Neural Network For Feature Recognition in Automated Process Planning

Herry

Abstract  The objective of this project is to study on the integration of Neural Network in Computer Aided Process Planning especially in feature recognition application. An introduction about process planning will be covered in brief. Following this, the concept of feature shall be discussed as an integral part in CAPP. The next section presents one of the feature recognition methods developed by Nezis and Vosniakos. The strategy their approach is to use a Neural Network system to recognize feature from a sub-graph. The sub-graph is decomposed from an Attributed Adjacency Graph of a solid model. In the last section of this report, a radial basis function network is proposed to substitute the current Neural Network type. Although, the feature recognition method presented in this report is not a new, it is still worthwhile to discuss as there are not many publications exploited this method. In addition, the advantages of the proposed NN over the current one seem to give a new atmosphere in this field. Thus, this can be a starting point and stimulator for further researches.

Key Words: CAPP, Neural Network, CAD, CAM, Boundary representation.

1. Introduction And Background

In manufacturing of any product, performing of any service, or carrying out of any individual activity, the person or a mechanism doing it follows a certain, usually, predetermined sequence of steps. This is just as true for the production of any product. The steps predetermined to make the product or service more efficiently are commonly called operations. Thus, a sequence of operations and related activities is called a process. And the work involved in analyzing the product or service, and specifying the operations and equipment required, is called process planning [1].

Making plans for a mechanical part involves the preparation of a plan that outlines the processing route, operations, machine tools, fixture, and tools required to produce the part at the most minimum cost without suffering its quality. All together, these planning tasks may consume more human effort and time than the actual design itself. Therefore, to fully realize the benefit of computer-aided in design (CAD) and manufacturing (CAM), computer support for process planning activities should also be available as a mean of a sophisticate and efficient system. From this point, people are encouraged to automate process planning [2].

1.1 Computer aided process planning

The idea of using computer to help producing process plans was first discussed by Niebel [3] in 1965. Since then, many efforts have been noticed in developing process planning systems. At that time, due to the limitation in machinery technology and current computing methodology, early Computer-Aided Process Planning (CAPP) systems generally followed the variant planning approach. This approach involves storing and retrieving standard sets of plans for parts that are usually classified into family class on the basis of their geometric shape [4]. As the result of this classification system, a significant amount of reducing in production lead time, work-in-process, labor, tooling, rework and scrap materials, setup time, and paper works [4]. This system is known as Group Technology scheme proposed by Mitrofanov [5] in 1966.
The next generation after variant approach is the generative system. In this method, the computer system attempts to synthesize the process plan directly [6]. It develops new plan for each part based on the part’s features and attributes. In the early year of this system, Trusky [7] and Wysk [8] developed generative systems for use in gear cutting firm and milling-hole creation application, respectively. However, the generative CAPP is too complex and difficult to develop even in the present day. Therefore, the true generative system, which automated the processes in a whole, is still a goal in the future [9].

1.2 Feature-based system

The definition of product geometry is needed as the input for process planning like many major computer-aided engineering (CAE) applications. However, the geometric information alone is not adequate to automate process planning. Therefore, a human planner is needed to interpret the output from CAD department as the input for CAM department. The planner must also pose a reasonable knowledge about most aspects in design and manufacturing [10].

In order to obtain an automated process planning (APP) system with less intervention from human, it needs some more data on the shapes, such as knowledge of the characteristic shape produced by the various processes, and their dimensions, locations, tolerances as well as surface finish. Moreover, it should also satisfy some questions such as; whether it is allowable to inspect, possibility of designing fixture or tool accessibility [10].

A collection of information needed as the input in order to make more flexible linkage between CAD and CAM in a fashion that support the use of manufacturing information during product design, and conversely design information during manufacturing planning and the actual manufacture have encouraged people to develop feature-based systems [10].

By itself, the definition of feature can be safely though as building blocks for product definition or for geometric reasoning. Consequently, features are related to some physical, that is, the geometric aspects of a part or assembly. According to Shah [10], characteristics of a feature can be defined as follow;

• A feature is a physical constituent (element) of a part
• A feature is ‘mappable’ to a generic shape
• A feature has engineering significance
• A feature has predictable properties.

These variations of feature definition are defined because there are semantically significant and distinct entities in one or more engineering viewpoints.

In conclusion, the advantages of feature-based in CAE especially in APP to bridge between CAM and CAD can be summarized as follow [10];

• Features are convenient input data representation for the planning system.
• Process knowledge needed for planning can be associated with feature classes.
• Geometric reasoning for various tasks of process plans are facilitated by features.

1.3 Automatic feature recognition

A feature-based CAPP should have a facility for creating feature models. Many approaches have been formulated to create feature-based models. In general, those approaches can be classified into two categories; based on whether geometry is created from the feature or whether features are extracted from geometry [10].

In the former, the designer intentions are captured at the beginning and feature representations are created, next they are stored during the design process. This approach is
called design by feature (DBF), which can be classified into two methods; procedural and declarative [10].

In the latter approach, the process starts from a pure geometric model (CAD data), and then a computer program processes the resulting model to automatically find features [10]. This approach is called automatic feature recognition (AFR). The first attempt for recognizing CAD model was in 1980 when Kyprianou developed a method for classifying between depression and protrusion feature [11]. Since then, there have been many attempts at developing feature recognition system. The techniques employed include syntactic pattern based [10,12], rule based [10,13], graph based [10,14] and Neural Network (NN) based [15,16,17,18,19] which extracts the solid boundary representation (B-rep) from a model. The other techniques such as CSG tree matching, boundary matching and DOF classification are extracted from volume base representation. The further information for the volume-based techniques can be found in [10].

1.4 Previous researches in feature recognition

Syntactic pattern methods use sequences of geometric elements to describe 2D feature. It recognizes simple primitive feature such as slots, steps, blind slots, blind steps and pockets [2]. However, Nezis and Vosniakos [17] summarized that this method is difficult to work in 3D,
because of the consistency needed in creating part codes out of geometric elements connected in space.

In the rule-based methods or expert system methods, number of rules is used to describe a feature. In this method, a set of heuristic rules is constructed to describe the operational definition for a particular feature [13]. For example, one want to define a set of heuristic rules to define, say, a rectangular pocket. So the set of the rule can may follows;

- A pocket is composed of 5 face (f1,f2,f3,f4,f5)
- Face f1 should be orthogonal to the rest
- Each face exclude f1 is adjacent to three faces
- There are two parallel face

In the graph-based methods, a class of features is first modeled by a graph structure, which describe the required topological and geometric constraints of for identifying a feature. In this method, there are two symbols; the nodes represent part faces and arches denote adjacencies. If two faces meet at an edge, then there is an arc joining the two nodes. In turn, if there are two parallel faces, then the two nodes will not be connected. A variation of this method is to attach geometry information or any attribute on the arcs proposed by Joshi [14] which called The Attributed Adjacency Graph (AAG). The attribute serves as a flag indicating the convexity or concavity of two faces. This flag can be ‘0’ for concave adjacency or ‘1’ for the other [10]. The illustration can be seen in figure 7.

The other method used in feature recognition is Neural Network-based (NN-based) methods. This method was initiated when people have found that all the three methods above have some limitations in recognizing solid model that have a complex and compound feature. One of the limitations of those methods is that they do not have the ability to learn or dynamically improve their behavior [19]. The other is that those conventional methods need lots of time to recognize a compound and complex model [20].

By itself, NNs are a class of computing systems that uses a highly parallel architecture that have a natural tendency for storing experiential knowledge and making it available for use [21]. NN works in the same way as biological nervous systems do. They can analyze large quantities of data to establish patterns and characteristics in situation where the rules are hardly available or, in many cases, are in fuzzy or noise conditions. A typical NN system usually consists of 3 main layers; an input layer as the receptor, a hidden layer and an output layer. The input and output layer consist of many nodes which called neurons. Simple NN architecture is shown in figure 3.

The work done by Prabhakar and Henderson [15] in 1990 seems to be the first attempt in applying NN for feature recognition [17]. In their work, they used 5 layers NN with the input to the net is Adjacency Matrix. However, their system seemed to only work for basic and simple feature and it would make a mistake in handling more complex feature or interacting feature such as slot and pocket, complex pocket or any non standard type features. Although this work can only recognize a limit number of simple feature, Prabhakar and Henderson had proofed that NN-based can be used in feature recognition [20]. In addition, they also proofed that NNs are easy to train and recognition speed much faster and better than other method such as expert system based or rule based which needs some relevant knowledge in implementing the design.
Since then, various NN methods in feature recognition have been proposed. One of NN-based recognition method was proposed by Hwang [19]. In this method, the 3D object model is mapped into 2D face set in order to understand the relationship between each face and its neighboring faces. A face definition consists of a face geometry and a set of bounding edges and vertices. If a value is assigned to edges and vertices based on their geometric and topological information, then these values can be converted to a face score includes face geometry information together with the information about the edges and vertices on the faces. These face scores will be the input to the NN to recognize features. One of the latest researches, developed by Osturk and Osturk in 2001, also used face score method to identify 8 feature types in solid object [22]. In their work, some complex and interacting features or any non-standard feature could also be solved.

Other type of NN-based method is proposed by Nezis and Vosniakos [17]. In their works, the input to the NN is a binary vector of 20 elements called representation vector (RV) and the output of the net is also a binary vector of 7 elements representing the feature class. RV is actually a strategy to normalize a sub-graph, extracted from an AAG of a solid model, in that way it can be used as an input for NN. In their method, a heuristic algorithm is used to decompose a full AAG of a solid model (CAD data) into several sub-graphs that each sub graph will be used to identify a feature. There are also some other NN based approaches to solve feature recognition problem such as one developed by Meeran and Zulkifli. One who wants to explore their method can be referred to [18].

The results of all the NN methods, as mentioned before, have proofed their significance advantages over the conventional methods to recognize a feature which are not include in feature library or not defined previously since the NN-based systems have a capability to learning and recognizing a new object based on their previous experiments [19]. The next section will be devoted to explore a NN-based method for feature recognition developed by Nezis and Vosniakos [17].

2. Neural Network-Based Feature Recognition

The basic approach of the method developed by Nezis and Vosniakos [17] is based on the information concerning the topology of faces in a solid object where features are identified by an NN system that has a capability to work like a biological neuron. As mentioned before, the topology of the 3D model is converted into a full AAG. From that point the AAG will be decomposed into several sub-graphs which later, each of them is used as the input to the NN after they have been ‘re-formed’ in the way that the network are able to use them.

The processes in this method to recognize features from a 3D model can be mainly classified into three processes;

- A process that a full AAG of a solid model is decomposed into several sub graphs by using a heuristic algorithm.
- A process where each sub-graph is formed and normalized in a way it can be understand by the NN system.
- A process where the NN system is trained to recognize features by providing appropriate examples of each feature.

In order to fully understand the processes, one should agree that a feature definition in Nezis and Vosniakos method is considered to be a collection of faces that having some manufacturing significances. In this method, there are some basic features; pocket, passage, protrusion, step, slot, blind slot and corner pocket that illustrated in figure 5. The description of each feature defined by Nezis and Vosniakos can be found in their paper [17]. This feature
concept will be used thorough this method in feature recognizing. The flow process of this system is also illustrated at Appendix 2.

Figure 4 Breaking down an AAG into sub-graphs. The black (darkest) dots represent the primary face of a feature and the light gray dots are the internal faces. The other dots represent external faces.

Figure 5 Most basic features used in Nezis and Vosniakos method [17]

2.1 Attributed Adjacency Graph & heuristic principle for decomposition

The AAG proposed by Joshi [14] can be defined as follows;

\[ G = \{N, C, A\} \]  

where graph \( G \) is a function of \( N \) which is the set of nodes representing the faces of the object. \( C \) is the set of connection arcs between the nodes and \( A \) is the set of connection attributes which denotes the kind of edge relationship (the convexity and concavity)[17]. The simple rules for constructing the AAG for a given component quoted from [14] are:

- For every face of the object, there is a unique node \( n \) in the set \( N \).
• For every edge between two faces $fa$ and $fb$, there exists a unique connection arc $c$ in $C$ connecting the two nodes $na$, $nb$ that represents faces $fa$ and $fb$.

• Every connection arc $c$ in $C$, is assigned as attribute $a$ in $A$. This is equal to ‘1’ or ‘0’ if the connection arc $c$ represents convex or concave edge respectively.

Having the full AAG of the model, the next process is to decompose it into sub-graphs. In the work of Nezis and Vosniakos, they firstly mark and categorize the faces in the full AAG. The classification follows the division of face’s classes:

• ‘A’ category faces have only concave (0) connection with others

• ‘B’ category faces have concave (0) and convex (1) connections, but only with internal faces.

• ‘C’ category faces are feature boundary faces having connections of any kind with internal faces of the sub-graph and being connected to at least one of the sub-graph’s external faces.

There are two marks for every face. The internal mark refers to the connection to internal faces, and the external one related to external faces.

After the classification done, the next step is to look for the base class. In most cases, each sub-graph has a base face around which the sub-graph is assembled. Then the process is carried on for looking the internal faces of each sub-graph. This process is done by using a searching algorithm that depends on the format of the AAG file. The further information about the searching algorithm can be found in [17]. An example of marking table from the model in figure 4 can be shown in table 1.

![Figure 6 A model with three features and its AAG](image)

Table 1 Part example from figure 4 to be marked
<table>
<thead>
<tr>
<th>Face ID</th>
<th>Status</th>
<th>Category</th>
<th>Internal Mark</th>
<th>External mark</th>
<th>Sub-graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>internal</td>
<td>A</td>
<td>4</td>
<td>0</td>
<td>SG1</td>
</tr>
<tr>
<td>F2</td>
<td>internal</td>
<td>C</td>
<td>30</td>
<td>1</td>
<td>SG1</td>
</tr>
<tr>
<td>F3</td>
<td>internal</td>
<td>C</td>
<td>30</td>
<td>1</td>
<td>SG1</td>
</tr>
<tr>
<td>F4</td>
<td>internal</td>
<td>C</td>
<td>30</td>
<td>1</td>
<td>SG1</td>
</tr>
<tr>
<td>F5</td>
<td>internal</td>
<td>C</td>
<td>30</td>
<td>1</td>
<td>SG1</td>
</tr>
<tr>
<td>F6</td>
<td>internal</td>
<td>A</td>
<td>4</td>
<td>0</td>
<td>SG2</td>
</tr>
<tr>
<td>F6*</td>
<td>external</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SG1</td>
</tr>
<tr>
<td>F7</td>
<td>internal</td>
<td>C</td>
<td>10</td>
<td>3</td>
<td>SG2</td>
</tr>
<tr>
<td>F8</td>
<td>internal</td>
<td>C</td>
<td>10</td>
<td>3</td>
<td>SG2</td>
</tr>
<tr>
<td>F9</td>
<td>external</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SG2</td>
</tr>
<tr>
<td>F9*</td>
<td>external</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SG3</td>
</tr>
<tr>
<td>F10</td>
<td>external</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SG2</td>
</tr>
<tr>
<td>F11</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>F12</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>F13</td>
<td>external</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SG2</td>
</tr>
<tr>
<td>F13*</td>
<td>external</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SG3</td>
</tr>
<tr>
<td>F14</td>
<td>external</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SG2</td>
</tr>
<tr>
<td>F14*</td>
<td>external</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SG3</td>
</tr>
<tr>
<td>F15</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>F16</td>
<td>internal</td>
<td>A</td>
<td>4</td>
<td>0</td>
<td>SG3</td>
</tr>
<tr>
<td>F17</td>
<td>internal</td>
<td>C</td>
<td>30</td>
<td>1</td>
<td>SG3</td>
</tr>
<tr>
<td>F18</td>
<td>internal</td>
<td>C</td>
<td>30</td>
<td>1</td>
<td>SG3</td>
</tr>
<tr>
<td>F19</td>
<td>internal</td>
<td>C</td>
<td>30</td>
<td>1</td>
<td>SG3</td>
</tr>
<tr>
<td>F20</td>
<td>internal</td>
<td>C</td>
<td>30</td>
<td>1</td>
<td>SG3</td>
</tr>
</tbody>
</table>

2.2 Input representation by adjacency matrix

From the previous section the Full AAG has been decomposed into some sub-graphs containing features to be recognized. The next process is to represent each sub graph in the way that the NN system can use it as the input. Since the input for NN system is only a number, a strategy have to be developed in order to obtain the goal at this process without distorting or loosing any information. Nezis and Vosniakos have concluded that there are two basic properties that should be include in developing a strategy or a representation scheme [17];

- It should be in a format that can be identified by the input layer of an ANN that can be implemented in computer software or hardware.
- There must be no overlap in the representation patterns. This means that the representation of two different features must be different. However, two different instance of the same feature class might have different representation patterns, but these have to be ‘similar’.

According to those basic properties, the adjacency matrix (AM) approach will be used as the representation pattern for the sub-graphs (feature instances). The adjacency matrix is a two-dimensional matrix that the number of the row and the column is similar to the number of faces of a sub-graph. In this matrix, the upper triangular area represents the convexity relationship.
Figure 7 The Attributed Adjacency Graph and Adjacency Matrix of a pocket feature with external face concave connections.

between faces and the lower triangular are represents the concavity relationship. The connection between two faces will be signed ‘1’. If the two faces form convex edge, the sign will be put in the convex area and, in turn, if they have a concave relationship, the sign will be placed in lower triangular area. If there is no connection between two faces, it will then be signed 0. The diagonal area (where Row \[i = Column \] ) which connects each face itself, will be signed 0.

As illustrated in figure 7, the adjacency matrix representing this pocket feature instance with eight wall faces is a matrix of \([9 \times 9]\). The base will be coded as the first face (the primary face) and the rest will be coded as the second to the ninth respectively. Consequently, the relationship between the eight wall faces and the base face will be a concave relationship. Thus, the implementation of this relationship will be that in all column elements of the first row will be signed 1 exclude the first column that categorized as the diagonal area. The relationship among the internal faces in this feature should also be taken into account and, again, their connections are also concave relationship. Thus, the representation of internal faces relationship will form the lower diagonal sign by 1 as can be seen in figure 7.

Examples of the AM of various feature instances (sub-graphs) can be seen at Appendix 3. From that representation, it can be observed that a particular pattern is created for each feature class. For example, a pocket feature will have the AM with have two unique patterns; the ‘normal’ pattern and ‘lower diagonal’ pattern without concerning how many wall faces for this feature instance.

Unfortunately, two problems have been identified regarding the use of AM [17]. First, duplication due to internal topology such as the feature of blind slot, corner pocket and step with five wall faces. The second problem was encountered is the size of the network’s input layer. To illustrate this problem, a five-wall faces pocket feature instance will have a \([6 \times 6]\) adjacency matrix. Thus, this matrix will be converted into a row vector of 36 elements corresponds to 36 input neurons. However, the input to the NN will be 121 neurons to represent a ten-wall faces pocket. This situation will be impossible for NN as there is no consistency in the input system.

2.2 The representation vector

The solution for the second problem, as it mentioned before, is to normalize the adjacency matrix for each sub-graph extracted and found from an AAG so the final representation scheme of the NN is a binary vector of 20 elements. This input scheme is called The Representation Vector (RV) [17]. The first 12 digits in the RV of any feature instance (sub-graph) reflect the answer of 12 questions that actually normalize the sub-graph. A positive
answer of each question will be numbered ‘1’ whereas the negative answer is ‘0’. Then, elements 13 to 20 of the RV of each sub-graph deal with the number of external faces linked to the particular sub-graph. By doing so, the first problem is solved and each sub-graph will be unique one to another. In addition, to deal with more complex AAG, the current RV is able to represent sub-graphs with up to 256 external faces [17] and this was suggested by Nezis and Vosniakos to deal with practical applications. The list of those questions can be seen in table 2 which quote and modified from [17].

Table 2 Questions used for filling in The RV elements

<table>
<thead>
<tr>
<th>RV element</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Is the number of internal faces in the sub-graph equal to 2 ?</td>
</tr>
<tr>
<td>2</td>
<td>Is the number of internal faces in the sub-graph equal to 3 ?</td>
</tr>
<tr>
<td>3</td>
<td>Is the number of internal faces in the sub-graph equal to 4 ?</td>
</tr>
<tr>
<td>4</td>
<td>Is the number of internal faces in the sub-graph greater than 2 ?</td>
</tr>
<tr>
<td>5</td>
<td>Are all the side-diagonal couples &quot;full&quot; ?</td>
</tr>
<tr>
<td>6</td>
<td>Is there exactly one 'empty' side-diagonal couple ?</td>
</tr>
<tr>
<td>7</td>
<td>Are the empty side-diagonal couples more than one ?</td>
</tr>
<tr>
<td>8</td>
<td>Are all the elements in the first normal equal to 1 ?</td>
</tr>
<tr>
<td>9</td>
<td>Is the &quot;A&quot; element equal to 1 ?</td>
</tr>
<tr>
<td>10</td>
<td>Is the &quot;B&quot; element equal to 1 ?</td>
</tr>
<tr>
<td>11</td>
<td>Are all the elements in the &quot;upper area&quot; is equal to 0 ?</td>
</tr>
<tr>
<td>12</td>
<td>Are all the elements in the &quot;lower area&quot; is equal to 0 ?</td>
</tr>
</tbody>
</table>

To answer the 12 questions, one should understand a number of conventions concerning the regions of the adjacency matrix. The conventions can be drawn according to the matrix formation illustrated in figure 8. In picture 8a, the main or first diagonal area of the matrix is signed by the black elements, and the gray area is referred to as the side-diagonal areas. Each of the cells in the side diagonal areas with diagonally symmetric cell forms a side diagonal couple. If one of the two cells has a non-zero element, then the couple is considered to be ‘full’, else it is ‘empty’.

![Figure 8 Characteristic areas on the AM, illustrated from [17]](image)

The darker region of the matrix shown in figure 8c is called ‘upper’ area of the matrix. This area includes all the matrix elements above and the upper diagonal except the second
element of the last column. And in figure 8d, the ‘lower’ area of the matrix includes all the matrix elements below the lower diagonal except the second element of the first normal.

The darker area on the figure 8b is called the ‘first normal’ of the matrix. And the light dark area of the figure 8e is called the ‘A’ and ‘B’ element of the matrix formation, as it was labeled respectively.

In summary, having a good understand to these conventions, one will be able to answer the routine of 12 questions and then solved the problems by producing the representative vector of any sub graph. Consequently, a sub graph of 5-wall-faces pocket feature will be the same as 10-wall-faces pocket feature and those sub-graphs will be recognized as a pocket feature.

2.4 Neural network training scheme

Before recognizing a feature from any representative vector, the NN system should be firstly trained by providing a set of training data. In the training set, the population of training pairs (input - output vectors) from each feature class have to be almost equal. The reason behind this is simply because the NN should give a fair evaluation in recognizing any input to them [17]. In addition, if there is a significant different in the number of training data of each feature, the final result may be too ambiguous. For example, if the set of training data for step class is less than the set of blind slot, a representative vector that should be a step feature will be recognize as a blind slot or the final output will give an equal values. Some examples of feature instance as the training data to the network is shown at appendix 1.

During the training process the system tries to find some functional relationships between the input and output in the set of training data given. Thus, this functional relationship is called the weight of the NN. The accuracy of the weight depends on the number of training data in the set, the cumulative desired difference between the ‘wanted or desired’ output data (the output in the training data set) and the ‘fact’ output data the actual data obtained as the function of the weight) and the activation function (a function that generalize the output data).

3. Radial Basis Function: A Proposal For Better And Faster System

Up to this point, the information of artificial NN (ANN) is limited to their basic construction, that is the input layer as the receptor, output layer and the intermediate layer. The other information so far is that a layer of a NN consists of either a neuron or two or more neurons.

In this section, some more information on the basic of NN will be discussed in brief including two popular NN types; radial basis function (RBF) and multilayer perceptron (MLP) with back propagation learning rule (BP). A heuristic comparison of the two methods will be also presented as the basis in proposing RBF network for feature recognition. In order to have a good understanding about the theory of artificial NN, one is suggested to read some resources about ANN that can be found in [21,22,23].

3.1 Network architectures

So far, the most NN type used in solving feature recognition problem is MLP network with BP learning rule [15,16,17,18]. The common network architecture of MLP usually has one hidden layer that is invisible to the user. However, two hidden layers had been used for feature recognition in Ozturk’s work [19]. The reason for this is to obtain the optimal NN architecture related to convergence and generalization performance of the network [19].

The MLP can be categorized as supervised NN. In a supervised NN, a set of inputs and outputs are already known for the network to process the learning. The way they learn is to evaluate the difference between the desired output, which already known, and the actual output
comes from the learning process as a function of the weight. The difference between the two is called the error value (cost function), which will be used to correct and change the weight in order to minimize the error. This learning process will continue until it meets the desired error, or convergence. The process pattern of the learning in MLP is called back propagation learning algorithm. The existence of this network architecture is to overcome non-linear problem when a single perceptron was not able to solve it [21].

Like MLP networks, the process flow of RBF network starts from input layer and end up at output layer. Thus, these networks can be categorized as feedforward NN. The architecture of RBF is similar to one hidden layer MLP network and they can contain one or more neuron in their input and output layer. However, learning process between the two is difference. RBF consists two phases of learning process. First, the set of input data, which is in a high dimensional space, is transformed into two dimensional input data using Euclidean distance. The result is affected by the activation function used. In this stage, it can be said that this NN performs unsupervised training. In the second stage, the mapping function (that is, the weight) can be commonly obtained by applying a linear regression algorithm. Thus, this step is a supervised learning process. A general formula of basic RBF network will be discussed later this section.

3.2 MLP problems from previous works

BP learning uses gradient descent rule that minimize the sum-squared error (SSE) over the entire training data set. The convergence to the optimal solution is accomplished iteratively by adjusting the mapping functions (the weights) through the partial derivatives of the SSE with the respect to the weights [19]. It is apparent that one main advantages of this learning rule is the time needed to proceed until it is convergence as the process would be accomplish iteratively. Although the process can be speeded up by putting momentum, a great consideration has to be taken in choosing the right momentum. The wrong momentum and learning rate could lead the learning process come to the situation where the process become unstable or divergence [22].

In the work of The Ozturk [19], They have concluded that a single layer in the intermediate of NN was not sufficient to generalize the performance of the network after error verification done because one hidden layer NN will be likely to converge in local minimum. On the other hand, Nezis and Vosniakos [17] have confirmed that a great number of training data set would make the learning process become much slower to converge. They also encountered the number of training data set used should be ‘sensible’ in the mean that a small population of data training set would demonstrate instability learning process. However, a very large data would consume too much time in the learning process.

On the other hand, one is now able to see the advantages of RBF network over MLP with BP learning rule. In RBF network, the iterative process can be avoided because the optimum weight can be obtained in linear system as follows [22];

\[
\begin{align*}
    [G] [w] &= [D] \\
    \text{where } G \text{ represents the input data that has been transformed from multi dimensional space into non-linear problem which has the form of two dimensional array. } W \text{ is the mapping function or the weight and } D \text{ is the output from the training data set.}
\end{align*}
\]

3.3 A general formulation of RBF network

To present the general formula of RBF networks, consider a feed-forward network with an input layer, a single hidden layer, and an output layer consisting of a single unit. The network
is designed to perform a nonlinear mapping from the input space to the hidden space. Let \( P \) be the dimension of the input space and the network represents a map from the \( P \)-dimensional input space to a single-dimensional output space, written as [21],

\[
s : \mathbb{R}^P \rightarrow \mathbb{R}
\]  

(3)

Quoting from the theory of multivariable interpolation in high dimension space, which can be stated as follows [24]:

"Given a set of \( N \) points \( \{X_i \in \mathbb{R}^P \mid i = 1,2,\ldots,N \} \) and a corresponding set of \( N \) real numbers \( \{d_i \in \mathbb{R} \mid i = 1,2,\ldots,N \} \), find a function \( F : \mathbb{R}^N \rightarrow \mathbb{R} \) that satisfies the interpolation condition: \( F(x_i) = d_i , i = 1,2,\ldots,N \)"

So, the general form of radial basis function with \( N \) data at point \( x \) is [21];

\[
F(x) = \sum_{i=1}^{N} w_i G(||x - x_i||)
\]  

(4)

where \( G \) is a basis functions, and \( \|\cdot\| \) denotes a norm that is usually taken to be Euclidean. \( ||x - x_i|| \) is simply a distance which tell how far the \( x \) from the point \( x_i \). The known data points \( x_i \in \mathbb{R}^P, i = 1,2,\ldots,N \) are taken to be the centers of the radial basis functions.

Inserting the interpolation conditions of above equations, we obtain the following set of simultaneous linear equation for the unknown coefficients (weight) of the expansion;

\[
\begin{bmatrix} G_{11} & G_{12} & \cdots & G_{1N} \\ G_{21} & G_{22} & \cdots & G_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ G_{N1} & G_{N2} & \cdots & G_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix}
\]  

(5)

where;

\[
G_{ji} = G(||x_j - x_i||) \rightarrow j, i = 1,2,\ldots,N
\]  

(6)

\[
d = [d_1,d_2,\ldots,d_N]^T
\]  

(7)

\[
w = [w_1,w_2,\ldots,w_N]^T
\]  

(8)

The \( N \)-by-1 vectors \( d \) and \( w \) represent the desired response vector and linear weight vector respectively. Let \( G \) denote an \( N \)-by-\( N \) matrix with elements \( G_{ij} \):
According to light’s theorem [21], the coefficient of matrix $G$ is positive definite, and so we may solve equation (5) for the weight vector $w$, obtaining:

$$[W] = [G]^{-1}[Y]$$

where $G^{-1}$ is the inverse of interpolation matrix $G$. Although in theory we are always assured a solution to this interpolation problem, in practice we are unable to solve it when the interpolation matrix is singular or close to singular. At this point, regularization theory can help by perturbing the matrix $G$ to $G + \lambda I$ [22]. A complete discussion of regularization technique can be found in [21,22].

4. Conclusion

The feature recognition method proposed by Nezi s and Vosniakos [17] is not perfect yet. This method is only able to recognize the predefined features and some basic compound features in $2\frac{1}{2}$ dimensional. The heuristic method, especially heuristic no. 9, in some cases, fails to assign into sub-graphs faces that are internal of features [17] something like T-slots would not be able to recognize by this method.

The RBF network, in general, could save much time compared to MLP network. However, it also comes with some limitations that one may encounter when it is used to solve medium to large and very large data set. For example, if we have 1 million training data, it means that we have to prepare the memory space in computer as many as $1 \times 10^{6 \times 2}$ memory space because it is needed to create two dimension matrix with order $R^{1 \times 10^6}$ and plus just about the same amount memory allocation for performing the calculation process. The total memory needed would be disastrous unreachable even for the-state-of-the-art computer technology nowadays [25]. One may say that it could be performed by iterative method to solve the linear. However, to solve very large data, iterative method will take a lot of time and this is also another limitation of RBF. Fortunately, there are some new methods regarding the limitation of RBF proposed by Beatson [26,27], from The University Of Canterbury. From the work his done, he could solved around 1 million data of linear problem for RBF within an hour when it was used to solve computer graphics. In addition, the time and memory required to solve RBF problem can be even reduced by implementing The Orthogonal Least Square (OLS) method. This algorithm has a function to reduce some redundancy in the use of basis function to model the data. A comprehensive literature about this method can be found in [22].
The heuristic rule developed to decompose the AAG into sub-graphs seems to depend on the format of the AAG file and if one has a different format of AAG file, the general rule of this heuristics should be modified.

Although, the feature recognition method presented in this report is not a new method, it is still worthwhile to discuss. In addition, the advantages of the proposed NN over the current one seem to give a new atmosphere in this field. Thus, this can be a starting point and stimulator for further researches in feature recognition.

References
Appendix 1: A variety of data training set for neural network learning

Figure A1.1 Features in a training set, scanned from [17]

Appendix 2: Feature recognition architecture system
3D model

AAG File

Heuristic algorithm for decomposition

Sub-graph  Sub-graph  Sub-graph

Input Training set data

Output Training set data

Adjacency matrix

Representative vector of a subgraph

\[ [W] = [G]^+ \ [Y] \]

Trained Neural Network system

Mapping function (The weight)

RBF Network, the proposed NN to use

List of Feature in the form of binary digits

001000000 000100000
000000100 100000000

Figure A2.1 RBF Network based feature recognition, re-drawn & modified from [17]
## Appendix 3: Adjacency matrix and RV for various simple features

<table>
<thead>
<tr>
<th>Feature Description</th>
<th>Adjacency Matrix</th>
<th>RV Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Walls Pocket with 1 external face</td>
<td><img src="image1.png" alt="Image" /></td>
<td>RV = 00011001111110000000</td>
</tr>
<tr>
<td>6 Walls protrusion with 1 external face</td>
<td><img src="image2.png" alt="Image" /></td>
<td>RV = 00011001101110000000</td>
</tr>
<tr>
<td>Single walls step with 7 external faces</td>
<td><img src="image3.png" alt="Image" /></td>
<td>RV = 10000001000111111110</td>
</tr>
<tr>
<td>2 Walls slot with 4 external faces</td>
<td><img src="image4.png" alt="Image" /></td>
<td>RV = 01000010000111110000</td>
</tr>
<tr>
<td>4 Walls Passage with 2 external faces</td>
<td><img src="image5.png" alt="Image" /></td>
<td>RV = 00010010100011000000</td>
</tr>
<tr>
<td>5 Walls blind pocket with 4 external faces</td>
<td><img src="image6.png" alt="Image" /></td>
<td>RV = 00010100001111100000</td>
</tr>
<tr>
<td>6 Walls slot with 4 external faces</td>
<td><img src="image7.png" alt="Image" /></td>
<td>RV = 00010010000111110000</td>
</tr>
</tbody>
</table>

Figure A3.1 Adjacency matrix for various sub-graphs extracted from figure 5 with the RV code as the input to the NN. The second column is sub-graph with only internal faces and the third column with external faces. The fourth column is the Adjacency Matrix where the external faces are taken into account to make each feature instance unique without any ambiguity. The brown/gray elements in the Adjacency Matrix represent the relationship between two faces.
Appendix 4: Summary of project documentation

Word count: 6743 words (Abstract – References)
143 words (Author’s profile)
7069 words (Total report including all appendices)

Font Type: Times New Roman, 12pts.

Supporting software: Mechanical Desktop 3.0, Visio 2000, AutoCAD 2000,
Microsoft Excel 2000, Microsoft MSWord 2000,
Adobe ImageReady 3.0.

Resources: Internet, scientific journal, reference books, lecturing hand-outs

Herry studied mechanical engineering in the National Institute of Technology and Science (ISTN-Indonesia) from 1993 to 1998. He obtained his degree with very satisfactory result. He decided to live in Auckland soon after graduate. After the first two years in Auckland, he then worked as a fulltime plant operator for Winstone Wallboard Ltd. In 2002 he gave up being a fulltime employee and enrolled in a postgraduate diploma course in Mechanical Engineering Department at The University of Auckland. He has currently been pursuing his study goal and hoping to get the master engineering study degree at the end of the first semester of 2003. His current interests are in the field of computer-aided drawing, computer graphics and visualization, web programming and flash web designing. In this semester, his diploma engineering research projects cover Life Cycle Assessment, Automated Process Planning and Computational Fluid Dynamic.