reliable starting vectors for curve calculation. So far we did not try any algorithm to handle these patch pairs; we just kept their diameter ε smaller than the display precision and treated these pairs as single points of the intersection. Since the numerical derivative information is not sufficient to handle these cases, there is some need for derivative–free algorithms around such degenerations.

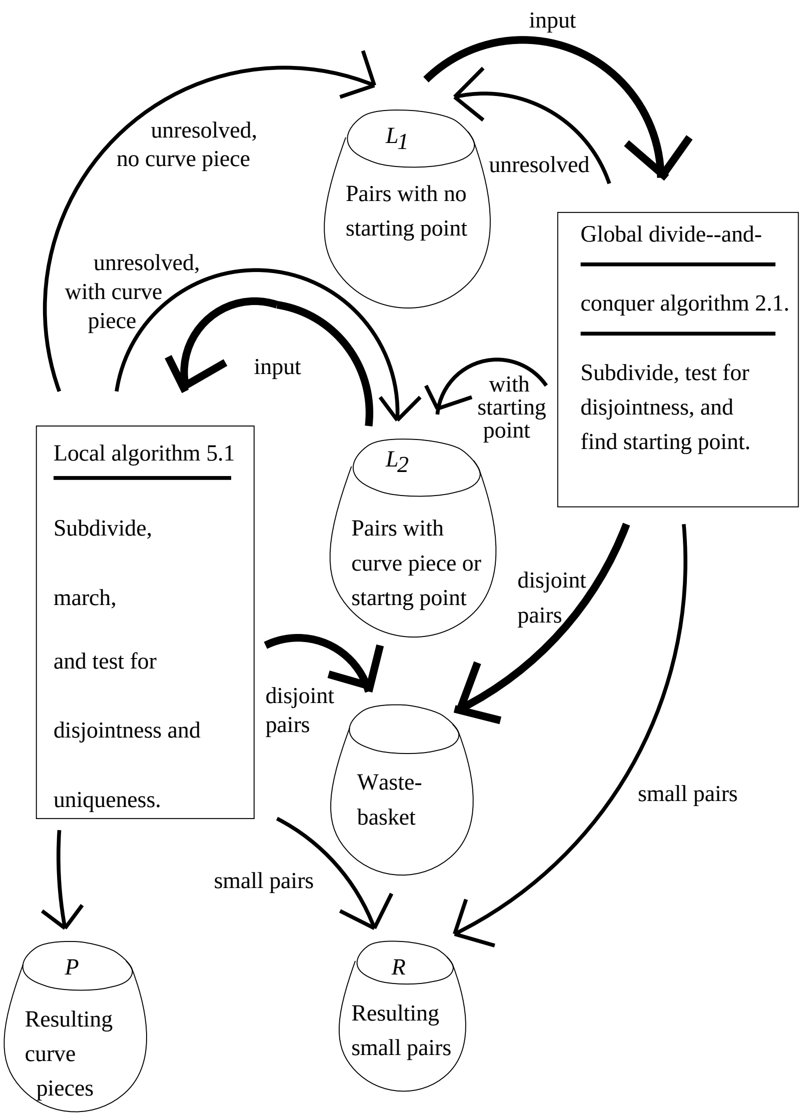

Figure 4 Data flow within multistage algorithm.

Table 2 contains data for an example with two intersection curves that have distance $\approx 10\varepsilon$. The local behavior of the intersection curves was similar to that in Figure 2 in the previous section, but the approximate singularity was so severe that the marching algorithm could not proceed along the bottleneck as it did in Figure 2. The notation of Table 2 is the same as for Table 1, but the new column $\mathcal{P}$ denotes the number of curve points generated by the marching method, while the column labeled "unique" gives the number of patch pairs which could be discarded because of being contained in uniqueness balls. The iteration cycle of the multistage method performed

Example 2		Oriented parallelepipeds					
Iterations / Routines		$\mathcal{L}_1$	$\mathcal{L}_2$	$\mathcal{P}$	$\mathcal{R}$	unique	disjoint
0	Subdivision and Boxing	39	–	–	–	–	187
1	Subdivision and Boxing	52	–	–	–	–	572
	Marching method	44	8	520	–	–	–
	Uniqueness criterion	44	128	–	0	0	0
	Test of box diameter	44	–	–	0	–	–
2	Subdivision and Boxing	45	–	–	–	–	659
	Marching method	45	128	0	–	–	–
	Uniqueness criterion	45	512	–	0	0	96
	Test of box diameter	45	–	–	0	–	–
3	Subdivision and Boxing	27	–	–	–	–	693
	Marching method	27	512	0	–	–	–
	Uniqueness criterion	27	2960	–	0	22	305
	Test of box diameter	27	–	–	0	–	–
4	Subdivision and Boxing	14	–	–	–	–	418
	Marching method	14	2960	0	–	–	–
	Uniqueness criterion	14	5648	–	0	171	2436
	Test of box diameter	14	–	–	0	–	–
5	Subdivision and Boxing	14	–	–	–	–	210
	Marching method	14	5648	0	–	–	–
	Uniqueness criterion	14	5520	–	0	555	4748
	Test of box diameter	14	–	–	0	–	–
6	Subdivision and Boxing	19	–	–	–	–	205
	Marching method	19	5520	0	–	–	–
	Uniqueness criterion	19	4800	–	0	713	4507
	Test of box diameter	19	–	–	0	–	–
7	Subdivision and Boxing	13	–	–	–	–	291
	Marching method	13	4800	0	–	–	–
	Uniqueness criterion	13	544	–	0	1102	3664
	Test of box diameter	13	–	–	0	–	–
8	Subdivision and Boxing	13	–	–	–	–	195
	Marching method	13	544	0	–	–	–
	Uniqueness criterion	13	0	–	0	273	271
	Test of box diameter	0	–	–	13	–	–

Table 2 Numbers of patch pairs treated by the multistage method

a sweep over “old” bags $\mathcal{L}_1$ and $\mathcal{L}_2$ by first the divide–and–conquer algorithm followed by the marching algorithm, generating new bags $\mathcal{L}_1$ and $\mathcal{L}_2$ for the next iteration. At this point we warn the programmer to implement bags in plain C without a garbage collector. Generating new bags as files or arrays by each iteration overcomes this problem in a straightforward but rather primitive way. For sequential machines, the bookkeeping of subdivided patches should better be implemented by a quadtree structure as in [Barnhill/Kersey '90], while parallel machines should do local bookkeeping in quadtrees and should exchange “bag data” only if necessary for load balancing, using advanced “hot potato” routing techniques.

Note that the intersection curves of this example, as far as they were traceable at all, are already obtained in the first iteration. The rest of the work is spent to guarantee that nothing else is overlooked, which is a nontrivial task in this example. The bag $\mathcal{L}_1$ of the divide–and–conquer method stays rather small and ends up with 13 undecidable patch pairs moved into $\mathcal{R}$ when getting small enough. This part of the algorithm copes with the singularity by recursive subdivision, and it does so quite effectively, keeping the number of non–discarded patch pairs approximately constant during subdivision. The bulk of the work consists of the application of the uniqueness criterion to subdivided patches containing intersection curves. It blows the bag $\mathcal{L}_2$ up until the subdivided patch pairs are small enough to be contained in uniqueness balls, if they are not already detected as being disjoint. Of course the algorithm is much faster when the solution branches are further apart. The hazardous case considered here necessarily leads to very small uniqueness boxes and many subdivision steps.

If safety is not required, the user may simply omit the last box in Algorithm 5.1. Then the algorithm would need only two iterations in the above example. The user can adapt the algorithm to any safety restriction by executing the last block only for certain subdivision levels.

6 Conclusions

Our numerical and theoretical experience with the multistage algorithm supports the following general statements:

- Disjoint patch pairs should be detected and discarded as soon as possible. This means that after each subdivision step there should be an immediate disposal of the garbage.
- Oriented parallelepiped boxes are superior for problems with medium or high accuracy requirements (see Theorem 3.1).
- One should apply a local marching method as soon as possible (see Theorem 4.1). This means that candidates for starting points should be calculated very early.
- The divide–and–conquer method cannot safely and efficiently handle patches with a known intersection curve piece and an undetected second curve piece nearby. Some additional local uniqueness argument around a curve must be numerically exploited to deal with such patches. Thus uniqueness boxes as provided by our marching algorithm are a useful tool to get more safety by exclusion of further solution branches around existing ones.
- All the degenerations and peculiarities that can possibly arise for marching methods should be handed back to a global divide–and–conquer method. This does not lead to

excessive computational problems, provided that the divide–and–conquer method obeys the first of these rules.

- As long as doubtful cases are handed back to the divide–and–conquer method, there is no need for utmost generality in the programming of the other routines. It suffices to make them work safely under well-defined restrictions on their inputs, for instance under smoothness, regularity, and transversality requirements for the intersection curves. In this sense the divide–and–conquer method makes the multistage algorithm stable and safe even for wildly degenerate cases, while the marching method, when started early enough, makes it fast on nondegenerate intersection curves.

References

[Barnhill et.al. '87] R. E. Barnhill, G. Farin, M. Jordan, B. R. Piper: *Surface/surface intersection,* Computer Aided Geometric Design 4, 3–16, 1987

[Barnhill/Kersey '90] R. E. Barnhill, S. N. Kersey: *A marching method for parametric surface/surface intersection,* Computer Aided Geometric Design 7, 257–280, 1990

[Bürger '92] H. Bürger: *Ein Mehrphasenalgorithmus für Parallelrechner zur Bestimmung von Schnittkurven zweier Flächen,* Ph.D. dissertation, Göttingen 1992

[Diener/Schaback '90] I. Diener, R. Schaback: *An Extended Continuous Newton Method,* J. of Optimization Theory 67, 87-107, 1990

[Dokken et.al. '85] T. Dokken, V. Skytt, A.–M. Ytrehus: *Recursive subdivision in iteratios and related problems,* Mathematical Methods in Computer Aided Geometric Design, T. Lynche and L. L. Schumaker (eds.), 207–214, Academic Press, Boston, 1989

[Farin '90] G. Farin: *Curves and surfaces for computer aided geometric design, a practical guide,* Academic Press, Inc., New York, 1990

[Houghton et.al. '85] E. G. Houghton, R. F. Emnett, J. D. Factor, C. L. Sabharwal: *Implementation of a divide–and–conquer method for intersection of parametric surfaces,* Computer Aided Geometric Design 2, 173–183, 1985

[Hoschek/Lasser '89] J. Hoschek, D. Lasser: *Grundlagen der geometrischen Datenverarbeitung,* Teubner, Stuttgart, 1989

[Müllenheim '90] G. Müllenheim: *Convergence of a surface/surface intersection algorithm,* Computer Aided Geometric Design 7, 415–423, 1990

[Müllenheim '91] G. Müllenheim: *On determining start points for a surface/surface intersection algorithm,* Computer Aided Geometric Design 8, 401–408, 1991

[Ortega/Rheinboldt '70] J. M. Ortega, W. C. Rheinboldt: *Iterative solution of nonlinear equations in several variables,* Academic Press, Inc., New York, 1970

[Pratt/Geisow '86] M. J. Pratt, J. Geisow: *Surface/surface intersection problems,* The Mathematics of Surfaces, J. A. Gregory (ed.), 117–142, Clarendon Press, Oxford, 1986

[Schaback '89a] R. Schaback: *Mathematische Grundlagen des Computer Aided Design,* Lecture Notes, Göttingen, 1989

[Schaback '89b] R. Schaback: *A local tracing method for computing intersection curves of surfaces in computer aided design,* Notes from a seminar talk, 1989

[Schaback '92] R. Schaback: *Error Estimates for Approximations from Control Nets,* to appear in Computer Aided Geometric Design

[Sederberg/Meyers '88] T. W. Sederberg, R. J. Meyers: *Loop detection in surface patch intersections,* Computer Aided Geometric Design 5, 161–171, 1988

Author's addresses:

Dr. H. Bürger
Prof. Dr. R. Schaback
Institut für Numerische und angewandte Mathematik
der Universität Göttingen
Lotzestraße 16–18

D–3400–Göttingen

Germany