A Taxonomy of Occlusion in View Signature II Representations: A Regular Language for the Representation of 3-D Rigid Solid Objects

Peter A.R. Cole

Department of Information Technology Murdoch University,
Murdoch, 6150, Western Australia.. p.cole@murdoch.edu.au

Abstract. The View Signature II (VSII) representation is a viewer-centred modelling scheme. The VSII representation requires one VSII view signature to represent a single view of an object and is invariant to rotation about the z axis. This is achieved by constructing a linear string of alpha-numeric characters from circular strings representing different concentric levels in a view of an object. The tokens and algorithmic methods for the construction of a VSII produce a regular language for the modelling of 3-D rigid solid objects. This paper briefly describes the VSII representation and the general construction methods. Due to the cyclic nature of the representation it has been possible to define a taxonomy of occlusion in VSII representations. This paper discusses the differing forms of occlusion and gives methods where appropriate to assist in matching occluded views with complete views in the model store.

Keywords : - 3-D syntactical pattern recognition, occlusion, viewer-centred modelling scheme

1. Introduction

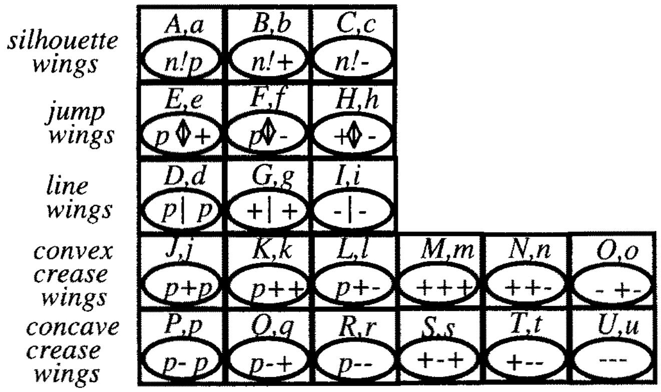
Studies have shown [1], that edge and junction information form very strong cues for perceptual recognition. We model the relationship between edges, junctions and surfaces to produce unique models of objects. The mechanism chosen to bind surface and edges together was Chen's Object Wings [2]. A *view signature II* (VSII) is a string of alpha-numeric characters that represent a single view of a 3-D solid rigid object. Each VSII is an ordered list of *level signatures* (LS). A LS is an ordered list of *junction signatures* (JS) and have a *concentric depth* dimension. The syntax of the View Signature representation is shown in Fig. 1. Junction signatures represent junctions and their coterminating edges by the use of non-numeric tokens which represent Wings. A simple wing is a triple of primitives, {surface, contour, surface}, where a contour is any edge type. A wing is well suited to model binary relationships, but, the paradigm it is based on is ill suited for extension into ternary or larger relationships. We therefore moved away from the wing paradigm to base our representation on Junction and View Signatures, described in detail in [3]. The VSII representation requires one VSII view signature to represent a single view of an object. The notation used to indicate junction type in a junction signature (JS) is the number of edges or

```
VSII            = PLS {VSIIseparator SLS}
PLS             = PJS {PJS}
SLS             = {LJS | DJV} LJS {LJS | DJV}
VSIIseparator   = ","
PJS             = W jsize W {W}
LJS             = jsize W {W}
DJV             = "."
jsize *         = integer
W               = binary | unary | NCS
binary *        = "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" | "J" | "K" | "L" | "M" | "N" |
                  "O" | "P" | "Q" | "R" | "S" | "T" | "U"
unary *         = "a" | "b" | "b" | "d" | "e" | "f" | "g" | "h" | "i" | "j" | "k" | "l" | "m" | "n" | "o" |
                  "p" | "q" | "r" | "s" | "t" | "u"
NCS **          = "!" | """ | "#" | "$" | "%" | "&" | "(" | ")" | "*" | "/" | "=" | "?" | "@" | "[" | "\" |
                  "]" | "^" | "_" | "{" | "}" | "~"
```

* *see discussion of view signature notation above,* ** *see discussion on occlusion below*

Fig. 1. Syntactic rules for the VSII representation in EBNF.

contours that coterminate at it, e.g., a junction with two edges radiating from it would be of junction type 2. The removal of 13 redundant wings from Chen's catalogue, resulted in our catalogue of 21 wings. We have also defined a new depth-resolution



surface types : n: null, p:planar, +: convex, -: concave.

edge types : !:silhouette, ◊: jump edge, | : line edge, +: convex crease, -: concave crease.

Fig. 2. The View Signature wings catalogue.

dependent edge type, the *line edge*. A line edge is an edge where the surfaces adjacent to the edge are of the same type and orientation. This change allows us to model 2-D features independent of resolution. Jump edges, are left as Chen defined them. Taking into account the changes mentioned resulted in the catalogue in Fig. 2. The individual wings in the catalogue for view signature building are denoted by alphabetic characters (Chen numbered his wings in the range 1..34). It is possible to have edges and contours in the image that are connected at only one end (unary cotermination), denoted by lower case characters. The upper case characters denote those that are connected at both ends (binary cotermination). The wings representation was thus extended to incorporate a description of all connectivity. Construction of a VSII follows a set of rotational rules which are shown in Fig. 3. A significant

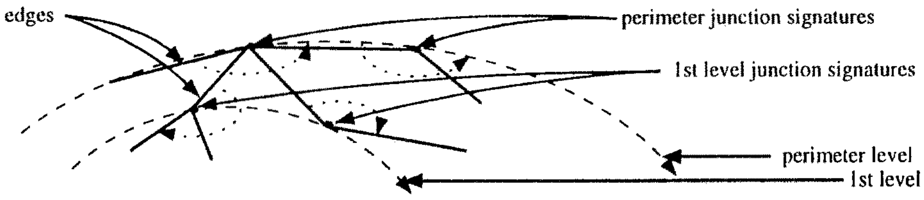


Fig. 3. Rotational rules for construction of junction signatures of different level.

difference between *Perimeter Junction Signatures* (PJS) and *Level Junction Signatures* (LJS) is their completeness. A PJS is created complete with all the edges attached to it listed. However, LJSs' are incomplete, missing the edge that represents the descending edge that lead to the junction, called the *lead-in edge.* A PJS is constructed by listing the entering edge, the junction size, any descending or unary coterminated edges and the exiting edge (see Fig. 4(a)). An LJS is constructed by listing the junction size and any descending or unary coterminated edges listed in a clockwise ordering in an arc beginning from the lead-in edge. The lead-in edge is not listed in the LJS as it is already listed at the previous level where it was a descending edge (see Fig. 4(b)). A definition of a *sub-level signature* (SLS) is a list of level junc-

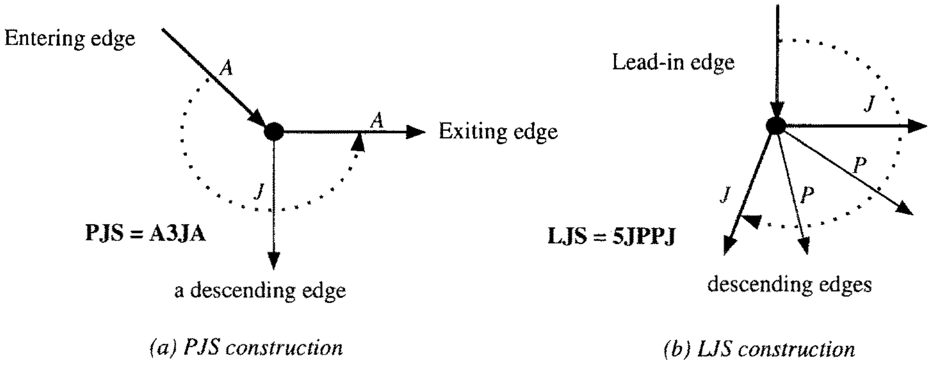


Fig 4. Detail of PJS (a) and LJS (b) construction.

tion signatures all having the same concentric depth, i.e., all junctions being connected via an edge listed on the immediate higher level. The essence of a VSII representation is its *cyclic* nature. We use the word cyclic, as the first level of each VSII is the perimeter level and we are required to totally circumnavigate the perimeter of the object, capturing the perimeter junction signatures (PJS) that lie about it, to produce the *perimeter level signature* (PLS). At this stage the PLS is a cyclic string (circular list) and we are required to normalise this list to obtain a linear PLS (linear string). The normalised PLS is appended to the VSII (previously an empty string). The normalisation process is discussed in [3]. When this first step has been achieved, each PJS that has a descending edge (see Fig. 4(a)), is investigated in order of its appearance in the PLS. A valid descending edge is an edge that leads to a junction within the object (binary cotermination). Each investigation of a descending edge will result in a level junction signature (LJS) being produced. The resulting ordered list of LJS's is termed a sub-level signature (SLS) and will be appended to the PLS if the segmentation of object is not complete. The PLS and subsequent SLSs are separated from each other by a VSIIseparator, a comma (,). The syntax of a PJS and LJS differ as shown in Fig. 1. The next step is to investigate the 1st SLS by listing the junction signatures that have not been seen on a previous level. If an LJS has been seen at a previous level it is marked with a *deja vu* (DJV) symbol, a period (.). The resulting SLS is appended as before and this process is repeated until all junctions that have a connected path to the perimeter of the object are found. The DJV symbol is very important to the representation. After a SLS has been constructed it is tested as to its contents. If it contains only DJVs the segmentation of the object is complete, as this condition indicates that the segmentation has already reached the lowest concentric level in the object. Therefore, due to the concentric nature of the construction process a VSII representation is invariant when the view of an object is rotated about the z axis.

2. Handling occlusion

To handle the presence of occlusion in a view signature requires an addition to the signature notation previously described. In a complete model, we are confident of the cotermination properties of all edges. When occlusion is present we no longer have that confidence. Therefore, to allow the modelling of occluded edges, we use an additional set of symbols, termed *No Confidence Symbols* (NCS). An individual NCS

Wing	A,a	B,b	C,c	D,d	E,e	F,f	G,g	H,h	I,i	J,j	K,k	L,l	M,m	N,n	O,o	P,p	Q,q	R,r	S,s	T,t	U,u
NCS	!	"	#	$	%	&	(	)	*	/	=	?	@	[	\	]	^	_	{	}	~

Fig. 5. Catalogue of No Confidence Symbols (NCS) with their respective Wings.

acts as an ambiguity, in that it does not indicate the cotermination property of the edge that it represents. When, during segmentation, we detect an occluded edge we mark it with an NCS. An NCS indicates that we know the object wing type of the edge but we do not know its cotermination properties. Each wing in the catalogue has a corresponding NCS as shown in Fig. 5 above.

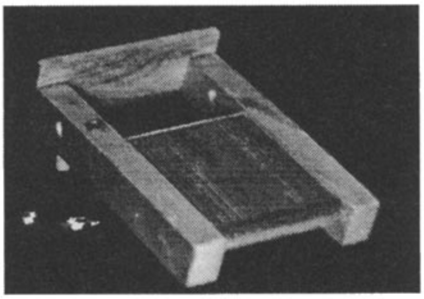
2.1. Discussion of examples of occlusion

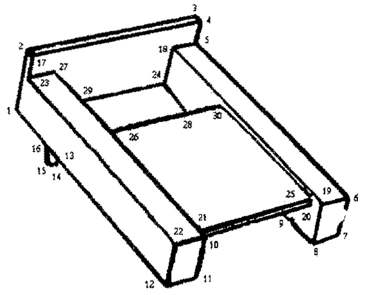
Fig. 6(a) is an image of a Note-roll Holder and Fig. 6(b) is its corresponding edge map before thinning with the junctions numbered according to order of first sight in the segmentation. Segmentation of the edge map and depth information in conjunction will produce the following VSII components :

perimeter level signature	: A2A3JA2A3JA3PA3JA2A3JA3DA4DJA2A3JA3EB2B2B3E
1st sub-level signature	: 3JJ3JJ3JP3JJ3JJ2P3DJ3JJ3JJ..
2nd sub-level signature	: .3PJ3PJ..3AA....3JP3AA3PP.3JP3JP.
3rd sub-level signature	:2D.2D.3DA3AD.2D3DA3AD2D..2D.2D

These level signatures concatenated produce the following complete VSII :
A2A3JA2A3JA3PA3JA2A3JA3DA4DJA2A3JA3EB2B2B3E,3JJ3JJ3JP3JJ3JJ2P3DJ3JJ3JJ..,
.3PJ3PJ..3AA....3JP3AA3PP.3JP3JP.,2D.2D.3DA3AD.2D3DA3AD2D..2D.2D

It should be noted that the junction marked 1 in Fig. 6(b) below is the first junction in the segmentation and also, coincidentally, after the normalisation of the perimeter level signature, the first junction of the normalised perimeter level.


(a)



(b)

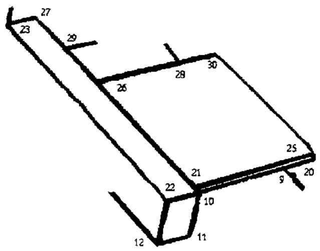
Fig. 6. Note-roll Holder

3. Taxonomy of occlusion in the VSII representation

When resolving matches with occlusion in a VSII representation, consideration of the type and degree of occlusion is required. We define the taxonomy of occlusion, and where apprσpriate, the strategies to match occluded VSII representations with complete VSII models in the model store. To assist in the visualisation of the occlusion we use the concentric representation of the object adjacent to the occluded object edge image.

3.1 Simple occlusion

Simple occlusion occurs when none of the junctions reachable from un-occluded perimeter junctions during a segmentation have a concentric depth greater than the concentric depth of the object. Fig. 7(a) shows the un-occluded portion of a Note-roll



(a) Occluded Note-roll Holder

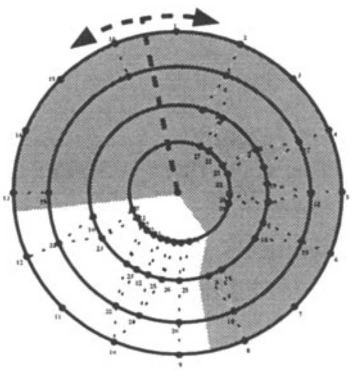

(b) representation of concentricity of complete model with occluded junctions shaded

Fig. 7. Simple Occlusion.

holder and Fig. 7(b) the representation of that occlusion in a concentric depth map of the complete model. It is clear that the un-occluded portion does not cross the centre of the of the concentric representation, therefore, the sub-string that can be extracted from the stored model will contain a complete match with the VSII that can be extracted from the input view. The segmentation algorithm would first detect junction 23 (Fig. 7(a)). The edges that coterminate at that junction are not silhouette edges, therefore, occlusion is present. The segmentation will continue to completely circumnavigate the object listing silhouette edges. This process would produce the following partial PLSs :

(1)!3DA3DA3D!	(junctions 29,26,28)
(2)!3DA4DJA2A3J!	(junctions 9,10,11,12)

To maximise the efficiency of the search process, the PLSs of the models in the model store would be searched on the larger of the two perimeter sections found (2).

```
perimeter level signature :A2A3JA2A3JA3PA3JA2A3JA3DA4DJA2A3JA3EB2B2B3E
                                             !3DA4DJA2A3J!
```

On finding a partial match an attempt to match the second perimeter section (1) would be attempted. This attempted match would fail. The object detected in the store has silhouette edges on its sub-levels signatures as indicated below. Therefore, we know that the object is perforated, that is, it has a hole within it, and a complete mismatch is not yet declared.

```
1st sub-level signature  : 3JJ3JJ3JP3JJ3JJ2P3DJ3JJ3JJ3..
2nd sub-level signature : .3PJ3PJ..3AA....3JP3AA3PP.3JP3JP.
3rd sub-level signature  : 2D.2D.3DA3AD.2D3DA3AD2D..2D.2D
```

Segmentation of the descending edges in the occluded object would be carried out using the matched partial PLS(2). Normal segmentation rules would be used and would result in the partial VSII representation below.

```
!3DA4DJA2A3J!,2P3DJ3JJ3JJ,3JP3AA3PP.3JP3JP.,.2D3D!3!D2D..2D.2D
```

The partial VSII representation above uses NCS notation. The NCS symbols act as set identifiers, so matching is a membership test of a particular wing symbol in the stored model into the NCS set of that symbol in the occluded VSII. It is quite straight forward to extract from the stored model the matched portion of the model that descends from the partial PLS, as we know the position of the first matched PJS (Sp) and the number of PJS contained in the match (Lp). To extract from an SLS1 those junctions that are reachable from the partial match of the PLS, we are required to extract a string of LJSs of length Lp beginning from a start position in SLS1, S_1 where:

$$S_1 = \left(\sum_{n=1}^{Sp} (\mathrm{jsize}_n - 2)\right) + 1 \qquad \text{and} \qquad Lp = \left(\sum_{n=Sp}^{Ep} (\mathrm{jsize}_n - 2)\right)$$

and where Ep is the position of the last matched PJS in the PLS. The term jsize, refers to the integer value which quantifies junction size in both PJS and LJS representations as shown in Figure 1. To extract the descendants from an SLS_j of the SJSs present in SLS_{j-1} we need to extract a string of SJSs of length L_j beginning from a start position S_j where:

$$S_j = \left(\sum_{n=1}^{S_{j-1}} (\mathrm{jsize}_n - 1)\right) + 1 \qquad \text{and} \qquad L_j = \left(\sum_{n=S_{j-1}}^{E_{j-1}} (\mathrm{jsize}_n - 1)\right)$$

and where the subscript j is the level of the SLS that we wish to extract the string from. The resultant partial VSII extracted from the model store matches exactly, after resolution of the NCS occurrences, that which was extracted from the occluded object.

Extracted PJS =

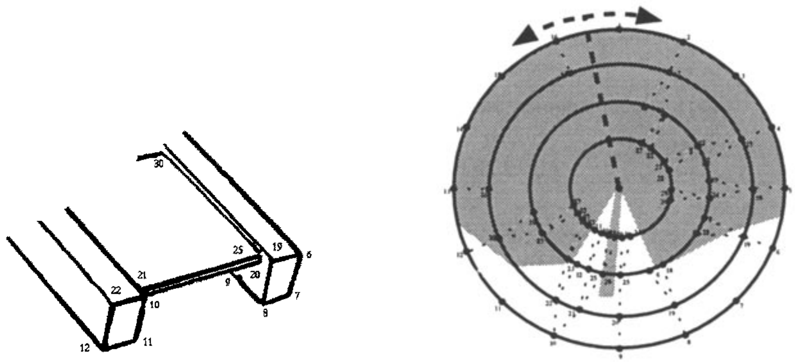
```
A3DA4DJA2A3JA,2P3DJ3JJ3JJ,3JP3AA3PP.3JP3JP.,.2D3DA3AD2D..2D.2D
```

Occluded PJS =

```
!3DA4DJA2A3j!,2P3DJ3JJ3JJ,3JP3AA3PP.3/P3/P.,.2D3D!3!D2D..2D.2D
```

The confidence of a match of this nature is reasonably high especially if there were no other complete matches with different objects in the store. However, all we can say with absolute confidence is that this portion of a model in the store has a complete concurrence with the occluded object.

3.2. Sub-furcating Simple Occlusion



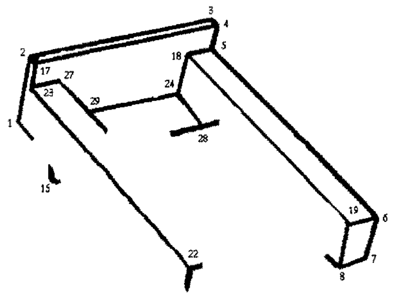
(a) Occluded Note-roll Holder (b) representation of concentricity of complete model with occluded junctions shaded

Fig.8. Sub-furcating Simple Occlusion.

Sub-furcating simple occlusion occurs when the occlusion present furcates an otherwise normal case of simple occlusion as shown in Fig. 8. The method to handle this form of occlusion is similar to the method above for simple occlusion. However, when we detect an NCS symbol in a sublevel signature (other than the lowest one) which is not matched to a DJV on the next level, the possibility of this condition exists. When this condition is detected, we need to match multiple noncontiguous segments of the occluded VSII. For a match to be declared there must be full correspondence between the object VSII and the stored model, not only for the un-occluded portions but for the occluded portion of the simple case also. This means that the tests for presence in the match must be confirmed by the corresponding absence of all the descendants of the original edge marked with the NCS.

3.3. Super-furcating Simple Occlusion

Super-furcating Simple Occlusion occurs when an object with simple occlusion has one or more perimeter junctions missing from its predominantly un-occluded portion. The occlusion of these missing junctions must not reach the lowest level of the unoecluded portion. In the object in Fig. 16, the occlusion reaches level 1 whilst the depth of the un-occluded portion is 3. This form of occlusion is simple to deal with in that the descendant of the occluded junction, (junction 13 in Fig. 9(b)), is a DJV and has no descendent junctions. The matching will use the simple occlusion resolution for the un-occluded portion (junctions 1-8) and a match of the perimeter segment which includes junction 15.


(a) Occluded Note-roll Holder

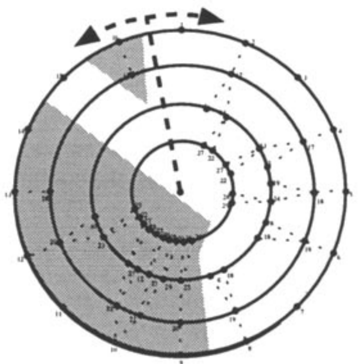
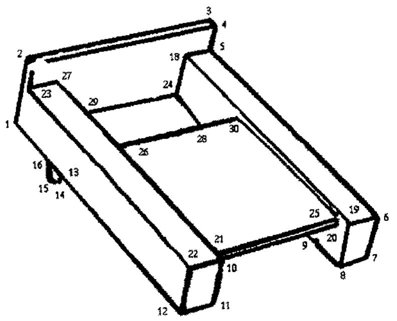

(b) representation of concentricity of complete model with occluded junctions shaded

Fig. 9. Super-furcating Simple Occlusion.

The more general case of super-furcating simple occlusion is where one perimeter junction is occluded, as in the situation where perhaps noise in the input data has caused the occlusion.

3.4. Perforating occlusion

Perforating occlusion is the occlusion of single sub-level junctions in otherwise complete inputs. Perforating occlusion is first indicated by the presence of an NCS on any level and the consequent mismatches that occur in the levels below it. The method to


(a) Occluded Note-roll Holder

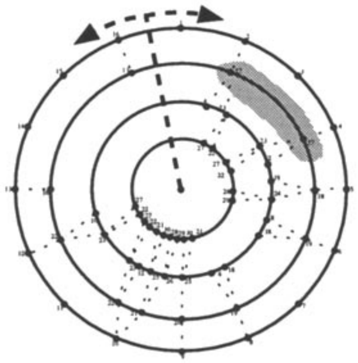
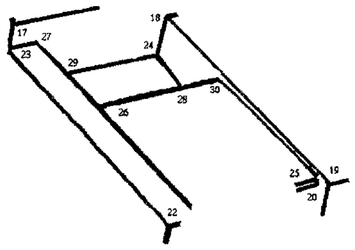

(b) representation of concentricity of complete model with occluded junctions shaded

Fig. 10. Perforating Occlusion.

deal with this is similar to that for handling sub-furcating simple occlusion, in that the descendants of the junction containing the NCS are removed from a possible match condition. After a match is satisfied to the level of the occlusion, any LJSs in deeper levels in the input need to be traversed using the known information from the partial match according to order and level. This process should produce a match to that expected for the sub-furcating case. Any junctions not yet included in the matching process then have to be reformed into proper order by taking the most likely candidate junction for the highest level of the occlusion and with its descendants, attempting a match with the remaining portion of the model. The occlusion present in Fig. 10 shows that the occlusion of a single junction can affect more than one bunch. In this case junction 17 is occluded and consequently the SJSs descending from it will be disordered. By treating all the LJSs descending from it as a related type of sub-furcating occlusion, and reconciling the match separately, we will be able to gain a complete match with the stored model with the omission of junction 17 only.

3.5. Complex occlusion

Complex occlusion in a VSII representation is defined as occlusion where no perimeter edges are detectable. This type of occlusion is the most difficult to deal with as one has no confirmed starting point for the traversal of the occluded object. Fig. 11(b) demonstrates the effect that the complex occlusion of the object in Fig. 11(a) has had. The occlusion reaches well into the concentric representation making resolu-



(a) Occluded Note-roll Holder

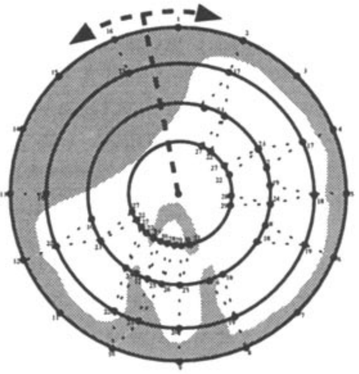
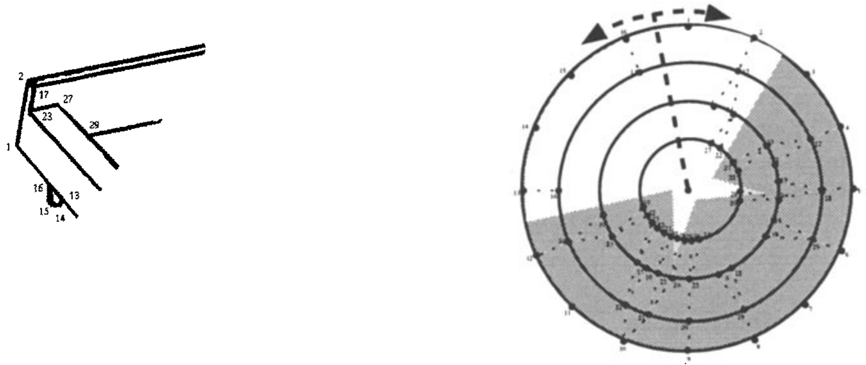

(b) representation of concentricity of complete model with occluded junctions shaded

Fig. 11. Complex Occlusion.

tion of complex occlusion very difficult when using the previously described methods. Examination of Fig. 11(b) reveals that the un-occluded portion of the object has many segments of its bunches reasonably complete, with junctions 17, 18, 19 and 20 offering the largest segments. This suggests that a matching strategy using sub-bunches would be a better option for determining a partial match than hypothesising which is the best starting junction in the object to begin segmentation from. However, the optimal method for dealing with complex occlusion has not yet been arrived at and it is the subject of further research.

3.6 Hybrid Occlusion

Hybrid occlusion is where simple and complex occlusion occurs in the same view, presents us with a slightly different problem. Here we can obtain the best match available using the simple occluded data. We can then attempt to resolve the complex occluded portion by the general method described above. The degree of confidence in the match would determine if the second stage was required.



(a) Occluded Note-roll Holder (b) representation of concentricity of complete model with occluded junctions shaded

Fig. 12. Hybrid Occlusion.

4. Conclusion

We have briefly presented the View Signature II representation, a regular language for the representation of 3-D solid rigid objects. The formulation of junction, level and view signatures has been described. The general method for the construction of a VSII has been described. The properties of a View Signature representation have been found to display terseness, adequacy, stability, locality and compatibility in general and uniqueness for all but the simplest of geometric objects [3]. We have defined a taxonomy of occlusion in the VSII representation and strategies for handling the differing occlusion types have been proposed to enable recognition of partially occluded objects. View signatures, like most modelling schemes can be adversely affected when occlusion is present. The above discussion of occlusion is by no means an exhaustive discussion of the issues involved in the resolution of occlusion in VSII models. However, it can be seen that occlusion can be handled in all but extreme cases where no connectivity between JSs in an input exists.

References

1. Chen S.W. and Stockman G.: Wings Representation for Rigid 3-D Objects. In: IEEE Proc. 10th Int. Conf. on Pattern Recognition, 398-402, Atlantic City, N.J. USA, (1990).

2. Cole P.A.R & Khan M.S.: Modelling 3-D Rigid Objects using The View Signature II Representation Scheme. In: Hlavac V. & Sara R. (eds.),: Proc. of 6th International Conference Computer Analysis of Images. Lecture Notes in Computer Science, Vol. 970, 154-161, Springer-Verlag, Berlin Heidelberg New York. Prague (1995)

3. Lowe D.G., Perceptual Organization and Visual Recognition. Kluwer Academic Press, USA, (1985)