A robotic welding system using image processing techniques and a CAD model to provide information to a multi-intelligent decision module.

Journal:	*Assembly Automation*
Manuscript ID:	AA-09-021.R1
Manuscript Type:	Original Article
Keywords:	Welding < Industrial Robotics, Just in time < Assembly, Industrial Robotics, Assembly < Industrial Robotics, Fabrication < Industrial Robotics, Seam Tracking < Welding < Industrial Robotics

A robotic welding system using image processing techniques and a CAD model to provide information to a multi-intelligent decision module.

Abstract

A system is proposed that uses a combination of techniques to suggest weld requirements for ships parts. These suggestions are evaluated, decisions are made and then weld parameters are sent to a program generator. New image capture methods are being combined with a decision making system that uses multiple parallel AI techniques. A pattern recognition system recognizes shipbuilding parts using shape contour information. Fourier-descriptors provide information and neural networks make decisions about shapes. The system has distinguished between various parts and programs have been generated so that the methods have proved to be valid approaches.

1. Introduction

Although some shipyards have used robots for welding steel for 20 years, the integration of robotic welding presents problems [1-3]. Low levels of repeatable welds within some ships means that, although the quality and speed of robotic welding are acceptable, the generation of programs capable of applying weld has proved difficult [4-6]. Many welding robots work primarily in "teach-and-playback" mode with teach pendants and joysticks [7-9] but that further limits flexibility and other programming methods are being considered such as intelligent pointers [10,11].

Although the superstructure of a ship may be complicated, it can be complexity of scale [4]. A ship's superstructure can be a complicated object made from a large number of simple objects. Most are made from either metal bar (of varying sizes and shapes) or metal plate and additional items are often cut from metal plate [12].

A new automated welding system is being created that uses artificial intelligence (AI) techniques to determine where to weld these sorts of parts. New image capture methods are being combined with a decision making system that uses multiple parallel AI techniques. The proposal uses object oriented programming techniques to create the framework for the system and uses imaging software to capture and process image data.

The system was to have used a combination of AI techniques [13-16] to suggest weld requirements [2,5]. The original flow diagram for the new system is shown in figure 1. Suggestions were to be evaluated and decisions made regarding weld(s) without any reference to the available computer aided design (CAD) information and without considering the use of a graphical user interface. The parameters were then to be sent to a program-generator to produce a robot program for the shop-floor.

The image-capture [17] and program-generator systems are working [4] and a camera mounted above the assembly line at VT Shipbuilding in Portsmouth captured images (frames) and new image-processing and object-recognition sub-systems have been successfully created that operate on the images. The decision-module is now under construction.

New sub-systems have successfully distinguished between various ships' parts by processing shape information so that Fourier-descriptors can be extracted and sets of descriptors associated with training-sets in order to make decisions [4]. In that work the images were broken into equal segments and the segments represented as complex numbers by referring coordinate points to a random starting point. Fourier-descriptors were extracted by transforming object descriptions into the frequency domain.

Since data points around the contour were expressed as complex number values and not as complex functions of length, the usual complex form of Fourier series was of little use. As contours were sampled, Discrete Fourier Transforms (DFTs) were considered but were replaced by more efficient Fast Fourier Transforms (FFTs). Once transformed then data was expressed as Phase & Magnitude. The modulus of this transformed data was considered in order to discard phase information, and consequently, discard operations that effected phase. Descriptors were now invariant (within a small error) for rotation, dilation and translation.

Figure 1 here – First System Flow Diagram for the planned new system

2. Artificial Intelligence Techniques

AI techniques are discussed that are being tested for use within the proposed system.

2.1 Fuzzy expert systems

Fuzzy logic can deal with uncertainties generated by incomplete or partially corrupt data. The technique uses the mathematical theory of fuzzy sets to simulate human reasoning. Humans can easily deal with ambiguity (areas of grey) in terms of decision making, yet machines find it difficult [18,19]. *Bloch* stated that there are a number of reasons why imprecision was inherent to images: imprecise limits between structures or objects, limited resolution, numerical reconstruction methods and image filtering[18]. Fuzzy Logic is well suited to this area. Applications in structural object recognition and scene interpretation have been developed using Fuzzy Sets within Expert systems. Fuzzy expert systems are suitable for applications that handle uncertain and imprecise situations but they do not have the ability to learn as the values within the system are preset and cannot be changed.

2.2 Rule based systems

A Rule-Based System describes knowledge of a system in terms of IF…THEN..ELSE. Specific knowledge can be used in order to make decisions. These systems are good at representing knowledge and decisions in a way that is understandable to humans. Due to the rigid rule-base structure they are less good at handling uncertainty and are poor at handling imprecision. A typical rule-based system has four basic components: a list of rules or rule base, which is a specific type of knowledge base; an inference engine [20,21] or semantic reasoner, which infers information or takes action based on the interaction of input and the rule base; temporary working memory; and a user interface or other connection to the outside world through which input and output signals are received and sent [10,11,22,23].

2.3 Case based reasoning systems

The concept in Case-Based Reasoning is to adapt solutions from previous problems to current problems. These solutions are stored within a database and can represent the experience of human specialists. When a problem occurs that a system has not experienced, it compares with previous cases and selects one that is closest to the current problem. It then acts upon the solution given and updates the database depending upon the success or failure of the action [24]. Case-Based Reasoning systems are often considered to be an extension of Rule-Based Systems. They are good at representing knowledge in a way that is clear to humans, but they also have the ability to learn from past examples by generating additional new cases. Case-based reasoning has been formalized for purposes of computer reasoning as a four-step process[25]: 1. Retrieve: Given a target problem, retrieve cases from memory that are relevant to solving it. A case consists of a problem, its solution, and, typically, annotations about how the solution was derived. 2. Reuse: Map the solution from the previous case to the target problem. This may involve adapting the solution as needed to fit the new situation. 3. Revise: Having mapped the previous solution to the target situation, test the new solution in the real world (or a simulation) and, if necessary, revise. 4. Retain: After the solution has been successfully adapted to the target problem, store the resulting experience as a new case in memory. Critics argue that it is an approach that accepts anecdotal evidence as its main operating principle. Without statistically relevant data for backing and implicit generalization, there is no guarantee that the generalization is correct. However, all inductive reasoning where data is too scarce for statistical relevance is inherently based on anecdotal evidence.

2.4. Fourier-descriptors

Describing shapes is essential for pattern recognition [4,26,27]. Shape description techniques divide into boundary-based and region-based. Region-based techniques consider whole objects while boundary-based techniques concentrate on boundary-lines. Boundary-based methods are more popular because shape classifications are based on contour features. Many integral transforms can be used as feature extractors, for example: general-integral, Mellin, Cross-correlation, Radon or Fourier-Mellin. Fourier-Mellin descriptors have tended to perform better than others in noisy conditions (such as those in shipyards) but are not translation-invariant. Properties of DFTs are analogous to continuous Fourier Transforms. Power spectra of DFTs are invariant under cyclic translation of the input vector. Fourier-based methods can be applied efficiently using FFTs. That was the selected method and shape information was processed so that Fourier-descriptors could be extracted. Fourier-descriptors characterize object shapes in a frequency domain. Shape-based objects can be classified using conventional Fourier-descriptors, generic Fourier-descriptors or wavelet-Fourier-descriptors. Generalized Fourier-descriptors are described by Smach[28].

2.5 Artificial Neural Networks (ANNs)

Previous work on active recognition differs in object representation, information combination and future planning. Invariant pattern recognition is complicated [29] and classification processes can be (1) invariant feature extraction or (2) feature classification. Feature classification can be achieved using Artificial Neural Networks (ANNs) [3,4,10,30-32]. ANNs typically have inputs and outputs, with processing within hidden layers in between. Inputs are independent variables and outputs are dependent. ANNs are flexible mathematical functions with configurable internal parameters. To accurately represent complicated relationships, these parameters are adjusted through a learning algorithm. In 'supervised' learning, examples of inputs and corresponding desired outputs are simultaneously presented to networks, which iteratively self-adjust to accurately represent as many examples as possible [33]. Once trained then ANNs can accept new inputs and attempt to predict accurate outputs. To produce an output, the network simply performs function evaluation. The only assumption is that there exists some continuous functional relationship between input and output data.

2.6 Feature Recognition

Recording an image of ship's part is simple, but recognizing what that image portrays requires comprehension. Feature recognition is a first step in translating an image of part of a ship into welding instructions. Methods in the literature for automated feature recognition tend to match structures identified in a part representation with some pattern in a knowledge base, often using if–then rules [34]. A disadvantage is that they cannot easily deal with features that cannot be matched with known patterns. Pattern Recognition techniques were originally posed as statistical problems, derived from work in Discriminant Analysis and applying Bayes Theorem [35].

2.7 Hybrid systems

The purpose of a hybrid system is to combine desirable elements from different AI techniques. For example, fuzzy expert systems are poor at learning due to the fixed nature of the values needed. This can be improved by the creation of neuro-fuzzy systems. Neural networks have the ability to learn which in turn can enable the fuzzy systems to learn. This work is attempting to create new systems that are hybrids of different AI techniques in an effort to use the best from each technique. Every natural intelligent system can be considered as hybrid because they perform mental operations on both the symbolic and subsymbolic levels. For the past few years there has been an increasing discussion of the importance of A.I. Systems Integration [36,37].

3. Proposed Solution

This Section explains the existing RinasWeld / Motoman System in place at VT Shipbuilding and discusses how additional systems may be integrated with the existing systems. The new proposed system is discussed including: software systems required, image processing systems and the use of multiple artificial intelligence techniques to make decisions.

3.1 Existing System

The existing system at VTS is shown in Figure 2. The system consists of two software systems working in series to construct viable robot programs. The first system, the CAD model interpreter, accepts a CAD model and determines the welds required. This data is fed to the Program Generator which re-orientates the weld requirements in line with the actual real-world orientation of the panel. The program generator then sends any programs sequentially to the robot (normally one program per weld line). Additional software systems could be incorporated into the existing system at the point where the robot programs are sent to the Robot System. This is because the transmission protocol at this point is standard TCP/IP and any programs to be sent can be viewed as text files.

3.2 Proposed System

The new proposed system in Figure 3 shows that data will be gathered from a post-processed image. The data will then be combined with the data contained within a CAD model. The Multi-Intelligent Decision Module will then use multiple AI techniques to suggest a required weld. This weld requirement will then be displayed for the operator to check. If the operator rejects the suggestion the system will learn from that rejection and suggest a different requirement. Assuming the operator now accepts the requirement, the system will generate a compatible robot program by using the program generator and post-processing systems.

Figure 2 here – Existing RinasWeld / Motoman System

Figure 3 here – Revised System Flow Diagram showing the inclusion of the computer aided design (CAD) model an the graphical user interface (GUI)

3.2 (a) Software Systems

In the same way that the construction of the superstructure of a ship is broken into smaller elements such as sections, units and panels; the weld requirements can also be sub-divided. Figure 4 shows that a Panel is considered the largest practical part.

Figure 4 here – Hierarchy of a Ship Panel

This is intuitive as the factory system is such that Panels have specific documentation. It has therefore been proposed that each Panel be made up of a collection of one or more Jobs. The inclusion of this layer allows collections of Welds (the next layer) to be logical grouped together in order to improve production efficiencies. The final layer is that Welds are collections of Points. This is where the anatomy concept falls back into line with the real-world. Any linear weld can be described by determining just two points, the start and end. All the other points that are required can be extrapolated from these two points. When creating the software systems to generate a required robot program, it was decided that an object oriented approach would reduce the development time. Object oriented technique offers easier handling of complexity within software and allow changes to be simpler during debugging.

Figure 6 shows some of the different positions that the end effector must move through to successfully weld. The touch sense points allow the robot to determine the precise location of the part to be welded in relation to the end effector using some simple force feedback sensing [38,39]. This is important as the end effector must be positioned within 2mm of the correct weld start point to achieve satisfactory weld quality.

Figure 5 here – End Effector Path Diagram

3.2 (b) Early Prototype Image Processing Systems

The image processing systems involved detecting edges, line identification and geometric data generation[17]. This data can then be used to identify the different objects within the image. A software package named 'WiT 8.3' by Dalsa Coreco was initially used to reduce development time of the first prototype image processing systems. This software had a graphical interface which was used to create and test prototype algorithms that were exported as VB.net compatible functions for inclusion within a .net framework software package. In the early prototypes, the image was read, converted to greyscale and then put through a low pass filter. The low pass filter removed some of the noise in the image and reduced the occurrence of small random edges. The image was then operated on by an edge tracing function which used a Prewitt edge detection algorithm and then collated any edges into a collection of geometric lines. These lines were then overlaid onto the filtered greyscale image for viewing. Later systems used Fourier-descriptors and Artificial Neural Networks and the most recent systems have introduced new corner finding algorithms to effectively reduce noise.

3.2 (c) Multiple AI techniques

The many different methods of implementing AI each have their own strengths and weaknesses. Some effort has been made in combining different methods to produce hybrid techniques with more strengths and fewer weaknesses. The Neuro-Fuzzy system which seeks to combine the uncertainty handling of Fuzzy Systems with the learning strength of Artificial Neural Networks is an example of this. This paper proposes a system of using multiple AI techniques to decide on weld requirements for a job. The system will combine the Real-world visual data captured through the image processing algorithms with the data provided by the CAD model. It will then use this combined data to present differing AI systems with the same information. These systems will then make weld requirement suggestions to a Multi-Intelligent Decision Module (Figure 7). This module will evaluate the suggestions and determine the optimum weld path. The suggestions will be passed to the existing robot program generator.

Figure 6 here – Multi-Intelligent Decision Module Diagram

4. Current Progress

The current state of the research is that the robot program generation systems have been created and tested. These systems have been used to produce consistent straight line welds. A simple edge detection system was created using the WiT software. Figure 1 shows the initial image. Figure 8 shows the edges as detected by the algorithm created during this research. The edge detection in this instance is good as the object can be identified from its perimeter detail. There is also detail present that has been caused by corners between metal pieces. This shows that the edge detection is not reliant upon a high contrast. The external perimeter detail is more defined than the internal detail. The work surrounding the AI systems is in the early stages and will be taken further over the next six months. During this time the multi-intelligent decision module framework will be created and combinations of AI techniques tested. The AI techniques to be tested will include Rule-based, Case-based and Fuzzy systems. Meanwhile, improvements were made to the image processing systems.

5. Image processing

Information about shape or pattern is held within contours, so Fourier-descriptors were applied to the contours of shapes being classified. The edge detected image in figure 8 was processed to produce closed line shapes so that no lines were left open and hanging. Contours were assumed to be closed curves in complex space. An arbitrary point moving around the contour generated a complex function `f'. If the point moved around the contour at a constant velocity `v', then at every time 't' a complex number `c' was defined such that $c = f(t)$. `t' is not necessarily real time, it represents a section of length around the contour. Because contours were closed, it implied that there existed a value `T' so that $f(t + nT) = f(t)$, where nT was the contour length. So f can be expressed as a complex Fourier series. These Fourier coefficients depended on starting point and differed with respect to a parameter `τ' along the contour, so that for each τ there was a set of Fourier coefficients of the function $f(t) = f(t + \tau)$. If $f(t) = f^{(0)}(t)$ then other functions around the contour will be $f(t) = f^{(0)}(t + \tau)$.

Considering: Translations, Rotation and Dilation.

Translation: If $A_n^{(0)}$ is a set of fourier coefficients from a contour function then translation by a complex vector Z results in a contour function expressed in the Inverse Fourier Series:

$$f(t) = f^{(0)}(t) + Z = \sum_{-\text{infinity}}^{\text{infinity}} A_n^{(0)} \exp[jnt] + Z$$

Therefore the Fourier coefficients of the translated contour are: $A_n = A_n^{(0)}$ for n (where not equal to zero) and $A_n^{(0)} + Z$ for $n = 0$. All coefficients except A_0 are invariant of translation. A_0 depicts the complex vector indicating the position of the centre of gravity.

Rotation: If centre of gravity is at the origin then a rotation of the contour function f(t) about the origin, with an angle of φ produces another function f(t) where $f(t) = \exp[j\varphi] f^{(0)}(t)$. With f(t) expressed as the inverse Fourier transform, coefficients of the rotated contour will be: $A_n = \exp[j\varphi] A_n^{(0)}$.

Dilation: Similarly, Dilation of the contour by scale factor R creates Fourier coefficients of form: $A_n = RA_n^{(0)}$.

6. Extracting Fourier-descriptors

The general form of fourier coefficients of a contour after Translation, Rotation and Dilation is $A_n = \exp[jn\tau] R \exp[j\varphi] A_n^{(0)}$, where coefficients $A_n^{(0)}$ are coefficients of the original contour. They are not useful in that form because they contain information on orientation, and shape only is needed. Considering $B_n = A_{1+n+1}.A_{1-n} / A^2_1$, then applying that expression after rotation, dilation etc… results in an expression that does not contain τ, R or φ. If coefficient A_0 is not used then these B_n coefficients are invariant under Translation, Rotation and Dilation. Thus B_n coefficients represent shape (or form). Fourier coefficients were invariant under Translation, Rotation and Dilation [4] and just represented shape. The ANNs were trained using Backpropogation algorithms. Nets were considered trained when error became zero (within pre-set ranges). A number of teaching runs were required before outputs converged. A teaching net was created to take two sets of inputs and two sets of demand vectors.

7. Testing

To test the systems, a teaching net was created to take two sets of inputs and two sets of demand vectors. The layout was a 5-38-4 pattern. After 150 test-runs the network gave the outputs shown in Table One. Errors were used to update weights within the ANN. A number of teaching runs were required before outputs converged. A teaching net was created to take two sets of inputs and two sets of demand vectors.

Table 1 here - Output from two sets of inputs

Weights were saved. The application net was combined with the Description Program and set up to analyze two shapes in different orientations. In 100 tests the program classified 98 shapes correctly after three frames of video. The 2 pattern program operated with a 98% classification rate within three frames. The training net was then modified to take 3 sets of inputs and demand vectors. Weights were frozen after 500 test runs and the outputs are shown in Table 2.

Table 2here - Output from three sets of inputs

Programs were tested with 3 different shapes in different orientations. In 100 tests the program classified 97 shapes correctly after three frames. The 3 pattern recogniser worked with 97% classification.

The results were good compared to other systems but attempts were made to improve the results further by carrying out some post-processessing on the edge detected image.

8. Improving the system

After processing the edge detected image (figure 8) to obtain a clear image (Figure 9) using geometrical rules, then the edge was sampled. The continuous line was converted to equally spaced line segments and then to polylines by specifying endpoints for each segment. The new sub-systems successfully distinguished between various ships' parts by:

- Edge detecting the image (figure 1 to figure 8).
- Sampling points around the edge detected image.
- Calculating distance between endpoints of windows around sampled points.
- Taking points with minimum distance to be corners.
- Using corners and connecting lines to extract Fourier descriptors.
- Associating sets of descriptors with training sets.
- Deciding.

Points were sampled and corners were detected based on the diagonal length of a segment's bounding box. Interspacing distance was equal to the diagonal of the bounding box divided by a constant M (set to 50). M was determined empirically by testing a range of values and finding the value that produced the best accuracy; increasing M increased noise and decreasing M created smoother edges so that some corners were removed.

Points could be sampled once an interspacing distance, S, had been calculated. An empty set was created to store sampled points. Each point was then appended to that set. A distance holder D was set to zero. The new algorithm was:

(i) Euclidean distance d between two consecutive points was added to D.

(ii) If D was less than the interspacing distance S, then i was increment by 1 and step (i) was repeated.

Otherwise: (a) A new point, q, was created, approximately S distance away from the last sampled point. qx and qy were calculated to be (S - D) / d distance between point i-1 and point i.

(b) Append q to the set of sampled points and insert q before point i.

(c) Repeat from step (i) without incrementing i until i > |points|.

The new algorithm found corners from this primitive information and from higher-level patterns that determined possible insertions or corner deletions. Firstly, corners were found based on the distance between the beginning of a line segment around a point and the end of that line segment.

For example, considering a point at pi

SEGMENTi = |pi – W, pi+W|

where W is a constant window and…

|pi – W, pi+W| is the Euclidean distance between the points pi - W and pi + W.

As the edge of a shape bends at a corner, the SEGMENT of points shortens, and a local minimum SEGMENT is a likely corner. To find an initial corner set, all SEGMENTs were first computed. Median SEGMENT length was found and a threshold t was set to be equal to the median x 0.9. For each SEGMENT, if the SEGMENT was a local minimum below the threshold t, then the SEGMENT was a corner. Line segments around a part all had a window of +/- 10 points either side of the point being considered (although +/- 5 were used in practice). Shorter SEGMENTs were around some points at corners and those points were considered corners. Points on straighter sections had SEGMENTs that were close to the median SEGMENT length and were not corner candidates.

After this set of corners was found, some higher-level processing found missed corners and removed false positives. The system checked to see if each consecutive pair of corners passed a Line-test. This similarity was represented through a ratio of Distance(points; a;b) to Path - Distance(points; a;b).

If the ratio was above a set threshold then the segment between points a and b was a line. If the part segment between any two consecutive corners did not form a line, then there were additional corners in-between. Missing corners were assumed to be approximately halfway between corners. Since these potential corners were below the original threshold t, the threshold was relaxed and the new corner was taken to be the point with minimum SEGMENT. This process of adding corners was repeated until all segments between pairs of consecutive corners were lines.

A check was then conducted on subsets of triplet, consecutive corners. If three corners were collinear, then the middle corner was removed. This process checked and removed false positives. Three consecutive corners were collinear if the part segment between the outer corners passed a Line-test.

230 images of nine different part shapes were initially used to test the corner-finder. A Douglas-Peucker's algorithm was implemented along with Sezgin's corner-finder and a simple differentiation algorithm. The algorithms had filters to remove close or overlapping corners. Two measures were used to determine the accuracy of the corner-finders: correct number of corners found and an all-or-nothing measure. The first was calculated by dividing the number of correct corners found by the total number of correct corners perceived by observation of each processed image. The second measure checked that only the minimum number of corners to segment a figure were found (in other words the part shape had no false positives or negatives). That was calculated by taking the number of correctly segmented parts divided by the total number of parts; it was either correct or incorrect.

Shapes were redrawn so that lines went directly from corner to corner. This removed noise. Fourier-descriptors were then extracted from the contours of the shapes being classified.

The corner finding system improved on other corner-finders that were considered. All-or-nothing accuracy for the new system was over 20% better than that of the Douglas-Peucker implementation.

9. Testing and results for the improved system

As an example, programs were tested with 3 different shapes in different orientations. In 100 tests the program classified 98 shapes correctly after just one frame and better than 99 after three frames. Programs were then modified to take 4 training sets and demand vectors. This ran for 6112 test runs. Over 50 tests the program classified 48 shapes correctly after just one frame and 49 after three frames.

These results were compared with those achieved by the most recently published system for identifying ship's parts *(Sanders, 2009)* and the same shapes were used for comparison. With the 2-pattern program (bottom graph) that system only operated with a 98% classification rate within three frames whereas this system operated with close to a 100% classification rate with three frames. The 3-pattern recogniser worked with 97% classification after three frames but the new system worked with 99% classification.

The graph in figure 7 shows the result for distinguishing between two different ships' parts (upper graph) and three different ships' parts (lower graph). It shows a substantial improvement when the new corner finder was

added. The improvement was especially significant when a part needed to be identified quickly (after only one frame) or when a part needed to be identified within several other ships' parts. The percentage accuracy of the most recently published algorithm and the initial prototype system described here is shown as blocks in figure 7 and percentage accuracy of the new algorithm is shown as blobs.

Figure 7 here – Comparing the prototype system with the new system incorporating the corner finder.

10. Discussion and conclusions

A proposed system has been presented that uses image processing techniques in combination with a CAD model to provide information to a multi-intelligent decision module. This module will use different criteria to determine a best weld path. Once the weld path has been determined then the program generator and post-processor can be used to send a compatible program to the robot controller. The progress so far has been described.

The initial results from the whole work are suggesting that a combination of systems (Case Based Reasoning, Fuzzy Expert Systems, Rule-Based Systems and ANN) could offer the ability to handle the necessary uncertainty whilst still returning a correct weld path (when all / enough factors are known).

Different shapes were successfully identified using a simple pattern recognition system that used an ANN and that system was improved by using a corner identifier. The system provided shape contour information that was invariant of size, translation and rotation. Since acquiring and processing new images is an expensive task, it is desirable to take a minimal number of additional views and the new methods quickly and successfully identified parts after only one frame.

The new system used a rudimentary curvature metric that measured Euclidean distance between two points in a window but the improved accuracy and ease of implementation can benefit other applications concerning curve approximation, node tracing, and image-processing, but especially in identifying images of manufactured parts with distinct corners.

References

1. Jacobsen NJ, (2007) Robot welding of hatch coamings for large container ships. Industrial Robot: An International Journal 34 (6): 456–461.

2. Chen SB (2007) On the Key Technologies of Intelligentized Welding Robot. T.-J. Tarn et al. (Eds.): Chapter in Robot. Weld., Intellige. & Automation, LNCIS 362. Springer-Verlag Berlin Heidelberg, pp. 105–115

3. Fan Ding, Shi Yu, Li Jianjun, Ma Yuezhou and Chen Jianhong (2007). Computer Simulation of Neural Network Control System for CO2 Welding Process. Chapter in Robot. Weld., Intellige. & Automation, LNCIS 362. Springer-Verlag Berlin Heidelberg, pp . 117–125.

4. Sanders DA (2009) Recognizing shipbuilding parts using artificial neural networks and Fourier descriptors. Proc of the Institution of Mechanical Engineers Part B – J of Engineering Manufacture 223 (3): 337-342.

5. Tarn, Tzyh-Jong; Chen, Shan-Ben and Zhou, Changjiu (Eds.) (2007) Robotic Welding, Intelligence and Automation, Lecture Notes in Control and Information Sciences Vol. 362, ISBN 978-3-540-73373-7. Springer-Verlag Berlin Heidelberg.

6. Sanders DA and Rasol, Z (2001). An automatic system for simple spot welding tasks. Total vehicle technology: challenging current thinking pp 263-272.

7. Sanders, D (2008). Controlling the direction of “walkie” type forklifts and pallet jacks on sloping ground. Assembly Automation 28 (4), pp 317-324.

8. Stott IJ and Sanders D (2000) The use of virtual reality to train powered wheelchair users and test new wheelchair systems. Int Jrnl of Rehab Res 23 (4), pp 321-326.

9. Sanders DA and Baldwin A (2001). X-by-wire technology. Total vehicle technology: challenging current thinking pp 3-12.

10. Sanders, DA; Tewkesbury, GE (2009). A pointer device for TFT display screens that determines position by detecting colours on the display using a colour sensor and an Artificial Neural Network. Displays 30 (2), pp 84-96.

11. Sanders DA, Urwin-Wright SD, Tewkesbury GE, et al (2005). Pointer device for thin-film transistor and cathode ray tube computer screens. Electronics Letters 41 (16), pp 894-896.

12. Sanders DA, Lambert G and Pevy L (2009). Pre-locating corners in images in order to improve the extraction of Fourier descriptors and subsequent recognition of shipbuilding parts. Proc of the Institution of Mechanical Engineers Part B – J of Engineering Manufacture *Paper JEM1553 in press*.

13. Sanders D (2008). Progress in machine intelligence. Industrial Robot: An International Journal 35 (6), pp 485-487.

14. Tewkesbury GE and Sanders DA (2001). The use of distributed intelligence within advanced production machinery for design applications. Total vehicle technology: challenging current thinking pp 255-262.

15. Sanders D (1999). Perception in robotics. Industrial Robot: An International Journal 26 (2), pp 90-92.

16. Hudson, AD; Sanders, DA; Golding, H, et al (1997). Aspects of an expert design system for the wastewater treatment industry. J of Systems Architecture 43 (1-5), pp 59-65.

17. Sanders D (1993). System Specification 2. Microprocessing and microprogramming 38 (1-5), pp 833-834.

18. Bloch I (2005) Fuzzy spatial relationships for image processing and interpretation: a review’, Image and Vision Computing 23 (2), pp: 89-110.

19. Liao S-H (2005) Expert System Methodologies and Applications – A Decade Review from 1995 to 2004, Expert Systems with Applications, 28 (1), pp 93-103.

20. Sanders DA and Hudson AD (2000). A specific blackboard expert system to simulate and automate the design of high recirculation airlift reactors. Mathematics and computers in simulation 53 (1-2), pp 41-65.

21. Sanders, DA; Hudson, AD; Tewkesbury, GE, et al (2000). Automating the design of high-recirculation

airlift reactors using a blackboard framework. Expert systems with applications 18 (3), pp 231-245.

22. Chester, S; Tewkesbury, G; Sanders, D, et al (2007). New electronic multi-media assessment system. Web Information Systems and Technologies 1, pp 414-420.

23. Bergasa-Suso J, Sanders DA and Tewkesbury GE (2005). Intelligent browser-based systems to assist Internet users. IEEE Transactions in Education 48 (4), pp 580-585.

24. Kolonder JL (1994) Case-based reasoning. Morgan Kaufmann.

25. Aamodt A and Plaza E (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. Artificial Intelligence Communications 7 (1), pp 39-52.

26. Sanders DA, Harris P and Mazharsolook E (1992). Image modelling in real-time using spheres and simpe polygedra. 4th Int Conf on Image processing and its applications 354, pp 433-436.

27. Sanders DA (1995). Real-time geometric modelling using models in an actuator space and cartesian space. J of robotic systems 12 (1), pp 19-28.

28. Smach F, Lemaitre C, Gauthier JP, Miteran J and Atri M (2008). Generalized Fourier descriptors with applications to objects recognition in SVM context, Jnl of mathematical imaging & vision, 30 (1), pp 43-71.

29. Wood J (1996) Invarient pattern recognition, Pattern Recognition, 29 (1), pp. 1-17,

30. Erwin-Wright S, Sanders D and Chen S (2003). Predicting terrain contours using a feed-forward neural network. Eng Apps of AI 16 (5-6), pp 465-472.

31. Urwin-Wright, S; Sanders, D; Chen, S (2002). Terrain prediction for an eight-legged robot. J of Robotic Systems 19 (2), pp 91-98.

32. Sanders, DA; Haynes, BP; Tewkesbury, GE, et al (1996). The addition of neural networks to the inner feedback path in order to improve on the use of pre-trained feed forward estimators. Maths and computers in simulation 41 (5-6), pp 461-472.

33. Hinton GE (1992) How neural networks learn from experience. Scientific American, 267 (3), pp: 144-151.

34. Babic B, Nesic N and Miljkovic M (2008) A review of automated feature recognition with rule-based pattern recognition, Computers in Industry 59, pp 321–337.

35. Ripley BD (1996) Pattern Recognition and Neural Networks, Cambridge University Press.

36. Sun R and Bookman L (eds.) (1994). Computational Architectures Integrating Neural and Symbolic Processes. Kluwer Academic Publishers.

37. Wermter S and Sun R (eds.) (2000). Hybrid Neural Systems. Springer-Verlag, Heidelberg.

38. Sanders D (2008). Environmental sensors and networks of sensors. Sensor Review 28 (4), pp 273-274.

39. Sanders D (2007). Force sensing. Industrial Robot: An International Journal 34 (4), p 268.

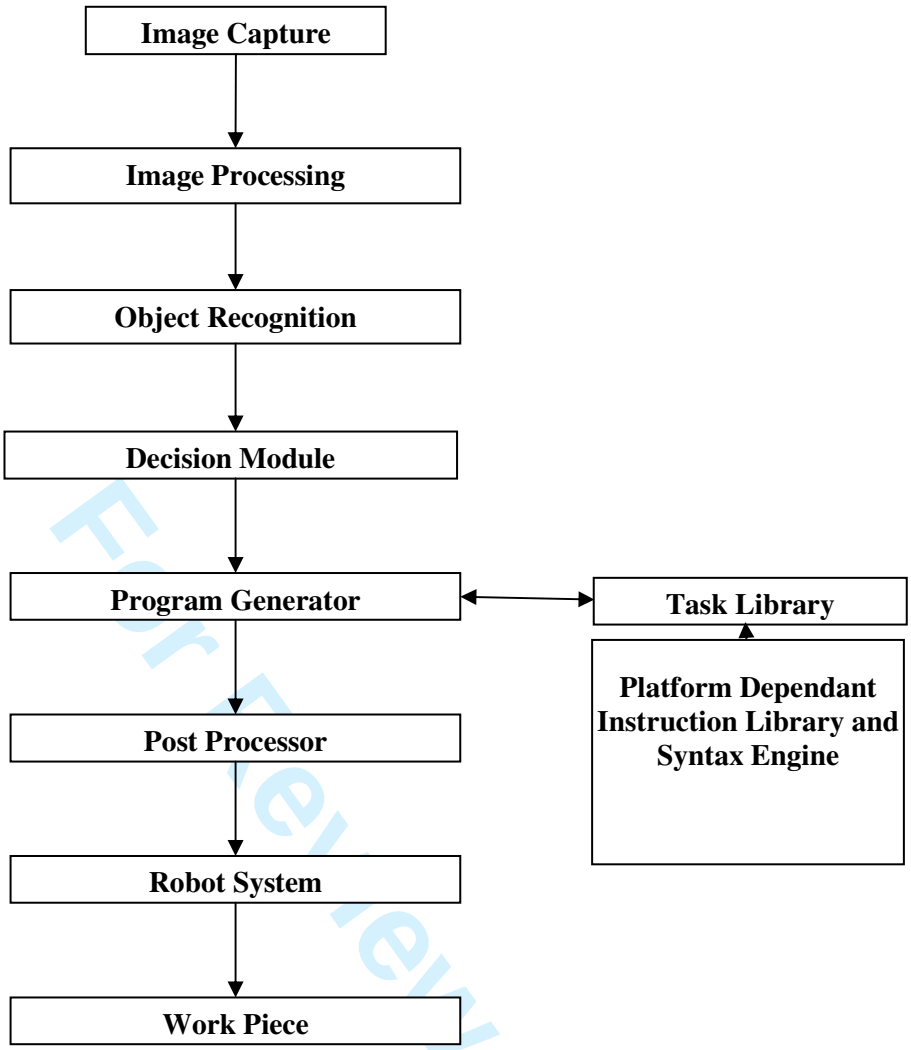

Figure 1 – First System Flow Diagram for the planned new system

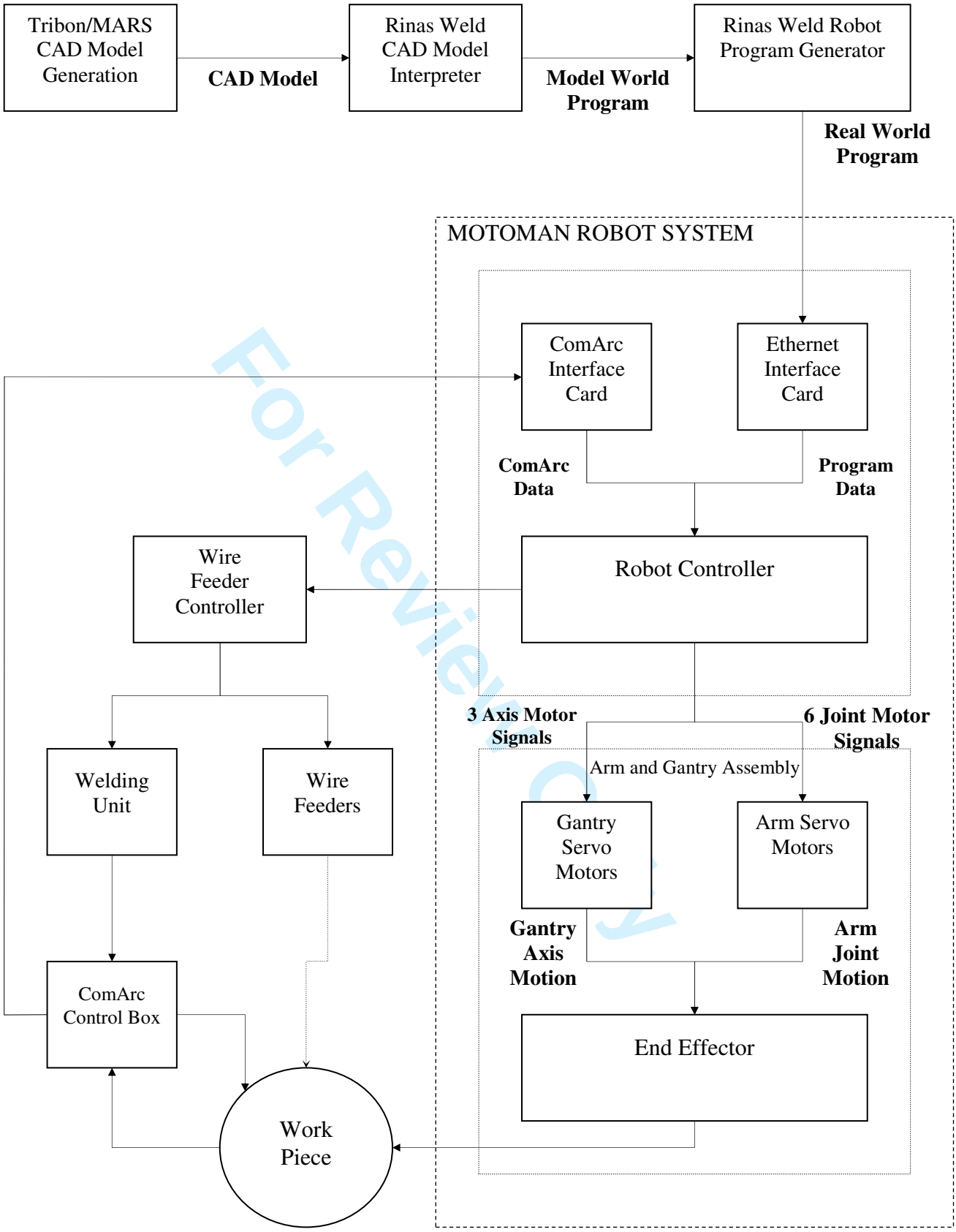

Figure 2 – Existing RinasWeld / Motoman System

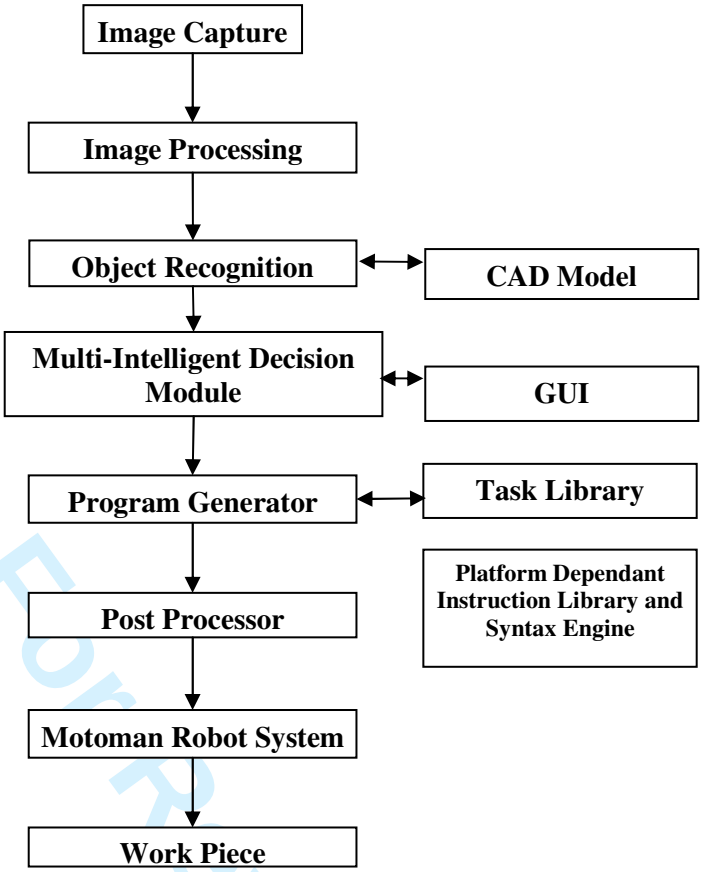

Figure 3 – Revised System Flow Diagram showing the inclusion of the computer aided design (CAD) model an the graphical user interface (GUI)

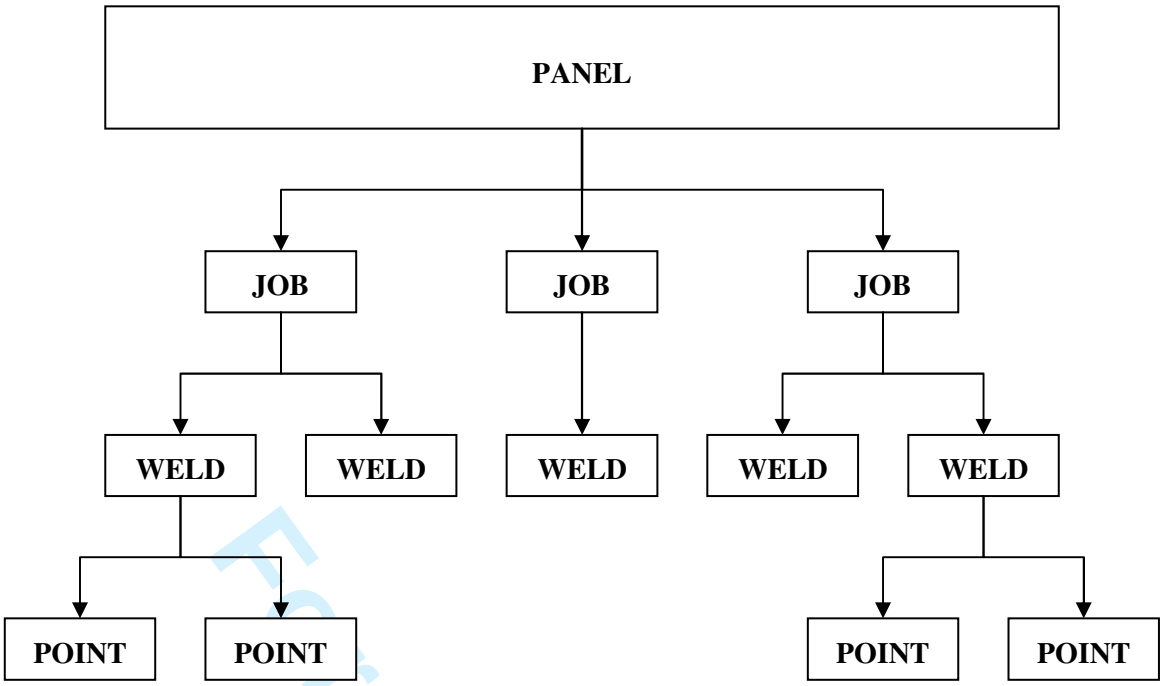

Figure 4 – Hierarchy of a Ship Panel

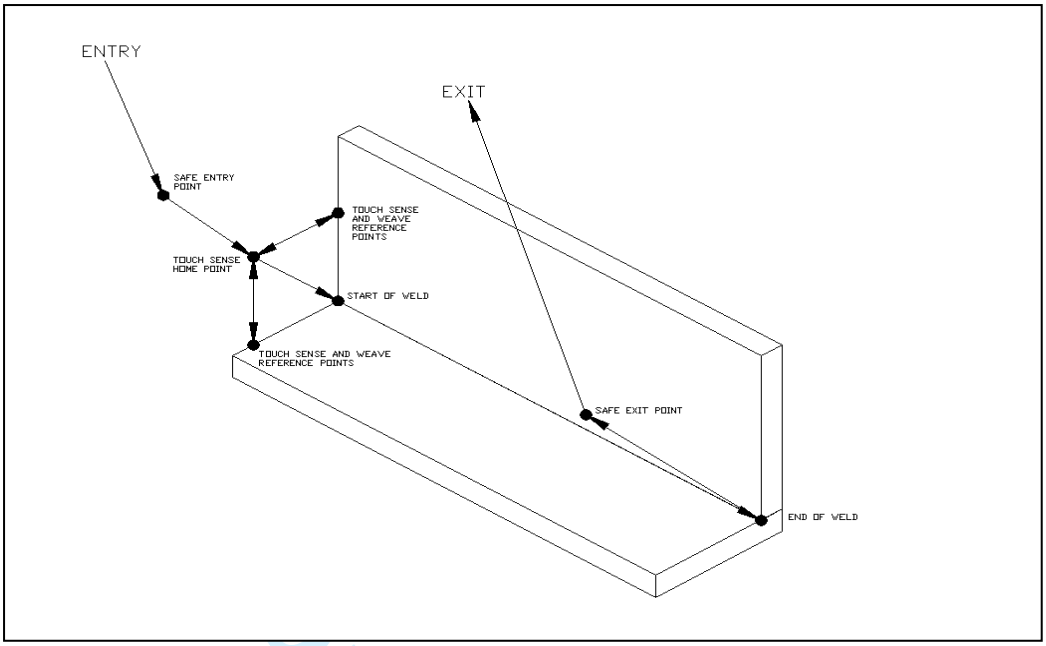

Figure 5 – End Effector Path Diagram

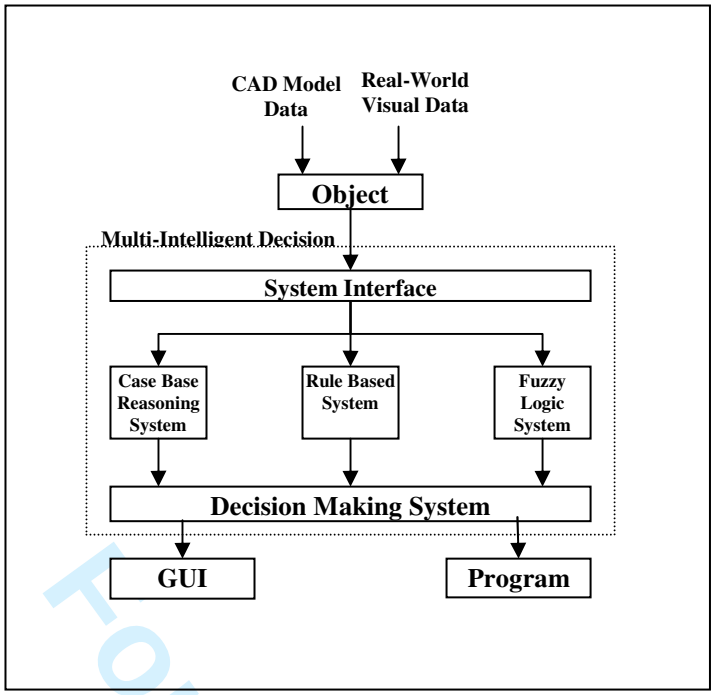

Figure 6 – Multi-Intelligent Decision Module Diagram

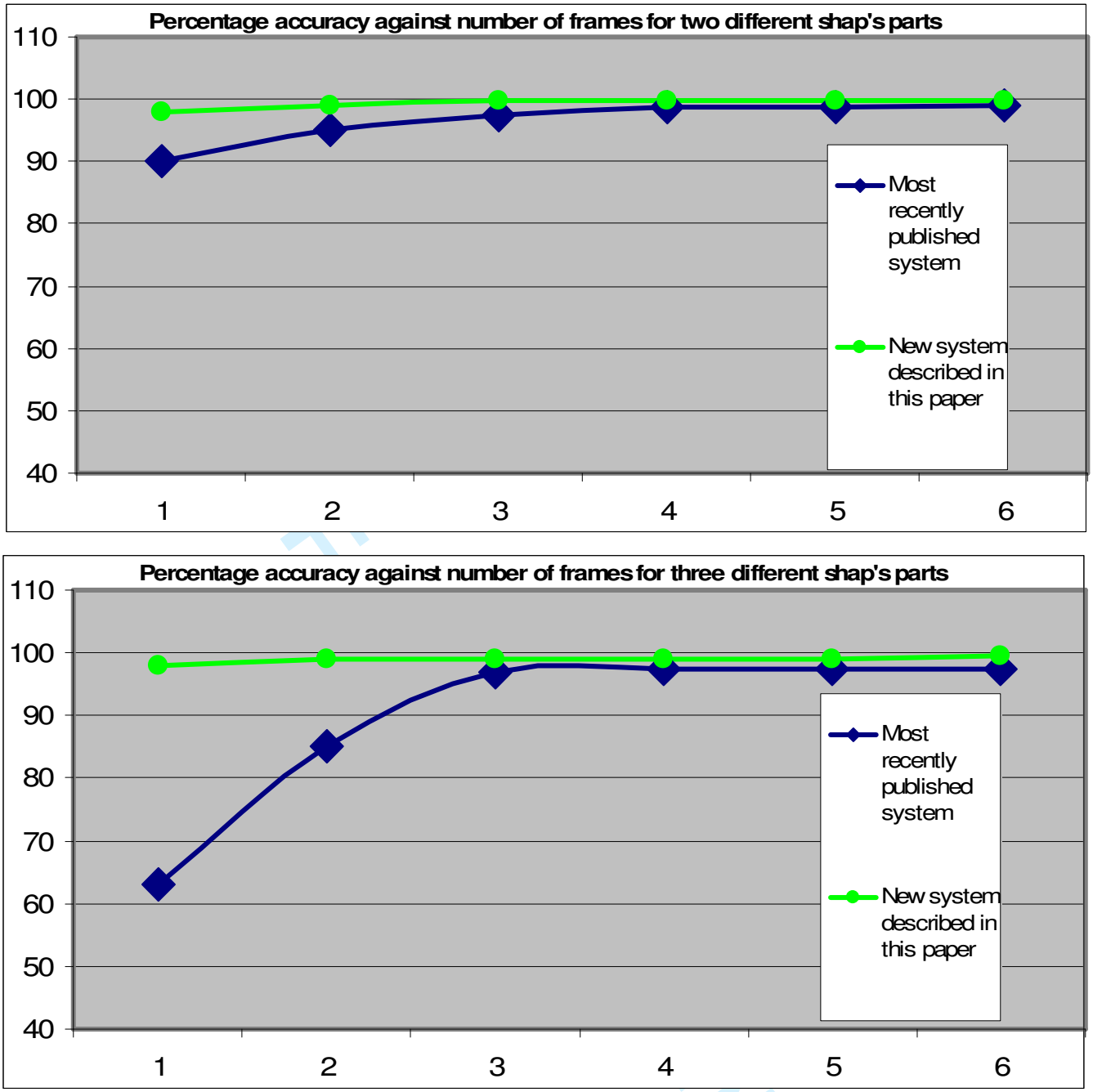

Figure 7 – Comparing the prototype system with the new system incorporating the corner finder.

Table 1 - Output from two sets of inputs

Input Set	Output	Demand	Input Set	Output	Demand
1	1 -5.99E-06	1 0	2	6.05E-05 .9999	0 1

Table 2 - Output from three sets of inputs

Input Set	Output	Desired Output	Input Set	Output	Desired Output
1	1	1	3	9.87E-08	0
	1.2E-06	0		4.6E-07	0
	7.5E-07	0		.99999	1
2	3.86E-06	0			
	.9998	1			
	5.69E-07	0			