Setting the Parameters of the LFT Shape Matching Algorithm

J.S.M. Vergeest, A. Kooijman, Y. Song
Delft University of Technology
Landbergstraat 15, NL-2628 CE Delft, The Netherlands
j.s.m.vergeest@tudelft.nl

ABSTRACT

The LFT algorithm (Large Fat Tetrahedron) is used to detect congruent subsets amongst unordered point sets and forms the kernel of a partial shape matching method. Although the method yields several advantages and is relatively efficient, its performance depends highly on the choice of various geometrical threshold parameters, as *e.g.*, for the length difference of two edges. We present an overview of the key parameters of the algorithm and their influence on the computation, a guide line to provide an initial value for the parameters and we propose an approach to their automatic adjustment.

Keywords

Scan view registration, partial shape matching, fat tetrahedron, threshold parameters

1. INTRODUCTION

The LFT algorithm (Large Fat Tetrahedron) was designed to detect approximately congruent tetrahedrons in two point sets [Vergeest 2010]. If such tetrahedrons are found, they might indicate the overlapping region of partially matching shapes. The assumption was that if a large fat tetrahedron with particular dimensions occurs in point set *A*, the probability that a congruent tetrahedron is found in point set *B* is small, unless both tetrahedrons reside in the overlap region of *A* and *B*. Thus, LFTs can serve as indicators of partial shape matching. The algorithm will be briefly described in Section 2.

One important application of partial shape matching is 3D scanning of physical objects. To construct a geometric model from a physical object, multiple scan views are taken, each consisting of range data, *i.e.* 3D points representing the object's surface. Since the orientation of the object relative to the scanning device is different for different scan views, the collection of points from all scan views do not as such represent the object's surface. First the points need to be aligned to each other, that is be transformed to a common coordinate system. The process of aligning the scan views is called scan view registration. From the aligned point sets the surface of the object can be reconstructed, either fully or partially, depending on the coverage of the scan views. If the surface can be fully reconstructed, it can be assumed to represent the boundary of a volume, or solid model. Then a solid model can be derived, which can serve as input to a CAD (Computer-Aided Design) system for further modeling and processing. Basing a design on an existing object or on reuse of precedent models is an important paradigm in some industries, such as industrial design engineering. Such a method can be successful only when even occasional users of scanning devices can easily operate the system. However, the registration task is, even nowadays, still an impeding factor. In practice the user could be a stylist who has manually created a clay model of a future household device. Whereas taking the scan views of the clay model is a commonsense task to him/her, the registration of view pairs is not. The scanning system's manufacturer normally offers an interactive software package, allowing the user to designate correspondences he/she observes amongst the scan views, as to provide a starting position for a shape matching algorithm, typically based on the ICP (Iterative Closest Point) method. The user has to aligned each scan view with the set of scan views aligned previously. Generally this way of operating

the scanner is perceived as slow and tedious, both by incidental users and by trained operators.

Several approaches have been reported to the problem of partial shape matching. From here on we assume that the input data consists of unordered point sets only. That is, we will not rely on preprocesses that generate surface meshes, nor on additional information such as color, texture or material properties of the scanned object. We focus on the kernel problem of matching two point sets. Most methods make use of geometric descriptors and/or feature points. The geometric descriptor can take many forms, including moments, FFT coefficients, spin images etc [Johnson 1999]. Defining a feature using integrated quantities rather than using derivatives reduces the influence of noise [Gelfland 2005]. Another approach to diminish sensitivity to noise and data outliers is taken by [Aiger 2008]. He collects sets of 4 planar points in each of the two point clouds. If a particular set from one point cloud is approximately congruent with one from the other point cloud, a candidate corresponding pair of 4-points sets is found. If the 4-points set is relatively wide, then the method is less sensitive to noise. We refer to [Gelfland 2005] and references therein for a more extended description of registration methods.

Our approach is inspired by the 4-points congruent sets as in [Aiger 2008]. We look for 4-points sets which define a large fat tetrahedron (LFT). The assumption is that the geometry of a large tetrahedron is relatively rare and therefore can serve to detect correspondences in the two point clouds. However, since true correspondence exists in the overlap region only, an upper bound must be set to the size of the tetrahedrons. Secondly, since the number of fat tetrahedrons in point sets can be very large, a straightforward comparison of two sets of tetrahedrons (each derived from one point cloud) would not be efficient. Our algorithm derives a limited number of fat tetrahedrons from one point cloud. Then each tetrahedron is tested for being approximately congruent to any point neighborhood in the other point set. In [Vergeest 2010] we have speculated about the advantages of the LFT algorithm compared to other strategies. However, lacking a benchmark platform we cannot demonstrate this. In the next section the LFT algorithm will be briefly described. In section 3 we present the influence of parameters on the computational performance of the method. Conclusions and recommendations are given in section 4.

2. THE LFT ALGORITHM

Let two point sets A and B be given, originating from sampling of a portion of the surface of a three-dimensional object. There may exist subsets $A' \subseteq A$ and $B' \subseteq B$ such that A' and B' are samples of the same subsurface of the object. A' and B' are then said to represent an overlap region of the samples.

Let a set of sets B_i be a partitioning of B defined as follows. A three-dimensional grid is constructed, aligned with a bounding box of B. The grid has the size of the bounding box of B. The block-shaped grid elements, or cells, all have the same size and have index i, $i = 1, ..., G$, where G is the number of cells of B. Each cell encloses zero or more points of B. Each point of B is enclosed by exactly one cell. B_i is the set of points enclosed by cell indexed i.

Let A' and B' be the largest overlap of A and B, informally defined as follows. Assuming that A and B are range images of a physical object, let S_A and S_B informally be defined as the portions of the surface of the physical object represented by A and B, respectively. Then, both A' and B' represent all or some part of the physical overlap surface $S_A \cap S_B$. Depending on the extent of $S_A \cap S_B$, A' and B' each may contain zero up to as many points as the cardinality of A and B, respectively.

Let $B'_i = B_i \cap B'$, that is the portion of cell B_i coinciding with the overlap region. Our search strategy is based on the assumption that the overlap region is connected and has the extent of at least the size of a cell. In such cases there might exist sets B_i containing multiple points of B_i'. A property of any point of B' is that its Euclidian distance to A is relatively small, provided that A and B are defined in the same coordinate system. However, since A and B originate from independent sampling processes, they will in general be defined in different coordinate systems. The difference between the two coordinate systems can be described by a rigid body transformation M, such that MB and A are defined in the same coordinate system, where MB is the set of points of B to which transformation M has been applied. We name this transformation the *matching transformation*.

Since neither the overlap region, nor the matching transformation M are known, we determine which of the sets B_i is fully or partly contained in B'. We do so by constructing the largest and fattest tetrahedron in each cell and test each such LFT against congruency with 4-points sets of A.

2.1 Finding the LFT

When a small set of points of B_i is close to A *and* if these points are sufficiently non-planar, then the transformation to match this set with A is a relatively good candidate of the M we are looking for. Relying on this principle we base the algorithm on matching 4 points to A, where the 4 points are contained in the same cell. The 4 points, denoted v_1, v_2, v_3, v_4, are

selected from B_i such that they form an LFT as follows: v_1 and v_2 are the points in B_i which are furthest apart. v_3 is the point in B_i furthest from the line through v_1 and v_2, that is it maximizes $|(v_3 - v_1) \times (v_2 - v_1)|$. v_4 is the point in B_i furthest from the plane defined by v_1, v_2 and v_3, that is it maximizes $|((v_3 - v_1) \times (v_2 - v_1)) \cdot (v_4 - v_1)|$. An example of an LFT is shown in Figure 1.

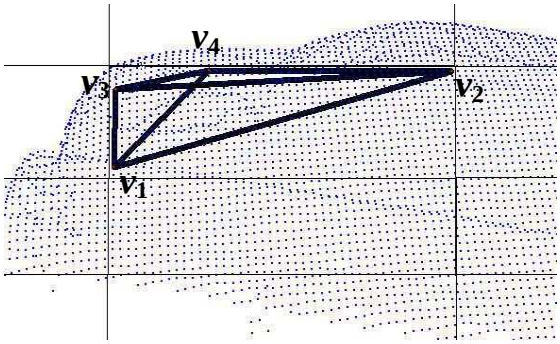

Figure 1. Point cloud *B*, its cell structure and the LFT contained in a cell. Data are from the Stanford Bunny [Stanford 2010].

2.2 Calculating the transformation *M*

Once an LFT has been determined in a particular cell of point cloud *B*, we look for potential corresponding 4-points sets in *A*. As mentioned, when the LFT resides in the overlap region of shapes *A* and *B* then there should exist 4 points in *A* representing a tetrahedron with dimensions equal to those of the LFT, that is up to some precision since the point sets *A* and *B* are obtained as independent measurements. The deviation between "corresponding" points can be expected to be as large as half of the scan spacing practiced. We have applied two methods to detect (approximately) congruent 4-points sets in *A*. We look for points a_1, a_2, a_3 and a_4 in *A* such that there exists a transformation *M* with $Mv_i \approx a_i$ for i=1,...4. The edge lengths of the LFT are denoted $l_{ij} = |v_i - v_j|$.

Method 1

For each point in *A*, name it a_1

Translate the LFT such that v_1=a_1

Search for points in *A* at distance l_{12} from a_1 and name them a_2

For each such a_2

Search for points in *A* at distance l_{13} from v_1 and at distance l_{23} from v_2 and name them a_3

For each such a_3

Rotate the LFT about v_1 such that v_2 gets closest to a_2

Rotate the LFT about axis (v_1, v_2) such that v_3 gets closets to a_3

If then v_4 is close to any point in *A* (called a_4) the accumulated transformations so far applied to the LFT represent a candidate *M*

End of method 1

Alternatively we can explicitly test congruency by comparing the six edge lenths of the LFT to the corresponding distances between candidate points a_1, a_2, a_3, a_4. We define δ_1 as the threshold value for length comparisons.

Method 2

For each point in *A*, name it a_1

For each point in *A*, name it a_2

If $|\text{dist}(a_1, a_2) - l_{12}| < \delta_1$

For each point in *A*, name it a_3

If $|\text{dist}(a_1, a_3) - l_{13}| < \delta_1$ and $|\text{dist}(a_2, a_3) - l_{23}| < \delta_1$

For each point in *A*, name it a_4

If $|\text{dist}(a_1, a_4) - l_{14}| < \delta_1$ and $|\text{dist}(a_2, a_4) - l_{24}| < \delta_1$ and $|\text{dist}(a_3, a_4) - l_{34}| < \delta_1$

Compute *M* from the v_i and a_i.

End of method 2

In method 2 the calculation of the transform *M* is postponed until the congruency is fully checked. One way to obtain *M* (as we implemented it) is to concatenate the translations and rotations in exactly the same way as done in method 1; see [Vergeest 2010] for the explicit equation. The two sets (v_1, v_2, v_3) and (a_1, a_2, a_3) are sufficient to determine *M*, but they do not lead to a set of linear equations with a unique solution since the congruency is approximate only. However, a solution based on minimizing $\Sigma |Mv_i - a_i|^2$ would be feasible and accurate (not implemented).

2.3 Computing the degree of overlap

Typically thousands of candidate *M* transforms are found, depending on threshold δ_1. If δ_1 is increased the number of candidates will rise steeply, as discussed later. We need to test whether or not a particular *M* is the matching transform. If an LFT *L* is contained in *B´* then the directed Hausdorff distance of *ML* to *A* will be (by definition) small if *M* is the matching transformation. It can be expected that then a significant portion of the points MB_i (from the current cell) will be close to *A* as well. Conversely, when many points MB_i appear close to *A* the probability that *M* is the matching transform is large.

In our algorithm, all points in cell B_i are subjected to M and their distance to A is determined. If a sufficient fraction f of the points are closer than δ_2 to A then M is saved as a candidate transformation. As a final step, each of the candidate transformations is used to compute the set MB, involving all points of B. The degree of matching of B to A is defined as N, the number of points in MB closer than δ_3 to A. The transformation producing the largest N is the outcome of the method.

3. PERFORMANCE AND PARAMETERS

As reported in [Vergeest 2010] the algorithm has been successfully applied to practical scan view registration. A typical CPU time of partial shape matching was 500s, which could be reduced to about 10s in a CUDA-GPU implementation [Kooijman 2009].

We have now studied the influence of the parameters δ_1, δ_2, δ_3 and f on the computational performance of the algorithm for method 2. The granularity of the subdivision into G cells is also of influence to the algorithm. Not all cells produce an acceptable LFT. We have set a lower limit to the number of points from B contained in a cell; if the cell contains too few points we do not consider it. If a particular LFT is too small or too thin, it is discarded. Therefore, typically 10% of the cells produce an LFT for further processing. We focused on method 2 since its implementation is relatively simple and it will be compared to its CUDA implementation in the near future. An upper bound of the complexity of the algorithm is $C \propto GP^6$, where P is the number of points occurring in a scan view (we assumed that A and B are of comparable size). In the search process, for each cell, each point in A is visited at least twice in order to form the line segment a_1a_2. When the test against l_{12} is passed, another loop over all elements of A is made to find candidates a_3 and finally one more loop to find a_4 (the maximal cost is proportional to GP^4 so far). If a_4 is found, then all points in the cell are compared to A (cost proportional to P^2) and possibly another check of all B against A is performed, as described in section 2.3. The main terms of the cost C are:

$$C \propto P^2 + \beta_1 P^3 + \beta_2 P^4 + (\beta_3 + \beta_4)P^6. \qquad (1)$$

β_1 is the probability that the l_{12} test is passed, β_2 is proportional to the probability that the l_{13} and l_{23} test are passed and β_3 is proportional to the probability that the l_{14}, l_{24} and l_{34} tests are passed. β_4 depends on parameter f and the degree of overlap found of points in the current cell with A.

If we would set δ_1=0 then no LFT would practically pass the l_{12} test and terms 2 and 3 would vanish, or $\beta_1 = \beta_2 = \beta_3 = \beta_4 = 0$. If aforementioned fraction f and all δ_i are large then C behaves like an 6th power function of P for large P; the number of candidate M transforms would be very large and many of them have to be checked by Hausdorff twice, namely once involving MB_i and possibly once more involving MB.

To achieve efficient partial shape matching the algorithm should detect the matching M (therefore, the parameters should not be too small), without superfluous candidates (therefore, the parameters should not be too large).

To gain insight in the effects of the parameters we have performed numerical tests on one particular set of A and B, with cardinality 3188 and 2407, respectively. These sets are down-sampled versions of the Bunny data from the Stanford Scan Data Repository [Stanford 2010]. The data points are relatively evenly spaced, about 2mm apart, on a surface with a diameter of approximately 150mm.

Table 1 gives an impression of the computational expense of the algorithm for Method 2. The 10 runs differ only by the parameter δ_1. The cell division was $G = 5\times5\times5 = 125$ equally sized blocks of 24×31×27mm. Out of these, 77 were empty and 7 cells contained an LFT that was sufficiently large and fat. Out of these 7, only one LFT appeared to be located in the overlap region and could produce the correct matching transform. This particular LFT had l_{12} = 40.6mm and a base triangle of height 20.2mm (that is the distance of point v_3 to edge v_1v_2) and thickness 3.1mm (distance of v_4 to the base). The algorithm includes threshold parameters for thickness to accept LFTs and also for the minimum number of points contained in cells that are considered as carrier of an LFT. For the runs of Table 1, the limitation parameters were chosen 3.0mm for height and 96 for the number of points in a cell. The latter number was calculated as $n_B / G^{2/3} = 2407/25 \cong 96$, which reflects the fact that the data points represent a dimensionality 2 boundary rather than a volumetric content. The cell located in the overlap region contained 100 points. We have set fraction parameter f = 0.9. If 90% or more of the cell points after transformation got closer than δ_2 to any point of A then the LFT yielded "cell OK" in the table. δ_2 was set to 1.5mm. The threshold for computing the overlap explicitly was set to δ_3 = 1.5mm. When the correct overlap (and thus the correct matching transform) was found, the associated LFT was classified as "all OK".

From the table we see that δ_1 = 0.67 is approximately the lowest threshold for which at least one of the 7 LFTs produced the correct matching transform M. The number of points from MB closer than δ_3 = 1.5mm to A was 540 (22.4%), reflecting the size of the overlap region (we were able to judge the

n_A = 3188, n_B = 2407, G = 125, LFTs = 7, δ_2 = 1.5mm, δ_3 = 1.5mm, f = 0.9									
δ_1 (mm)	a_{12}	a_{12} OK	a_{123}	a_{123} OK	a_{1234}	a_{1234} OK	cell OK	all OK	CPU(s)
0.00	7.1×10^7	0	0	0	0	0	0	0	2
0.15	7.1×10^7	2.0×10^5	6.3×10^8	6.5×10^3	2.0×10^7	10	0	0	18
0.30	7.1×10^7	4.0×10^5	1.3×10^9	5.1×10^4	1.7×10^8	438	0	0	40
0.45	7.1×10^7	5.9×10^5	1.9×10^9	1.8×10^5	5.6×10^8	4345	2	0	95
0.60	7.1×10^7	7.9×10^5	2.5×10^9	4.2×10^5	1.3×10^9	22,900	4	0	330
0.66	7.1×10^7	8.7×10^5	2.8×10^9	5.6×10^5	1.8×10^9	36,638	6	0	410
0.67	7.1×10^7	8.9×10^5	2.8×10^9	6.9×10^5	1.9×10^9	45,274	10	1	460
0.75	7.1×10^7	9.9×10^5	3.2×10^9	8.2×10^5	2.6×10^9	83,868	15	2	780
1.05	7.1×10^7	1.4×10^6	4.4×10^9	2.3×10^6	7.2×10^9	5.8×10^5	64	4	4800
1.50	7.1×10^7	2.0×10^6	6.3×10^9	6.6×10^6	2.1×10^{10}	4.5×10^6	371	26	36,700

Table 1. Complexity terms of length tests by Method 2 as function of threshold δ_1. Data are from the Stanford Bunny.

correctness of the transformation of these particular scan views based on ground knowledge from alternative shape matching methods). Out of the 100 points in the particular cell containing the "correct" LFT, 97 were at close distance to *A* (within threshold δ_2 = 1.5mm).

Obviously, the 4 vertices $v_1...v_4$ of the LFT itself belong to that set of 97 points, since they are close to the points $a_1...a_4$. For the same LFT one different 4-points set in *A* was found leading to a transformation passing "Cell OK". With that transformation 91 points of the cell were close to *A* but so were only 159 points of *B*, making it very unlikely that the transformation was the matching transform. Another LFT generated 8 transforms passing "Cell OK" but turned out not the matching transform. The total of 10 cases of "Cell OK" is depicted in Figure 2. The number of overlap points in *B* is plotted against the rotation exerted by the transform. The latter quantity was chosen as it characterizes the transform, although there is, of course, not a one-to-one relationship between the angle and the transformation matrix. The orientation of scan view *B* relative to *A* is 89.9 degrees, according to the solution found at δ_1 = 0.67. The number of l_{12} tests is, independently of δ_1, equal to n_A^2 times the number of LFTs considered. For the runs of Table 1 this number, labeled a_{12}, was $3188^2\times7 = 7.1\times10^7$. For δ_1 = 0.67, 1.3% of the l_{12} comparisons passed the test, labeled "a_{12} OK". This percentage is collective over all accepted LFTs in the run. The number of (l_{13}, l_{23}) tests is n_A times the "a_{12} OK" cases, or 2.8×10^9. This number depends on the third power of n_A. Further, β_1 in equation (1) is dependent on δ_1.

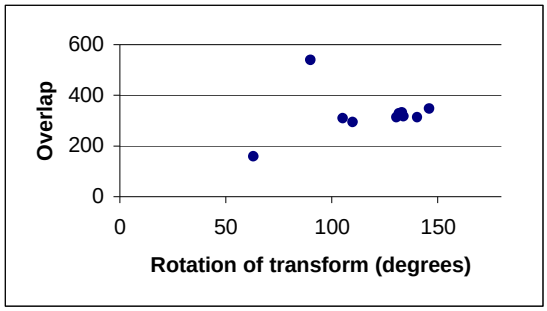

Figure 2. Number of points in *MB* close to *A* versus rotation of *M* for δ_1 = 0.67 and *f*=0.9. The plot contains 10 data points.

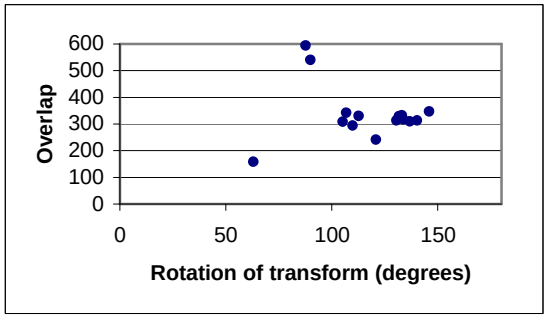

Figure 3. Number of points in *MB* close to *A* versus rotation of *M* for δ_1 = 0.75 and *f*=0.9. The plot contains 15 data points.

For the particular runs in Table 1 it turned out that β_1 = 0 for $\delta_1 \le 0.66$ and β_1 is rising for increasing δ_1. For δ_1 = 0.67, 0.02% of the a_{123} tests was positive. This percentage is proportional to β_2 and depends on δ_1. From the 1.9×10^9 a_{1234} tests, 0.002% or 45,274 resulted positively. This fraction affects β_3. Finally, the number of exhaustive tests of distance *MB* to *A* depends on the fraction of "cell OK" (0.02% in the run for δ_1 = 0.67). This fraction is proportional to β_4.

The increase of computation time with increasing δ_1 is obvious from the rightmost column of Table 1. The choice of δ_1 seems most critical, whereas δ_2 and f affect the number of distance evaluations, which will be increasingly critical for large n_A and n_B.

The effect of changing f from 0.9 to 0.8 is shown in Figure 5, which should be compared to Figure 3. The number of candidates passing the f threshold rises from 15 to 308. Instead of having only two correct transformations for f=0.9, there are 4 of them for f=0.8. They show up as a narrow peak in Figure 5 near 90°.

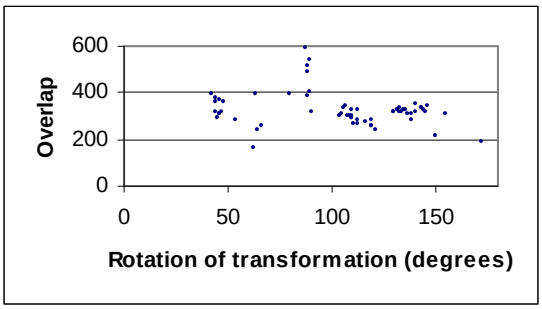

Figure 4. Number of points in *MB* close to *A* versus rotation of *M* for δ_1 = 1.05 and f = 0.9. The plot contains 64 data points.

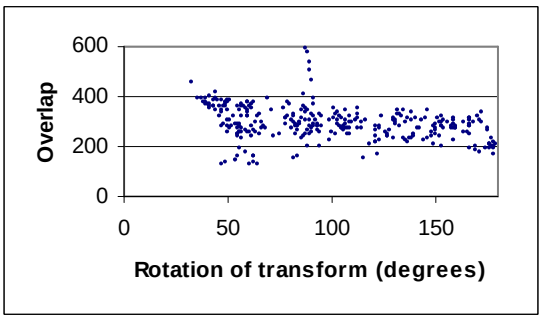

Figure 5. Number of points in *MB* close to *A* versus rotation of *M* for δ_1 = 0.75 and f = 0.8. The plot contains 308 data points.

Lowering f to 0.5 does not lead to more correct transforms, but increases the background of incorrect candidates (Figure 6). In Table 2 the number of candidates and the performance of the algorithm, for the different values of f, are presented. It should be noted that coefficient β_4 increases with decreasing f, leading to 6th power behavior of the complexity, see equation (1). The increase of CPU time in Table 2 can be practically completely attributed to the number of full distance computations. When the number of tested cells increases from 308 to 8695 (factor 28.2), the CPU time for *B*-to-*A* distance computation increases by factor (2114-780) / (829-780) = 27.2. In all cases the number of cell-to-*A* distance computations is 83,868, which would increase with the 6th power of number points as well. However, the β_3 coefficient is small when δ_1 is moderate and the number of points in a cell is small.

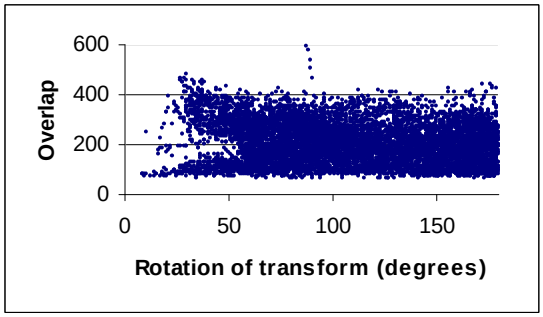

Figure 6. Number of points in *MB* close to *A* versus rotation of *M* for δ_1 = 0.75 and f = 0.5. The plot contains 8695 data points.

n_A = 3188, n_B = 2407, G = 125, LFTs = 7, δ_1 = 0.75mm, δ_2 = 1.5mm, δ_3 = 1.5mm				
f	a_{1234} OK	cell OK	all OK	CPU(s)
0.9	83,868	15	2	780
0.8	83,868	308	4	829
0.5	83,868	8695	4	2114

Table 2. Dependence on f of the performance of Method 2.

We observed that increasing δ_1 and/or decreasing f puts a dramatic load on the computation. Further, having approximately 1.6 times more unordered points in the sets, rises the CPU times by a factor 20.8, see Table 3 for details. For a pure 6th order behavior one would expect a factor 16.8, but there are other factors such as the number of accepted LFTs, which changed from 7 to 9.

4. CONCLUSIONS

The two most critical factors influencing the performance of the algorithm are the point set sizes and δ_1. In the test runs we have used down-sampled versions of point sets containing originally 40,200 and 30,400 points. The down-sampling algorithm removed points conservatively from the set such that no two points were less than ε length units apart, where ε was set to 2mm for the runs in Tables 1 and 2, and ε = 1.5mm for the runs in Table 3. For given ε an upper limit of δ_1 would be 0.5ε in a worst-case one-dimensional setting. Using a small value for δ_1 could lead to zero LFT matches.

n_A = 3188, n_B = 2407, G = 125, δ_1 = 0.75mm, δ_2 = 1.5mm, δ_3 = 1.5mm, f = 0.9									
n_A, n_B, LFTs	a_{12}	a_{12} OK	a_{123}	a_{123} OK	a_{1234}	a_{1234} OK	cell OK	all OK	CPU(s)
3188, 2407, 7	7.1×10^{7}	9.9×10^{5}	3.2×10^{9}	8.2×10^{5}	2.6×10^{9}	83,868	15	2	780
5140, 4127, 9	2.4×10^{8}	3.1×10^{6}	1.6×10^{10}	4.0×10^{6}	2.1×10^{10}	737,044	703	40	16,240

Table 3. Effect of increasing the density of the point sets by approximately a factor 1.6 for Method 2.

Empirically we have found δ_1=0.66mm=0.33ε as an upper limit in the particular setting of the runs we performed. This could be considered as a rule of thumb for δ_1, although it presumes evenly spaced points. The choice of G has turned out to be critical as well. When we selected G=6×6×6=216, none of the LFTs yielded a correct transform, unless we increased δ_1 from 0.67 to 1.05mm. Indeed, refining the cell subdivision could exceptionally imply that *fewer* cells are fully included in the overlap region. When we lowered G to 4×4×4=64, none of the LFTs yielded a correct transform, not even at δ_1 = 1.05. This could have been expected since the LFTs all exceed the size of the overlap region and are therefore unlikely to fit "correctly" to A at any place.

The subdivision has been implemented on an arbitrarily orientated evenly spaced grid, namely a grid aligned with the global coordinate system. This subdivision method could be improved significantly to reduce the number cells that should be considered further. If we assume that the overlap region contains at least 20% of the points of B then the 5×5×5 subdivision seems appropriate.

The values of δ_3 and δ_4 are less critical, *provided* that δ_1 is not unnecessarily large. The values we supplied (1.5mm or 0.75ε) seem reasonable. f=0.9 turned also a good starting value; the correct transforms yielded about 95% overlap of the cell with A, and we observed that even with increased δ_1, it was not useful to set f lower than 0.9.

As mentioned, these recommendations for initial parameter values are still case-specific. When the scanning process would result in very unevenly distributed points, the parameters should be derived from the highest occurring ε.

A possible strategy for automatically adjusting the parameters is to set ε relatively large (*e.g.* 0.01 times the diagonal of the bounding box of A) and perform a run with δ_1 = 0.3ε. Then the largest overlap detected can be tested for compactness and connectedness. If the distribution of the overlapping points is not consistent with a connected portion of A and/or B, runs with increased δ_1 can be carried out.

Both the length tests and the distance computations can be implemented with a good degree of parallelization. Unlike purely two-dimensional processes such as image restoration, 3D scan view data cannot be subdivided in portions which can be processed completely independently. Still an acceleration of the computation by a factor of 10 to 100 appears feasible in Cuda implementations that are presently under investigation.

REFERENCES

[Aiger 2008] Aiger, D., Niloy, M., Cohen–Or, D. 2008. 4–Points Congruent Sets for Robust Pairwise Surface Registration. ACM Trans. Graph. 27, 3.

[Johnson 1999] Johnson, A.E. and M. Hebert, "Using spin-images for efficient multiple model recognition in cluttered 3-D scenes," IEEE Trans. Pattern Anal. Mach. Intell., vol. 21, no. 5, pp. 433–449, 1999.

[Gelfland 2005] Gelfland, N., Mitra, N. J., Guibas, L. J., Pottmann, Robust global registration. In Proc. Symp. Geometry Processing, Euro-graphics, pp 197–206.

[Kooijman 2009] Kooijman, A., Vergeest, J.S.M., A GPU-supported approach to the partial registration of 3D scan data. Proceedings of the Gravisma 2009, D. Hildenbrand and V. Scala (Eds), pp 1-5.

[Stanford 2010] Stanford Scan data Repository, http://graphics.stanford.edu/data/3Dscanrep

[Vergeest 2010] Vergeest, J.S.M., Kooijman, A., Song, Y., Partial 3D shape matching using large fat tetrahedrons. Journal of WSCG, Vol. 18, Nr. 2, pp 41-47, 2010.