SIMD Geometric Matching: From Polyhedra to Planar Curves

Martin Usoh and Hilary Buxton

Department of Computer Science,
Queen Mary and Westfield College,
Mile End Road,
London, E1 4NS, U.K.

Abstract

A key problem in object recognition is to evaluate the set of possible interpretations of a scene in near real-time. The SIMD parallel algorithm of Holder & Buxton [4] has been extended to deal with objects containing developable surfaces by incorporating the surface normal and axis of cone and cylinder primitives in CSG models into the matching scheme. There is no time penalty for this extension. In addition, an improved sorting to find consistent interpretations is presented which results in matching times of up to 1000 times faster over the earlier sorting technique to give total matching times of 0.17-0.56 seconds for the example scenes considered here. The geometrical transformations recovered in the validation phase are also found to be extremely accurate in the closed form tests which promises that the technique should find practical application.

1 Introduction

Computational vision may be thought of as the 3D interpretation of a 2D image or images. Many situations exist where it is advantageous for machines to accurately perceive the 3D nature of their environment and possibly take action according to the information received. An immediate example is robotic vision where 'the key task for the robot's vision system is to supply the control unit with a quantitative and symbolic description of its surroundings' [1]. This may range from the visualisation of simple geometric objects to complex multivarying scenes. Even a simple problem definition 'to identify an object from amongst a set of known objects and to locate it relative to a sensor' can break down into a set of extremely demanding operations [2].

The major problem with any recognition system is the task of investigating a vast number of possible interpretations of a scene, and to do so in real-time. The ability for humans to do this is often thought of as trivial but, on consideration, this is definitely not the case. Attempts have been made to reduce the process time wasted on spurious scene interpretations by adopting a number of local constraints which are applied to an 'interpretation tree' of the scene space. Other methods have included the introduction of parallelism on SIMD machines, namely the AMT DAP [4], [3] and the Connection Machine [5], [6].

A problem with many recognition approaches which use motion or stereo data is that they only deal with polyhedral objects in the form of faces, edges, or vertices. However, in a world containing many curved surfaces, a dedicated polyhedral recognition system is very limited. It is therefore important for such systems to be extended to curved objects, or objects containing curved features, especially if the system is to operate in a real world environment.

An obvious problem with the recognition of curved surfaces, when thinking in terms of edges, is that they do not necessarily possess real, creased edges. Very often, a simple curved geometric model has no discontinuities, e.g a sphere. An edge based approach is therefore unlikely to yield a sensible result. However, in the case of other primitives, such as cylinders and cones, it is possible to recover true edges in the form of their planar ends. Using such planar features we can apply a polyhedral based recognition algorithm, such as the Grimson and Lozano-Pérez matching technique, to the primitive. It is, however, still not possible to incorporate the developable surfaces of such primitives directly.

In the following sections we present an extension of the polyhedral edge based matcher to incorporate curved, creased edges. This will be able to deal with objects containing planar curves, such as those defined by a developable surface, and may easily be adapted for face matching. The approach we adopt uses the surface normal of the planar ends and the axis of rotation of the primitive in addition to the linear, creased edges of polygonal sections of a CSG model. These are all treated as 'real' edges and processing proceeds using the Grimson & Lozano-Pérez matching paradigm in an extension of the parallel matching algorithm of Holder & Buxton [3]. We also show that match ordering of this algorithm is very important and present a new sorting technique which, in tests conducted, has shown a significant increase in speed in terms of the total matching time over that of [3].

2 The Grimson and Lozano-Pérez Matching Paradigm

Many algorithms have been developed in recent years to match scene data to object models held in memory. They have relied on dynamically growing and pruning an interpretation tree built up from information received from the scene. The problem, in most cases, has been to identify and retrieve the orientation and location of an object from a set of known objects. The objects are modelled as polyhedra with six degrees of freedom (3 rotational, 3 translational) relative to the sensor. The basic procedure is outlined:

1) Build an accurate geometric representation of the model.
2) Recover the data features of an object in the scene of interest using an appropriate reconstruction algorithm.
3) From an interpretation tree (IT) generate a set of feasible interpretations using local constraints to prune out incorrect data to model pairings.
4) Validate feasible interpretations by testing the global validity of each match for compatibility with the model.

A feasible interpretation is one which is possible but has not yet been tested for global consistency. After sensing an object with e real edges, we can recover up to s sensed data edges. This can be formed into an interpretation tree with each node bearing e children down to a level of s-plys.

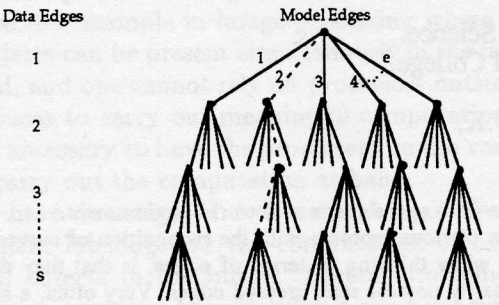


Figure 1: Interpretation tree

Note that this is a bottom-up data based approach whereby the scene data is matched to the model primitives in the interpretation tree. This means that there may be multiple data edges on a single model edge and, therefore, the number of branches remains constant at all levels. The path to any leaf node in the IT completes an interpretation of the tree. Consider the path formed by the dashed lines in Figure 1. If there were 5 model edges, then the partial interpretation down to level 3 concludes that it is possible for data edge 1 to be on model edge 2, whilst data edge 2 is on model edge 4, and data edge 3 is on model edge 1.

To put the problem of matching into perspective, let us consider an example. Given s number of sensed data edges to be matched against a model with e edges, the number of possible interpretations is e^s. Clearly this results in a combinatorial explosion of interpretations with an increase in the number of sensed edges. For a model possessing 12 edges and being tested with 5 sensed edges the number of possible interpretations stand at 248,832. If the number of recovered edges is increased to 8 there is a dramatic **increase** of 429,732,864 alternative interpretations to be considered. It is, therefore, not feasible to apply a brute-force search of the IT. The number of possible interpretations must be drastically reduced to a more manageable level.

The interpretations are limited (pruned) by taking pairs of data edges and applying simple pairwise constraints to check relationships like:

can data edge *a* pair with model edge *i* while data edge *b* pairs with model edge *j*?

The constraints we adopt are termed the distance, angle, and directions 1, 2, and 3 constraints. They are coordinate frame independent and, with the exception of the angle constraint, involve a range of measurements. The distance constraint requires that the range of distances between a point on data edge a and a point on data edge b, must be wholly included in the range of distances between a point on model edge i and a point on model edge j. The remaining constraints adopt similar measurements of angles and detailed descriptions of these may be found in [1]. These constraints maintain a consistency in the interpretation tree which becomes more powerful as we move to deeper levels.

One thing which is important to note when using edges in a matching algorithm is the ambiguity of direction. As the edge is merely a line in space, its sign and, thus, direction must be determined. A data edge fragment is termed '+' if the choice of start and stop end positions is consistent with the model edge. If different, it is termed '-'. Thus, a data edge may have the same (+) or opposite (-) direction to a particular model edge. The directions are determined during the search of the interpretation tree and also serve to increase the strength of the local constraints. In the early stages of processing, it is an overhead but, as the matching algorithm proceeds, it becomes increasingly effective. It is necessary to ascertain the direction of the data edges relative to their corresponding model edges for the validation of the interpretation. As we are working with vectors the directions must be known when calculating the model rotation [1].

3 Validation of Interpretations

After obtaining a set of feasible interpretations, it cannot be guaranteed that these are consistent with the model. This is because the constraints are applied successively to pairs of data and are therefore local. As it stands, it is not possible to know which interpretations are consistent with a single global transformation between model and sensor spaces. Each interpretation is therefore tested to:

- Determine the actual transformation parameters from model space to sensor space.
- Confirm that each data edge lies sufficiently close to its particular model edge.

The transformation is found using the quaternion method of Faugeras and Hébert [7] as an average of *all* the data edges by assuming that a vector v_m in model space is transformed into sensor space by

$$v_s = S[R]v_m + t$$

where S is the scaling, R is the rotation, and t is the translation. The transformation is applied to each datum in turn, requiring that each endpoint, after back transformation into model space, is sufficiently close to the model edge to which it is paired within a small degree of error.

4 The Holder and Buxton Algorithm

In order to reduce the processing time, two main procedures may be adopted. The first is to heavily prune the interpretation tree, thus leaving a fraction of the original to be investigated. The second is the use of a faster machine and thus, parallel processing. The parallel matching algorithm devised by Holder and Buxton (H&B) [3] and based on work by Grimson and Lozano-Pérez [2] uses SIMD parallelism on the DAP in order to obtain a much reduced program run time. A parallel method has also been adopted by Flynn and Harris (F&H) on the Connection Machine [5].

The parallelism of H&B when compared with F&H is very effective. They found that with n model faces and m data points, n^m processing elements are required to achieve a processing time of the same order of magnitude. Also, for 250 objects, each with 10 faces and 3 data measurements, F&H required 250,000 of the 256,000 processors on the MIT Connection Machine [3]. This will rise by a factor of 10 with each additional data point as they require a separate

processor for each interpretation. A comparison of run times for the two methods using a simple chipped block model, composed of seven faces, is shown in Figure 2. The matching times are illustrated for three data points with projected run times for four and five data points. It can be seen that except for very small problems, the linear scaling of the H&B method means that it performs better.

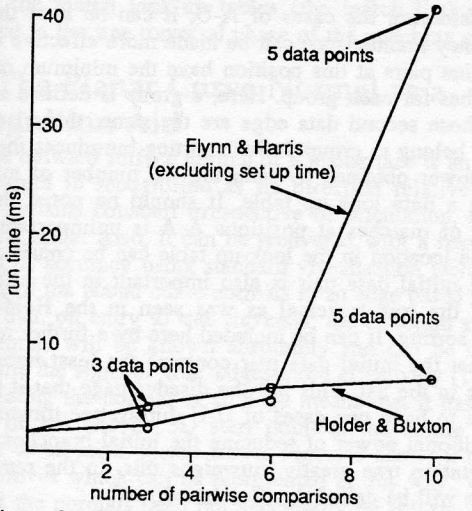


Figure 2: Comparison of H&B and F&H methods

The effectiveness of the algorithm is achieved by initially determining matches of data pairs to model pairs in parallel and then testing these for local consistency. This means that in one instruction *all* the possible pairings of a data pair to model edges can be found. The order in which the constraints are applied to the data pairs is unimportant here because we only require the result obtained from application of all the constraints. Note, however, that this is in contrast to the serial algorithm where the order of application of the pruning constraints may be significant. This is because, in many serial implementations, the constraints are applied in order of effectiveness so as to obtain cutoffs as quickly as possible. The serial algorithm, therefore, implements non-uniform termination whereas, in the parallel method, we perform an exhaustive search of the heavily pruned tree.

4.1 Logical Look-up Tables

The results obtained from application of the pruning constraints are maintained in 32 × 32 logical look-up tables. These determine possible pairings of data to model edges. From Figure 3 we can see that for the data edges 1 and 2, taken as a pair, possible matches are achieved where 'true' values 'T' are present in the matrix. Thus, data edge 1 may be associated with model edge 2 when data edge 2 is associated with model edge 1 or 3, and data edge 1 may be matched to model edge 4 when data edge 2 is on 2, 4, 5, or 7. The effectiveness of the algorithm is achieved by maintaining a large amount of data of this kind in the array store.

As well as storing the model look-up tables, the matcher is also made more effective by maintaining a one dimensional list of matches between data edges (a data look-up table) for each of the constraints. Matches between data edges are therefore calculated once only which significantly improves the performance of the matching process.

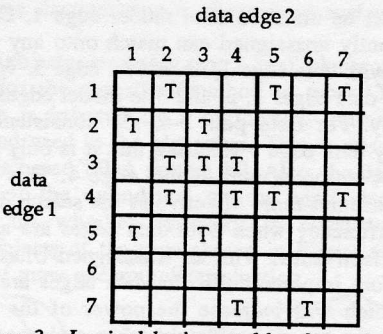


Figure 3: Logical look-up table of matches

Feasible interpretations are achieved by making trial assignments of alternative data pairs to model edges. These assignments are checked for consistency to assignments made higher up in the interpretation tree. This must be done because it is not allowed for a data edge to be assigned to more than one model edge using bottom-up matching.

4.2 Consistency Checking

In order to illustrate the serial search for consistency of assignments, we consider a list of possible matches of data pairs to model faces, as shown in Figure 4, for five data items recovered from an L-shape model of 8 faces.

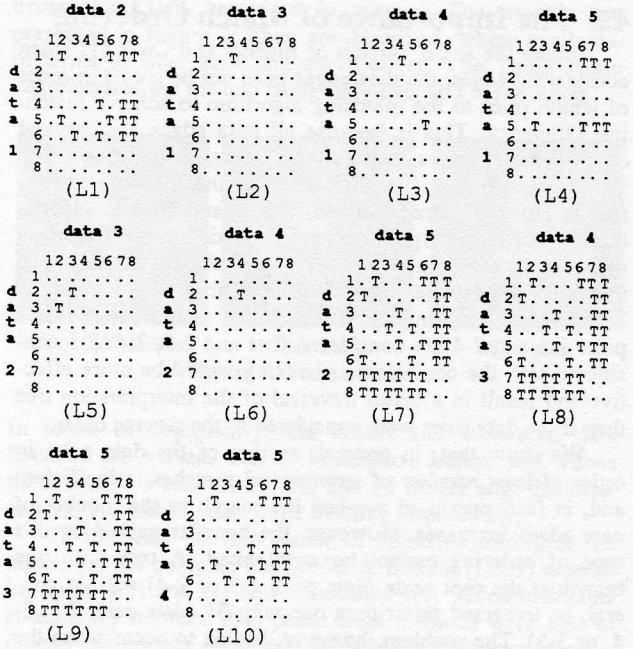


Figure 4: List of matches

The serial search algorithm requires that if a data edge is not currently assigned then it remains consistent for it to be assigned to any model edge along the appropriate row or column of the match look-up table. That is, for data pair 1-2 in the state Unassigned-Unassigned, all the 15 matches of the look-up table are consistent. Consider first the match of data edge 1 to model edge 1, and data edge 2 to model edge 2. For the level 2 look-up table (data pair 1-3), we can see that to maintain consistency with the above assignments, data

edge 1 must be matched onto model edge 1. Data edge 3, being currently unassigned can match onto any model edge consistent with data edge 1, i.e. model edge 3. We now have matches of data edges 1, 2, and 3 to model edges 1, 2, and 3 respectively. For data pair 1-4, the consistency is again enforced by data edge 1 ensuring that it is only possible for data edge 4 to match with model edge 4. This consistency checking becomes more effective as the search deepens with maximal efficiency when both data edges are assigned and minimal effectiveness with an Unassigned-Unassigned pair. It is therefore important that the data edges are sorted in a manner which will increase the power of the consistency cutoffs, i.e. the search backtracks when assignment of a data pair is not consistent with earlier assignments.

As an interpretation is an assignment of each of the data items to a model face, the model faces can be repeated in an interpretation but each data item only appears once and must be consistent with all the previous assignments of that data item in the interpretation. From the above example, two feasible interpretations result:

Data	Model
1	1
2	2
3	3
4	4
5	7

Interpretation 1

Data	Model
1	1
2	2
3	3
4	4
5	8

Interpretation 2

4.3 The Importance of Match Ordering

Unlike the serial version of Murray and Cook [1] when considering edges, the data edges need not be sorted in order of length prior to the matching algorithm to achieve matching efficiency. This is because all data edges are checked against model edges in parallel. Sorting of the data edges will therefore make no difference to the possible assignments. However, according to Holder and Buxton [4], the traversal of the interpretation tree is made more efficient by considering the data pairs in order of the least number of matches. The data pairs are thus sorted in a list accordingly. From Figure 4 it can be seen that if the assignments of data pairs 1-3 and 2-4 are considered first and 2-5, 3-4, 3-5 considered last, the consistency checking would be more effective and result in a faster traversal of the interpretation tree than if the data pairs were considered in the reverse order.

We show that, in general, sorting of the data pairs in order of least number of geometrical matches is inefficient and, in fact, proves to weaken the search as the number of data edges increases. However, the benefits gained by this type of ordering cannot be overlooked. A tree with one branch at the root node (data pairs 1-3 or 2-4) will, in general, be traversed faster than one with 34 (data pairs 2-5, 3-4, or 3-5). The problem, however, begins to occur when the ordering causes the generation of Unassigned-Unassigned data pairs during the search. If the match list was sorted and traversed according to the least number of matches, e.g.

[1-3, 2-4, 1-4, 2-3, 1-5, 1-2, 4-5, 2-5, 3-4, 3-5]

as in Holder and Buxton, then immediately, an Unassigned-Unassigned pair is produced at level 2. Fortunately, in this example, the look-up table contains one possibility, but it could have easily been more. Consistency can be maintained so that only one Unassigned-Unassigned data pair (at the top level) is present in the match ordering. This is obtained by ensuring that a data pair is encountered only after at least one of its data edges has been earlier assigned. Thus, for our example, the ordering could be:

[1-2, 1-3, 2-3, 1-4, 2-4, 3-4, 1-5, 2-5, 3-5, 4-5].

This ensures effective consistency cutoffs when traversing the match list as it now reads

[U-U, A-U, A-A, A-U, A-A, A-A, A-U, A-A, A-A, A-A]

where U = Unassigned , and A = Assigned data edge.

Considering the cases of A-U, it can be seen that the consistency maintenance can be made more effective by ensuring that pairs at this position have the minimum number of matches for each group. Here, a group is defined as data pairs whose second data edge are the same, thus, 1-4, 2-4, and 3-4 belong to group 4. Such sorting introduces the additional power obtained by reducing the number of matched pairs in a data look-up table. It should be noted that the number of matches at positions A-A is unimportant since only one location in the look-up table can be consistent.

The initial data pair is also important in the effectiveness of the tree traversal as was seen in the Holder and Buxton sorting. It can be included here by a further requirement that the initial data pair contains the least number of matches in the list. This has the disadvantage that it is now possible to have two cases of U-U during tree traversal but the additional power of reducing the initial branches of the interpretation tree greatly outweighs this. In the remaining text this will be called the 'Combined sort'.

5 Extension of the Edge Based Matcher for Planar Curves

Using stereoscopic or motion techniques, it is possible to recover creased, curved edges at surface boundaries. These may be reconstructed to form a plane which may be described by the direction of its surface normal and the perpendicular distance from the origin of the coordinate system of the environment. When considering the recognition of curved objects, it is at present difficult to recover robust matching features from say, stereopsis, due to the lack of true edges often seen (or shall we say, not seen) in primitives such as spheres, ellipsoids, and toruses. Using trinocular vision, however, it may be possible to recover the curvatures of such objects and, possibly, use these as matching components. Other curved objects, which may also be described by a quadric function, e.g. cylinders and cones, are not devoid of true edge boundaries. The ends of these primitives define a plane which, obviously, can be defined by a surface normal. It is this surface normal which we propose to use in an edge based planar curve matcher. The approach is essentially the same as in the simple polyhedral matcher with the surface normal of the primitive being treated as a true edge. Using this approach, the polyhedral algorithm may easily be extended to a face based system which will use the planar faces of the cylinder and cone primitives. Another matching feature relating to the planar ends of primitives is the axis of rotation. This can be viewed as the line joining the centres of the planar curves in the case of the cylinder and cone.

The extension of the polyhedral algorithm is explicitly directed at the class of primitives which can be defined by a developable or 'ruled' surface, i.e. cylinders and cones, although the method still holds for a more general class of planar curved objects. A developable surface is defined as belonging to the group of surfaces which has one principal

curvature equal to zero. Using such a surface, we ensure that an axis of rotation will exist.

The parallel implementation is equivalent to that described in [3] with the exception that, given a priori knowledge of match feature type, a curve edge is only permitted to pair with another curve edge. This also applies to true and axis edges. After this removal of mispaired edges, the match look-up tables (the match list) may be searched in the tree traversal phase of the matching process.

5.1 The Planar Curve Normal as a Recognition Aid

The outward surface normal of a planar face is an important feature in recognition as its direction relative to the model remains constant irrespective of orientation, translation, and scale. Also, it can be recovered with a reasonable amount of accuracy using standard visualisation techniques. The use of the planar curve normals in an edge based fashion means that, on entry to tree traversal, the direction signs of the normals will be known absolutely and can be used to determine the signs of 'true' edges more effectively [1].

Treating the normal as an edge means that it has a unit length of 1.0 and its 3D start point at the centre of the circular or elliptical planar curve. Figure 5 shows a number of primitives which can be represented in this way. We can see that the normals need not necessarily be anti-parallel.

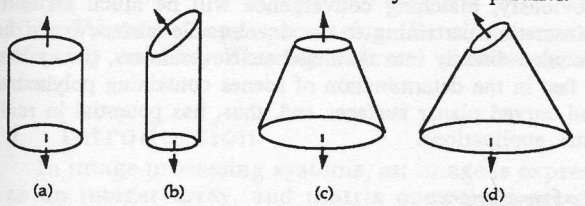


Figure 5: Normal 'edges' for various primitives

Using this methodology, the elliptical ends of sliced cylinders and cones may also be represented. In order to improve the representation of the primitives of the above figure, each surface normal is defined along with the size of the planar curve. This is merely the radius of the circle in the case of circular ends and the major and minor axes in the case of elliptical sliced ends. Using this, it is now possible to distinguish between the regular cylinder (a), and the regular cone (c), and also between the cylinder (b) and cone (d), of Figure 5. It is, however, not essential that a pair of normals be recovered during the segmentation process. A circle drawn on top of a box should give a sufficient visual cue to determine the orientation of the box.

5.2 Addition of Axis of Rotation

If the segmentation process allows, the axis of rotation of the curved primitive may be recovered and used in the matching process. Here, the axis of rotation is taken as the line joining the centres of the planar curves of the primitives. This can be used to impose additional constraints in the recognition process when we treat it as an edge. We note, however, that this matching feature does not take into account the size of the planar curves as in the case of the surface normal to the planar ends. Therefore, on its own, the regular cylinder (a) may not be immediately distinguishable from the regular cone (c) of Figure 5. This, however, does not pose a problem as in order to obtain the axis of rotation, some data relating to the curved surface must be available.

It is not strictly necessary that a 'real' planar curve must be present for the described method to work. The planar curve which is inferred by a cylindrical cut-out may be recovered and used in the algorithm. This is possible using the stereopsis based TINA system [8]. Also, the deduced axis of rotation of the cylindrical cut-out may also be used (depending on whether recovery is possible) in this approach. Objects of the type shown in Figure 6 may now be capable of more accurate determination.

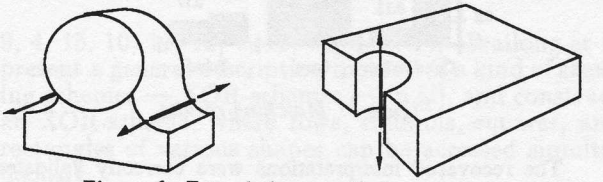
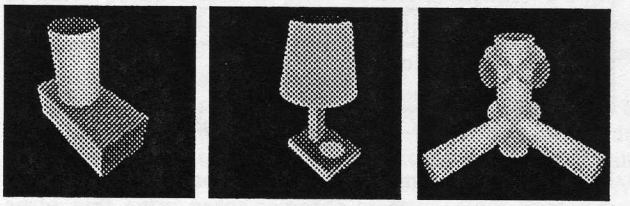


Figure 6: Extended range of planar curves

6 Experimental Results

The extended method has been demonstrated on various data scenes containing cylindrical planar curves. Results have been very promising and comparable to those obtained for true edge analysis. Here, we present results of an application of the planar curve parallel matcher on the three data scenes generated using a CAD-based parallel solid modeller [9]. The matching features were subsequently recovered using the TINA stereoscopic system. The models are presented in Figure 7. They are described, where available, in terms of true, axis, and curve edges. The Tubes scene demonstrates the method using only axis and curve edges.



(a) CylinderBox (b) Lamp (c) Tubes

Figure 7: The test scenes

In terms of complexity, the model and recovered data features are varied. The CylinderBox, Lamp, and Tubes models have a total of 15, 28, and 15 model edge features, and 12, 20, and 11, data edge features respectively.

To illustrate the importance of effective sorting of the match list, we compare the Holder & Buxton (H&B) ascending order sort and the new Combined sort which maintains consistency between data pairs. Both methods were able to deliver the correct scene interpretation with some variation in total matching time. For the Tubes scene, however, an extra interpretation was returned. Figure 8 illustrates the results obtained for the test scenes under these two methods.

As expected, the Combined sort implementation was found to outperform the simpler H&B sorting. This is despite the fact that, for the first ten match look-up tables, the CylinderBox, Lamp, and Tubes averaged 16, 20, and 14 geometric matches for the Combined sort, and 2, 2, and 3 matches for the H&B sort. The speed of execution for the Combined sort technique shows real-time application potential. Also, note that a faster matching time by over

1800 times can be obtained using the Combined sort.

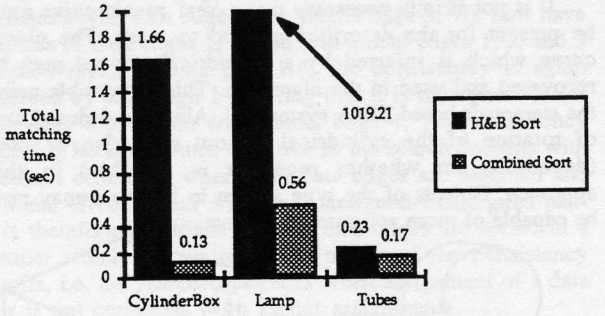


Figure 8: Matching results

The recovered interpretations were correctly validated for global consistency. However, for the Tubes scene both interpretations failed to converge due to a singular matrix leading to a failure in the Gauss-Jordan (used in estimation of the translation). The correct interpretation, however, returned a more accurate estimate of the rotation. The results obtained for the computed transformations are very accurate in view that 1) the data was non-ideal, and 2) we are using data which is derived, i.e. the surface normals and axes. Figure 9 illustrates the errors in computing the model to sensor space transformations for the test scenes.

Scene	Error in Projecting into Model Space		
	Rotation half-cone angle (°)	Δtrans. (x, y, z)	Δscale
CylinderBox	0.23°	(0.3, 0.01, 1.17)	0.0032
Lamp	0.61°	(0.46, 0.01, 1.49)	0.0024
Tubes	(correct) 1.05° (incorrect) 70.9°	Gauss-Jordan Fail	G-J Fail

Figure 9: Computed transformation errors

It can be seen that matching using the pseudo-edge features maintains the robustness observed when using true edges alone. Note, also, that the error along the z-axis (the most error prone axis using 2D image data) is surprisingly low. In order to appreciate the accuracy of the method, however, we must consider the Tubes scene containing only pseudo-edges. Here, the translational result is not available, but considering the rotation (from which it is derived), the value for the correct interpretation complies closely to the veridical rotation.

7 Concluding Remarks

We have shown that the surface normal of the circular planar ends of regular cylinders and cones can be accurately recovered using stereopsis. By treating these normals as edges along with the axis of rotation of the primitives, we were able to include them in an edge based parallel matcher based on the Grimson and Lozano-Pérez matching algorithm. From the experiments conducted, they have been shown to be robust matching features and have been used to accurately determine the model to scene space transformations. However, in the experimental tests, we have been limited by the data recovery technique to primitives containing circular planar ends. The principle is easily applicable to elliptical sliced ends. During validation, however, we must ensure that both major and minor data axes are closely fitting to the model elliptical axes. Further extensions may be made to include non-developable surfaces such as generalised cylinders but, again, we are constrained by the sensing and segmentation approaches. The generalised cylinder of Figure 10 may be included in the range of primitives if we can accurately segment its image scene such that a 'characteristic' pseudo-axis of rotation may be obtained.

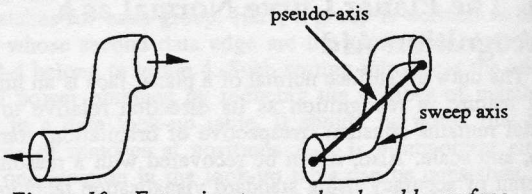


Figure 10: Extension to non-developable surface

If this is not possible, however, we may still include the primitive by merely using the normal of the planar ends.

Although the extension we have presented is simple, it has been shown to be attractive for the recognition of planar curves in terms of matching time. It also retains the robustness of the polyhedral matcher with very accurate computed model to sensor space transformations. Obviously, matching convergence will be much faster if information pertaining to the developable surface could be included directly into the matcher. Nevertheless, the system is fast in the determination of scenes containing polyhedral and curved planar surfaces and, thus, has potential in real-time applications.

References

[1] D.W. Murray and D.B. Cook, "Using the Orientation of Fragmentary 3D Edge Segments for Polyhedral Object Recognition", *Int. J. Comp. Vision*, 2:147-163, 1988.

[2] W.E.L. Grimson and T. Lozano-Pérez, "Model-Based Recognition and Localization from Sparse Range or Tactile Data", *Int. J. Robotics Research*, 3(3):382-414, Fall 1984.

[3] D. Holder and H. Buxton, "Polyhedral Object Recognition with Sparse Data in SIMD Processing Mode", *Image and Vision Comput.*, 7(1):71-78, Feb. 1989.

[4] D. Holder and H. Buxton, "SIMD Geometric Matching", *1st Euro. Conf. Comp. Vision*, 516-520, April 1990.

[5] A.M. Flynn and J.G. Harris, "Recognition Algorithms for the Connection Machine", *Proc. Ninth Int. Joint Conf. on AI*, Morgan and Kaufman, 57-60, 1985.

[6] R.V. Shankar, G. Ramamoorthy and M. Suk, "Three Dimensional Object Recognition on the Connection Machine", *Patt. Recog. Letters*, 11(7):485-492, July 1990.

[7] O.D. Faugeras and M. Hébert, "A 3D Recognition and Positioning Algorithm Using Geometric Matching Between Primitive Surfaces", *Proc. Int. J. Conf. on AI, Karlsruhe*, 996-1002, 1983.

[8] S.B. Pollard, J. Porrill and B.A. Thacker, "Tina.1 Tinatool User Documentation", *AIVRU Ref. No. 50*, University of Sheffield.

[9] M. Usoh, "A Generalised Ray Tracer Using SIMD Parallel Processing", *Technical Report No. 499*, Dept. of Comp. Sci., QMW College, Feb. 1990.