one position and orientation to another position and orientation, which preserves the shape (see Figure 15e).
- *Shearing*—a movement in the plane which produces a shape having the same area as the initial shape (see Figure 15f).

The methods, for the geometrical matching problems, match and discriminate the shape of objects. To this end, the common parameter used is the "distances" between the objects which are matched. To determine whether the objects are similar or not, "tolerances" or "thresholds" of the distances between the objects are used. Those methods using "tolerances" determine the objects being similar if the distances between them are within a predefined tolerance. Those methods using "thresholds" determine the objects being similar if the distances between them are below a minimum predefined threshold. The methods used for these problems in the approaches reviewed in this paper are *coordinate distance measurement, hashing method* (Akutsu, 1995), *dynamic matching method* (Akutsu, 1995) and *cross correlation* (Cooper, 1989).

The matching methods for the graph matching problems are directed at determining similarities between graphs being matched. The problem in determining this kind of similarity is often called the graph isomorphism problem (Hopcroft & Tarjan, 1973). This problem is, given two graphs, to determine if there exists a one-to-one mapping of the vertices of one graph onto the vertices of another graph which preserves adjacency of the vertices. These methods are referred to as *graph-theoretic algorithms*.

Each of the above matching methods are discussed below.

5.2.1 Coordinate distance measurement

Coordinate distance measurement basically refers to the measurement of the distance between points, lines, curves, etc. in a coordinate system. Work by Schwartz and Sharir (1987), Schneider et al. (1989) and Karbacher (1990) uses this method. They also use some or all of the types of geometrical transformations to normalise the objects before starting the process of pattern matching.

The work by Schneider et al. (1989) measures the contour similarity between two geometric drawings of rotational mechanical parts as shown in Figure 16. In this figure, A and I are the outer and inner tolerance-contours, respectively, of a newly constructed part which is represented by a normalised contour K (not illustrated). K_n is the normalised contour of a stored part which will be used to match against the contour K. The normalisation includes translation, rotation and reflection of K and K_n. It is then defined that K is similar to K_n if and only if K_n can be translated in the x-direction such that A completely contains K_n and K_n completely contains I.

In Schwartz and Sharir (1987), this method is used to identify unoccluded and partially occluded objects. To this end, the boundary curve of an observed object O is matched against a candidate model M. For this purpose, the portion of the boundary curve of each object is represented as a sequence of evenly spaced points lying on the curve. The length of this sequence of M must be greater than or equal to that of O, otherwise the portion of O is too long to match M. Since the object might match to a subsequence starting anywhere along the model, the translation and rotation of the sequence of points representing the portion of the curve of the object are required to minimise the *least-squares distance* (Gellert et al. 1989) between the two sequences of points. The model for which this least-squares distance is minimised is then considered to be the best match for the observed object.

In Karbacher (1990), this method is used for classification and individual recognition of objects. This method is based on a comparison of scalar distances (minimum distance classifier). That is, it measures the average difference between two normalised pictures by measuring the distances between these pictures pixel by pixel. For pattern classification purposes, the distances between the objects and the learned objects (templates) are measured. To determine whether the objects belong to a particular class of objects where a particular template belongs to, a threshold is defined. Those objects whose distance with the template is below the threshold are classified into this class; otherwise, they are regarded as not belonging to the class.

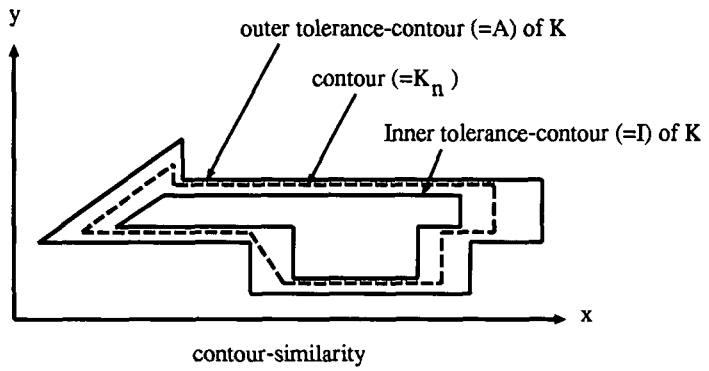


Figure 16 An illustration of the contour similarity between two geometric drawings (adapted from Schneider et al., 1989)

5.2.2 Hashing method

Hashing (also called *hash-addressing*) is a technique for providing fast and direct access to a specific stored record on the basis of a given value for some field (called *hash-field* or *hash-key*) in a database (Date, 1990). In this technique, each stored record/object has an address (called *hash-address*) computed as some function (called *hash-function*) of the hash field of the record. For example, for each record/object O, an integer value $f(O)$ is computed as the hash address. It follows that if O1 is identical to O2, then $f(O1) = f(O2)$. All records/objects are divided into groups so that records/objects in a group have the same $f(O)$. Thus, in object retrieval, if an input object (see I) is specified and an object identical to I is required to be searched, the system only checks the objects in the group associated with $f(I)$, making the search more efficient.

Work by Kalvin et al. (1986) and Akutsu (1995) uses this method. In Akutsu (1995), the method is used for chemical substructure search, i.e. given a pattern structure P and a text structure Q, the task is to find a part of a substructure (or enumerate all parts) of Q which matches P. Unlike pure hashing methods, which seem only to work for retrieving identical objects (Akutsu, 1995), the method by Akutsu works for similar object retrieval. Hence, it is called a *hashing based method*. This method is combined with Kabsch's (1976) method, a kind of least squares method (Gellert et al., 1989). In this method, P and Q are each represented as a sequence of three dimensional points; thus the substructure search is directed at finding the similarity between the sequence of points of P and that of the parts of Q. For this purpose, the hashing value (say $hs(X)$) of each sequence is computed. To determine each part of Q is similar to P, the inner product between $hs(q)$ (q is a subsequence of Q) with $hs(P)$ is calculated. It is to be noted that if the least squares distance between two sequences is very small, then $hs(P) \cdot hs(Q)$ is close to 1 (Akutsu, 1995). Thus, from this pattern matching process, those parts of Q whose inner products $hs(P) \cdot hs(q)$ are more than a predefined threshold, say $1 - d$ (d is a small positive fixed constant), are selected. For each of these selected parts, its least squares distance with P is computed using Kabsch's method and those parts whose least squares distances with P are less than the specified threshold are output as the most similar substructure to P.

In Kalvin et al. (1986), the method is used for geometrically hashing two dimensional model objects. It is able to recognise overlapping two dimensional objects, each being a collection of known model objects. It is based on the use of a synthetic attribute of an object called *footprint*. A footprint of an object O is any curve F derived from O in a rotationally and translationally invariant manner, and additionally in a manner that only depends on some relatively local portion (*window*) of the boundary of O around each given starting point s on O's boundary (Kalvin et al., 1986). In recognition of an overlapping two dimensional object, the boundary of the object is initially decomposed into subsections. The footprint of each boundary subsection (say S) is then formed and hashed. It is then matched against the footprints of candidate model objects. Those model objects whose footprints are sufficiently close to the footprint of S are retrieved from the database. Each of the candidate models retrieved is matched against S to select the nearest candidate.

5.2.3 Dynamic matching method

Dynamic matching is a method for matching sequences of three dimensional points which combines the dynamic programming technique with Kabsch's (1976) method. *Dynamic programming* is a set of parameterised recursive equations, which, in this case, express the cost of optimised longer segments of the program in terms of optimised shorter segments (Barr & Feigenbaum, 1982). The purpose of using the dynamic programming technique is to improve the efficiency of pattern matching, since it ignores redundant parts of the sequences to be matched. The work by Akutsu (1995) uses this combination method in searching for chemical substructure and common substructure. In the latter, given two sequences P and Q, the task is to find a part of P which matches a part of Q. In the dynamic matching method, Kabsch's method is initially used to compute the least squares distances between the subsequences of P and Q. The pairs of the subsequences of P and Q whose least squares distances take minimal values are selected and formed as a sequence of such pairs (e.g. S). The dynamic programming technique is then applied to

S. In this technique, an iteration process is used to find the best pair from *S*. More details of this technique can be found in Akutsu (1995).

5.2.4 Cross correlation method

This refers to a method of template matching which measures the degree of match between the template and the picture. This measurement by correlation involves the computation of the sum of the product between the pixel value of the template and that of the picture (Hall & Matias, 1993). This can be calculated by shifting the template over all possible locations in the picture and computing a measure of match at each location. Work by Cooper (1989) and Hall and Matias (1993) uses this method. In Cooper (1989), this method is used for the recognition of two dimensional rigid objects of a fixed orientation. In Hall and Matias (1993), it is used for identification of virus images. More specifically, in Hall and Matias (1993) rotation, scale and translation (RST) invariant template matching is proposed. With this kind of template matching the virus in any position, orientation and size can be identified.

5.2.5 Graph-theoretic algorithms

Graph-theoretic algorithms refer to the algorithms for finding graph isomorphisms. There are different types of graph isomorphisms. Harary (1969) and Berger (1976), for example, give three types of pure graph isomorphisms:

(1) *Graph isomorphism*; given two graphs (*V1, E1*) and (*V2, E2*), find a one-to-one mapping (an isomorphism) *f* between *V1* and *V2* such that for $v1, v2 \in V1$, $V2$, $f(v1) = v2$ and for each edge of *E1* connecting any pair of nodes *v1* and $v1' \in V1$, there is an edge of *E2* connecting $f(v1)$ and $f(v1')$.
(2) *Subgraph isomorphism*; find isomorphisms between a graph and *subgraphs* of another graph.
(3) *"Double" subgraph isomorphisms*; find all isomorphisms between *subgraphs* of a graph and *subgraphs* of another graph.

The examples of these algorithms given by Ballard and Brown (1982) are *matching metrics* (the measure of "goodness" quantified for a correspondence between graphs), *backtrack search* (a type of potentially exhaustive search organised in stages) and *association graph techniques* (an auxiliary data structure produced from two graphs to be matched). For more details on these algorithms, see Ballard and Brown (1982). Work by Sussenguth (1965), McGregor (1982), Giretti et al. (1994), McLaren (1994), one approach (using graph representation) in Voß et al. (1994) and the first and second approaches in Akutsu (1992) uses these algorithms.

The overall merit and drawback of using mathematical techniques as the matching methods have been given. The advantages in more detail are:

- Since the result of matching objects using mathematical techniques is likely to be accurate, those methods applied in the geometrical matching problems will also have this property. That is, the high accuracy or maximum preservation of the complete information regarding the matching geometric pattern can be achieved because the comparison between the patterns to be matched are carried out mathematically.
- Matching graph representations of objects using graph-theoretic algorithms is an efficient pattern matching. This is because a graph may represent an abstraction of an object, thus matching graphs implies matching the abstraction (in a particular aspect) of objects, without having to match the low level details of the objects.

The disadvantages of using these methods are:

- They take high computation time since they involve a relatively large number of mathematical expressions to represent objects and use different mathematical operations to do the pattern matching process.

- They take relatively large memory space of the database to store the diverse types of data structures to represent the mathematical expressions of objects and for the purpose of mathematical operations.

5.3 Symbolic pattern matching

Symbolic pattern matching specifically refers to matching semantic/symbolic patterns of objects. In the matching process, the patterns, representing the semantic/symbolic attributes of objects, are compared to see if one is similar to another. The overall merit of using this method is that in most cases these patterns are represented in simple data structures, usually in list expressions. With such a data structure, the matching process is easy to carry out because a datum or stored pattern in a database does not have to be accessed by its name, for example, the name of a variable, but by specifying its form (or meaning) instead. This is what Barr and Feigenbaum (1982) call *content addressing*. They give an example: 'a pattern, such as ((*FATHER-OF ?X*) *?Y*), is used as a kind of sketch of the datum being sought in the database and "matches" against stored items like ((*FATHER-OF MARY*) (*FLIES KITES*)). Thus, the name or address of the stored information need not be remembered. This makes the matching process much easier to accomplish because it is not necessary to specify which calls what, or what access path is taken to get to an item in the database'.

The overall drawback of using symbolic pattern matching is that because the patterns usually represent the semantic/symbolic attributes of objects and not the patterns of the objects themselves, it does not support the cognitive processes of recognising the patterns of the objects. Rather, it focuses on retrieving the attributes of the objects. This is line with what Peschl (1990) points out, in that 'the symbolic approach of orthodox artificial intelligence does not describe or investigate the process(es) of cognition, but rather describes and investigates the process of how to handle and manipulate descriptions of a reality. This means the approach commits a categorical error, because it mixes up the level of cognitive processes with the level of how these processes can be interpreted (in natural language, i.e. descriptions of objects)'.

As has been discussed in section 3, semantic/symbolic patterns take the form of characters (text and numbers) or symbols (such as diagrams, icons, sketches, etc.) and can represent the semantic/symbolic attributes of objects (such as the names, functions, sizes, colours and other features). The symbolic pattern matching method matches these attributes of objects. Symbolic pattern matching of the attributes of objects is the most commonly found matching method in the area of case based reasoning (CBR); as pointed out by Bareiss (1991) and Kolodner (1993), feature vectors (the semantic attributes of objects) are the standard representation in CBR. In features vectors the attributes and their values of an object are represented using text and numbers in attribute-value pairs.

Work by Dave et al. (1994), Giretti et al. (1994), Gross et al. (1994) and several approaches in Voß et al. (1994) uses this method. In Gross et al. (1994), symbolic pattern matching takes place in the sketch program where an input diagram is matched with templates in a catalogue of diagrams. Each diagram has its symbolic representation in the sketch program. For example, to identify spatial relations between the elements of a diagram the program uses a small list of predicates (such as above, contains and near). Thus, in matching the input diagram with the templates the symbolic representations of these diagrams are used. In Dave et al. (1994), association list data structures which represent the symbolic information of building design cases and the new site description of the new design problem are matched to find a case for case adaptation or cases for case combination. In Giretti et al. (1994), a part of the representation of a set of theory elements (i.e. spatial adjacencies and the relations between the access and other components of the different stored scenes) uses a set of classification rules. This rule set is used to analyse a given scene, that is, matching this rule set with the rule set representing the scene. This will produce the classification of the scene according to the theory which the rule set refers to.

In Voß et al. (1994), because of the large set of different objects contained in a building design, the layout retrieval is based on particular views of the building. Such views are called layout

fragments. An example of a layout fragment is the fresh air system in a particular room. Two of the retrieval approaches in their work use feature vectors; thus the matching method used is symbolic pattern matching. For layout fragment retrieval, two feature vectors are compared by computing their distance, i.e. the weighted sum of differences in the features.

This method has also been applied in the class geometric pattern. For example, the third approach in Akutsu (1992) (i.e. the approach using graphs in two dimensional space) and Reihani (1994) use this method. More specifically, as has been mentioned in section 3.1, in Akutsu (1992) the graph representation is transformed into a string representation, thus the geometrical matching problem becomes a string matching problem. In the string representation, a string may represent a vertex, e.g. symbolised *str*(*Pi*), or the whole graph, e.g. *str*(*G*). For the vertex, the string is made up of the sequence of the following: the angle between the two edges intersecting in the vertex, the length of one of these sides and the number of adjacent vertices, which all are represented inside a string (" "). *str*(*G*) consists of more than one *str*(*Pi*) and is defined as *str*(*G*) = *str*(*P1*)*str*(*P2*) . . . *str*(*Pn*). For matching two given graphs *G* and *G'*, the string matching is carried out by defining if *str*(*G'*) is a substring of a concatenation of *str*(*G*) with itself, i.e. *str*(*G*)*str*(*G*). Note that the lengths of *str*(*G*) and *str*(*G'*) are the same (Akutsu, 1992). Thus, this kind of matching is symbolic pattern matching because it matches two string of two graphs.

In Reihani (1994), a pattern recognition system for the processing hierarchy for two dimensional image structures is proposed. This system is based on a structural approach that couples symbolic and numerical methods to manipulate symbolic descriptions, namely spatial occurrences of lines, arcs, circular regions, etc., of hand-written characters. These symbolic descriptions are defined in a hierarchical manner. Hierarchical image structures have important roles in the pattern recognition area (Reihani, 1994), since they provide the decompositions of the structure into significant features, such as arcs and lines, with which matching the structures is easier to carry out. That is, in matching these kinds of structures only these significant features and the relations of these features are considered. Unknown objects in an image are matched with the structures of objects previously stored in a database by using symbolic object representations to find the most similar structure.

Symbolic pattern matching is also found in rule-based systems, which are also often called production systems. A production system uses sets of rules, called *production rules* (Waterman, 1986), to represent and solve problems. It is one of the most common ways of representing knowledge in expert systems. An expert system could be defined as a system which aims to exhibit a behaviour which can be considered intelligent, by tackling a specialised problem domain and demonstrating expertise at solving problems in this domain (Duffy, 1989).

A production rule (or a rule, for short) is an ordered pair of patterns with a left-hand side and a right-hand side (Ballard & Brown, 1982). A pattern may involve only database primitives but usually will have variables and special forms as subpatterns which are matched against a set of facts in the database by an interpreter. Thus, a production system has three basic components: a database, a set of rules and an interpreter for the rules. Symbolic pattern matching takes place

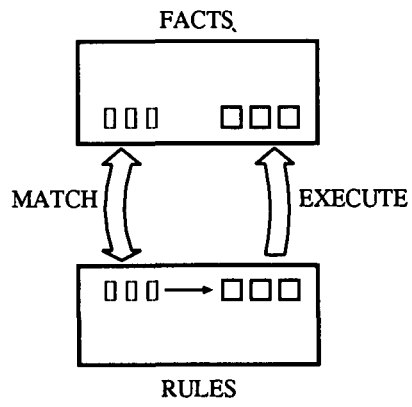


Figure 17 The rule interpreter cycles through a match-execute sequence (adapted from Waterman, 1986)

when matching the left-hand side with the facts in the database. Waterman (1986) gives a diagram of the rule interpreter cycles which consist of two processes: matching the left-hand side with the facts and executing the rule (see Figure 17).

To show more clearly where pattern matching takes place in a production system, consider an example given by Ballard and Brown (1982). Suppose examples of facts stored in the database are as follows:

(ABOVE (Region 5) (Region 10))
(SIZE (Region 5) 300)
(SKY (Region 5))
(TOP (Region 5) 255)

Applying the following rule, for example, to the database which includes the above facts, region 5 can be inferred to be sky:

(TOP (Region X) (GreaterThan 200))
→
(SKY (Region X))

The left-hand side of the rule matches the database facts and this causes (SKY (Region 5)) to be added to the database. In this example, the kind of matching that the interpreter must do is as follows: (1) the primitive TOP in the database fact matches the same symbol in the rule; (2) (Region X) matched (Region 5) and X is bound to 5 as a side effect; and (3) (GreaterThan 200) matches 255.

Work by Alvisi and Odorico (1988) and Meeran et al. (1993) uses production systems. In Alvisi and Odorico (1988) an OPS5 expert system has been developed for pattern recognition in geometric figures which are represented as tri-connected planar graphs. More specifically, this system is intended to recognise an arbitrary input pattern within a geometric figure by matching this input pattern with the parts of the geometric figure. Sets of rules in this system are used to recognise the feature elements of the planar graph representation (i.e. vertices, edges and cycles) of both the input pattern and the geometric figure. They are then used to explore the correspondence between these feature elements of the input pattern and the geometric figure to find the matching subpatterns of the geometric figure.

In Meeran et al. (1993) the approach is used for the process of feature recognition of two-dimensional CAD drawings. For the process of recognition, the drawing is initially encoded into a data exchange format, which is termed a *neutral file* format, the examples of which include IGES, STEP and DXF. In this work, the DXF file format was chosen. In this file format a geometric entity in the drawing (such as line, circle or arc) is represented by a name (a character string identifying its nature) and by defining numerical parameters (such as the line type and coordinates). Thus, a file entry for the line, for example, is represented as: *LINE- line type, x and y coordinates of the end points*. The approach is implemented in PROLOG, and thus before the process of recognition the DXF file is translated into a PROLOG database. The PROLOG facts in this database serve as the initial description of the drawing for the feature recognition process. For this process, a set of rules representing an input pattern are matched with the facts to recognise the pattern.

Symbolic pattern matching is also found in logic-based systems. For example, a logic-based system called Chaus (**C**hemical knowledge **a**cquisition and **u**tilisation **s**ystem) (Akutsu et al., 1991) uses this method for chemical substructure matching. This system is designed as a general purpose tool for building expert systems in the domain of organic chemistry. Operations concerning chemical structures are carried out through a set of built-in predicates called *procedural type atom* (PTA in short). For substructure matching, for example, two PTAs: (1) subpat (PS,CS) and (2) subpatmatch (PS,CS,MAP) are implemented. PS is a pattern structure, CS is a complete chemical structure and MAP is a list of pair of nodes which shows the correspondence between nodes in PS and nodes in CS. When the search for the first correspondence succeeds, backtracking occurs to search for another correspondence. Figure 18 gives an example of this substructure matching.

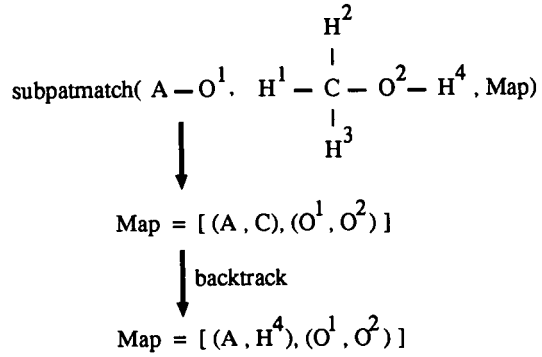


Figure 18 Function of the subpatmatch predicate. The superscript of a node is an index helping to distinguish the different nodes of the same atom name (adapted from Akutsu et al., 1987)

The overall merit and drawbacks of using symbolic pattern matching have been given in the opening paragraphs of this subsection. There are further advantages and disadvantages of using this method. The advantages are:

- Since this method can be viewed as content addressing (Barr & Feigenbaum, 1982) in that to access a stored pattern in a database the form of the pattern would be sufficient to use, it is possible for the method to have "best match" capability: 'instead of matching exactly, or failing, the method would do the best it could and return information on the points it could not find a match on' (Barr & Feigenbaum, 1982). It follows that the method could do *approximate* and *partial* matching which could be intended in particular application domains to extract particular information from an object.
- As a side effect of the first advantage, semantic/symbolic information extracted from applying this method is relatively sufficient for particular tasks, such as diagnoses and design. In design, for example, this information may be utilised as the initial semantic/symbolic attributes of a new design as in Dave et al. (1994), Giretti et al. (1994), Gross et al. (1994) and Voß et al. (1994).

The disadvantages of using symbolic pattern matching are:

- "An image (e.g. a drawing) can paint more than a thousand words". In other words, a large number of words may be required to explain an image and is likely to be an inadequate and imprecise representation medium. Consequently, this method is less efficient for image recognition since a large number of symbolic pattern matching processes are required to exhaustively recognise the image.
- The pattern matching process in this method often involves matching a set of input patterns with the stored patterns, such as that in rule-based systems (i.e. matching a set of rules against the facts in a database). The interpreter in these systems might only examine a particular subset of the facts for each rule, retrieving for matching only elements indexed as likely to match that rule, but it would still make as many passes through the facts as the number of rules (Jackson, 1986). This makes this method relatively inefficient.

6 Summary table

This review has summarized pattern matching approaches on the basis of three key issues: pattern classes, similarity types and matching methods. For each issue variations and commonalities of ideas between the approaches have been identified and summarised. It is worth noting that applicability of each of the matching methods depends upon the types of problems in the application domains, including what are the representations of the objects to be matched. The

Table 4 A summary of pattern matching approaches

Pattern Matching Approaches				
Author(s)	Pattern Class	Similarity Type	Matching Method	Application
Akutsu, 1992 (the first approach)	GP	R, OS	MT (GTA)	mechanical CAD systems
Akutsu, 1992 (the second approach)	GP	R, PI	MT (GTA)	mechanical CAD systems
Akutsu, 1992 (the third approach)	GP	R, PI	SPM	mechanical CAD systems
Akutsu, 1994	GP	R, PS	MT (HM, DMM)	chemical analysis/ identification
Akutsu et al	TR	R, PI	SPM	chemical analysis/ identification
Alvisi & Odorico	TR	R, PI	SPM	pattern analysis
Cooper	DP	R, OS	MT (CC)	image processing
Coyne & Postmus	DP	R, OS	NN	spatial design
Dave et al	S/SP	R, OS	SPM	spatial design
Giretti et al	TR, S/SP	R, OS, PI	MT (GTA), SPM	spatial design
Gross et al	S/SP	R, OS	SPM	spatial design
Hall & Matias	DP	R, OS	MT (CC)	image processing
Kalvin et al	GP	R, PS	MT (HM)	manufacturing
Karbacher	DP	R, OS	MT (CDM)	image processing
Korn	DP	R, OS	NN	pattern analysis
McGregor	TR	R, PI	MT (GTA)	chemical analysis/ identification
McLaren	GP	R, OS	MT (GTA)	underwater navigation systems
Meeran et al	GP	R, PI	SPM	manufacturing
Peters	GP	R, PS	NN	mechanical CAD systems
Reihani	GP	R, OS	SPM	handwritten character recognition and scene analysis
Schneider et al	GP	R, OS	MT (CDM)	mechanical CAD systems
Schwartz & Sharir	GP	R, PS	MT (CDM)	manufacturing
Sussenguth	TR	R, I, PI	MT (GTA)	chemical analysis/ identification
Teo & Sim	DP	R, PS	NN	handwritten character recognition and scene analysis
Voβ et al	TR, DP, S/SP	R, OS, PI	MT (GTA), SPM	spatial design

Legend :
GP = Geometric Pattern
TR = Topological Relations
DP = Distributed Pattern
S/SP = Semantic/Symbolic Pattern
R = Resemblances
OS = Overall Similarity
I = Identity
PS = Part Similarity
PI = Part Identity
NN = Neural Networks
MT = Mathematical Techniques
SPM = Symbolic Pattern Matching
CDM = Coordinate Distance Measurement
HM = Hashing Method
DMM = Dynamic Matching Method
CC = Cross Correlation
GTA = Graph-Theoretic Algorithms

representations of objects have been classified into four classes of patterns as given in section 3. Furthermore, concepts of similarity between objects proposed by different authors have also been reviewed. Of these concepts, Smith's (1989) concept has been adopted to be the concept for defining the similarity types used in the approaches since it gives the more concrete concept of the classification of similarity types. Additionally, advantages and disadvantages of each matching method have been identified. This identification may be used as important considerations when attempting to adopt the methods for particular applications. Finally, a summary of the three issues and applications of the approaches is given in Table 4.

7 Concluding remarks

Pattern matching is a key element in many applications and its main goal is to emulate the human ability to recognise and distinguish patterns of various types and their combinations. The technique was originally applied to pattern analysis and image processing as a means of pattern recognition. It is now used extensively in robotic vision and artificial intelligence.

It is also an essential component of case based reasoning systems, a main application of which is classification and retrieval of past design solutions. The symbolic pattern matching method has mainly been used for this purpose, but to support graphic information retrieval a technique which can represent geometric objects directly needs to be developed.

It is thought that pattern matching in various forms and implementations will be used in ever widening application areas in the future and new techniques will evolve in pace with the growth of its importance in technology.

References

Akutsu, T, 1992. "Algorithms for determining the geometrical congruity in two and three dimensions" In: Y Inagaki, K Iwama, T Nishizeki, T Ibaraki and M Yamashita (eds) 1992. *Algorithms and Computation* 279–288. Springer-Verlag.

Akutsu, T, 1995. "Efficient and robust three dimensional pattern matching algorithms using hashing and dynamic programming techniques" In: *Proceedings of the Hawaii International Conference on System Sciences* **5** 225–234. IEEE.

Akutsu, T, Suzuki, E and Ohsuga, S, 1991. "Logic-based approach to expert systems in chemistry" *Knowledge-Based Systems* **4** (2) 103–116.

Alvisi, L and Odorico, R, 1988. "A rule based approach for pattern recognition in planar geometric figures" *Computer Physics Communications* **51** (3) 443–450.

Ballard, DH and Brown, CM, 1982. *Computer Vision* Prentice-Hall.

Bareiss, R (ed) 1991. *Proceedings: Case-Based Reasoning Workshop 1991* Morgan Kaufmann.

Barr, A and Feigenbaum, EA (eds) 1982. *The Handbook of Artificial Intelligence vol 2* Pitman.

Barsalou, LW, 1989. "Intraconcept similarity and its implications for interconcept similarity" In: S Vosniadou and A Ortony (eds) *Similarity and Analogical Reasoning* Chapter 3, 76–121. Cambridge University Press.

Berge, C, 1976. *Graphs and Hypergraphs* Elsevier.

Clark, A, 1989. *Microcognitron: Philosophy, Cognitive Science and Parallel Distributed Processing* MIT Press.

Cohen, PR and Feigenbaum, EA (eds) 1982. *The Handbook of Artificial Intelligence* **3**. Pitman.

Cooper, MC, 1989. "Formal hierarchical object models for fast template matching" *The Computer J* **32** (4) 351–361.

Coyne, RD and Newton, S, 1989. "A tutorial on neural networks and expert systems for design" In: JS Gero and F Sudweeks (eds) *Expert Systems in Engineering, Architecture & Construction* 321–337. Sydney.

Coyne, RD, Newton, S and Sudweeks, F. 1989. "Modelling the emergence of schemas in design reasoning" In: *Modelling Creativity and Knowledge Based Creative Design* 173–205. Design Computing Unit, Department of Architectural and Design Science, University of Sydney.

Coyne, RD and Postmus, AG, 1990. "Spatial applications of neural networks in computer-aided design" *Artificial Intelligence in Engineering* **5** (1) 9–22.

Date, CJ, 1990. *An Introduction to Database Systems (5th ed)* Addison-Wesley.

Dave, B, Schmitt, G, Faltings, B and Smith, L, 1994. "Case based designs in architecture" In: JS Gero and F Sudweeks (eds) *Artificial Intelligence in Design '94* 145–162. Kluwer.

Domeshek, E and Kolodner, J, 1991. "Toward a case-based aid for conceptual design" *International J Expert Systems* **4** (2) 201–220.

Domeshek, E and Kolodner, J, 1992. "A case-based design aid for architecture" In: JS Gero (ed) *Artificial Intelligence in Design '92* 497–516. Kluwer.

Duffy, AHB, 1989. *Expert Systems in Engineering* Course Notes, CAD Centre, University of Strathclyde, Glasgow, Scotland.

Duffy, AHB and Kerr, SM, 1993. "Customised perspectives of past designs from automated group rationalitations" *Artificial Intelligence in Engineering* **8**(3) 182–200.

Fodor, J, 1968. "The appeal to tacit knowledge in psychological explanation" *J Philosophy* **65** 627–640.

Fu, KS (ed) 1976. *Digital Pattern Recognition* In *Communication and Cybernetics* Springer-Verlag.

Fukushima, K, 1988. "Neocognitron: a hierarchical neural network capable of visual pattern recognition" *Neural Networks* **1** 119–130.

Fukushima, K and Miyake, S, 1982. "Neocognitron: a new algorithm for pattern recognition tolerant of deformations and shifts in position" *Pattern Recognition* **15** (6) 455–469.

Gellert, W, Gottwald, S, Hellwich, M, Kastner, H and Kustner, H, 1989. *The VNR Concise Encyclopedia of Mathematics (2nd ed)* Van Nostrand Reinhold.

Giretti, A, Spalazzi, L and Lemma, M, 1994. "A.S.A: an interactive approach to architectural design" In: JS Gero and F Sudweeks (eds) *Artificial Intelligence in Design '94* 93–108. Kluwer.

Gross, M, Zimring, C and Do, E, 1994. "Using diagrams to access a case base of architectural designs" In: JS Gero and F Sudweeks (eds) *Artificial Intelligence in Design '94* 129–144. Kluwer.

Hall, G and Matias, A, 1993. "Rotation, scale and translation invariant template matching on a transputer network" *Microprocessors and Microsystems* **17** (6) 333–340.

Harary, F, 1969. *Graph Theory* Addison-Wesley.

Hopcroft, JE and Tarjan, RE, 1973. "A V log V algorithm for isomorphism of tri-connected planar graphs" *J Computer and System Sciences* **7** 323–331.

Jackson, P, 1986. *Introduction to Expert Systems* Addison-Wesley.

Kabsch, W, 1976. "A solution for the best rotation to relate two sets of vectors" *Acta Crystallography* **A32** 922–923.

Kalvin A, Schonberg, E, Schwartz, JT and Sharir, M, 1986. "Two-dimensional, model-based, boundary matching using footprints" *International Journal of Robotics Research* **5** (4) 38–55.

Karbacher, S, 1990. "Associative object recognition by hierarchic template matching" *Optical Engineering* **29** (12) 1449–1457.

Kolodner, JL, 1993. *Case-Based Reasoning* Morgan Kaufmann.

Korn, GA, 1990. "Interactive statistical experiments with template-matching neural networks" *IEEE Trans Systems, Man, and Cybernetics* **20** (5) 1146–1152, September/October.

Maher, ML and Zhao, F, 1987. "Using experience to plan the synthesis of new designs" In: JS Gero (ed) *Expert Systems in Computer-Aided Design* 349–373. Elsevier.

McClelland, JL and Rumelhart, DE, 1987. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: 2: Psychological and Biological Models* MIT Press.

McGregor, JJ, 1982. "Backtrack search algorithms and maximal common subgraph problem" *Software-Practice and Experience* **12** 23–34.

McLaren, N, 1994. "A sonar-based navigation system for underwater vehicles" *PhD thesis*, Department of Ship and Marine Technology, University of Strathclyde, Glasgow, Scotland.

Meeran S, Pratt, MJ and Key, JM, 1993. "The use of PROLOG in the automatic recognition of manufacturing features from 2-D drawings" *Engineering Applications of Artificial Intelligence* **6** (5) 409–423.

Peschl, MF, 1990. "Neural networks and symbolic computation in cognitive modelling" In: F Gardin and G Mauri (eds) *Computational Intelligence, II* 229–235. Elsevier.

Peters, TJ, 1992. "Encoding mechanical design features for recognition via neural nets" *Research in Engineering Design* **4** 67–74.

Preparata, FP and Shamos, MI, 1985. *Computational Geometry, An Introduction* Texts and Monographs in Computer Science. Springer-Verlag.

Putnam, H, 1960. "Minds and machines" In: S Hook (ed) *Dimensions of Mind* New York University Press.

Reihani, K, 1994. "Processing hierarchy for 2-D image structures" *Computing Systems in Engineering* **5** (1) 41–54.

Rips, LJ, 1989. "Similarity, typicality, and categorization" In: S Vosniadou and A Ortony (eds) *Similarity and Analogical Reasoning* Chapter 1, 21–59. Cambridge University Press.

Rooney, J and Steadman, P (eds) 1987. *Principles of Computer-aided Design* Pitman/Open University.

Rumelhart, DE, 1989. "Toward a microstructural account of human reasoning" In: S Vosniadou and A Ortony (eds) *Similarity and Analogical Reasoning* Chapter 10, 298–312. Cambridge: Cambridge University Press.

Schneider, R, Kriegel, H-P, Seeger B and Heep, S, 1989. "Geometry-based similarity retrieval of rotational parts" *Data and Knowledge Systems for Manufacturing and Engineering* 150–160, October.

Schwartz, JT and Sharir, M, 1987. "Identification of partially obscured objects in two and three dimensions by matching noisy characteristic curves" *International Journal of Robotic Research* **6** (2) 29–44.

Smith, LB, 1989. "From global similarities to kinds of similarities: the construction of dimensions in development" In: S Vosniadou and A Ortony (eds) *Similarity and Analogical Reasoning* Chapter 5, 146–178. Cambridge University Press.

Smith, EE and Osherson, DN, 1989. "Similarity and decision making" In: S Vosniadou and A Ortony (eds) *Similarity and Analogical Reasoning* Chapter 2, 60–75. Cambridge University Press.

Sussenguth Jr, EH, 1965. "A graph-theoretic algorithm for matching chemical structures" *J Chemical Documentation* **5** 35–43.

Teo, MY and Sim, SK, 1995. "Training the neocognitron network using design of experiments" *Artificial Intelligence in Engineering* **9** (2) 85–94.

Vosniadou, S and Ortony, A, 1989. *Similarity and Analogical Reasoning* Cambridge University Press.

Voß, A, Coulon, C-H, Grather, W, Linowski, B, Schaaf, J, Bartsch-Sporl, B, Borner, K, Tammer, EC, Durschke, H and Knauff, M, 1994. "Retrieval of similar layouts—about a very hybrid approach in FABEL" In: JS Gero and F Sudweeks (eds) *Artificial Intelligence in Design '94* 625–640. Kluwer.

Waterman, DA, 1986. *A Guide to Expert Systems* Addison-Wesley.