DYNAMIC TOLERANCE SETTING METHOD FOR PLANE SWEEP ALGORITHM

Xiaomin Zhu[a,b], *Hongchao Zhao*[a], *Jinyun Fang*[a]
[a]Institute of Computing Technology, Chinese Academy of Science, Beijing, China, 100190
[b]Gradute School of Chinese Academy of Science, Beijing, China, 100049

Plane sweep algorithm is a basic algorithm in Computational Geometry. The kernel of this algorithm needs such a process – getting the segments passing the current event point. In theory, we can compute whether the area value of the triangle (shown in figure 1) constructed by the event point and the two vertices equals 0.0.

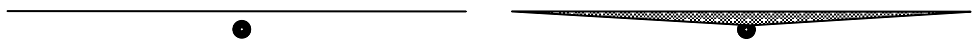

Figure 1. A simple method to get point to segment relation

As the coordinates of the points are usually stored as floating point numbers, and errors may occur in the computing process, so that in real implementation, we should set the tolerance value properly. Currently, some papers of this topic either just give an implementation in theory, or just set a static value (epsilon), or compute with high precision, or use rational coordinates. High precision and rational coordinates need extra time consuming, and static epsilon may just be appropriate for some types of data—for the real world data taking latitude and longitude as the units and the data taking meters or kilometers as the units, different tolerance values are needed because of their different value regions. How to set the tolerance value dynamically is the motivation, and this paper’s method solves the problem.

Most programming environment adopts the IEEE-754 standard of floating point computation. Generally, for simple computation, the add/subtract operations propagate absolute error, and multiply/divide operations propagate relative error. It follows the rules:

$$\delta(a \pm b) = \delta(a) + \delta(b)\,;\ \delta(a \times b) = \delta(a) \times |b| + \delta(b) \times |a|\,;\ \delta\left(\frac{a}{b}\right) = \frac{\delta(a)\times|b| + \delta(b)\times|a|}{b^2}.$$

It is an upper bound of the propagated error. In computation of real world data, the original rounding error is averagely distributed, and all propagations are isolated, so the propagated error is less than the upper bound far away. The usual rule is not adoptable for actual application computation.

If we use the upper bound as the tolerance, the computation must have a mistake, and this is validated in the plane sweep algorithm. Then we should adopt an appropriate method to estimate the propagated error. The tolerance must be proper enough—if it is bigger, we may get extra segments passing the current point; or we may omit some segments passing the current point.

In the experiments about Physics or Chemistry, all the measuring errors are affected by many slight factors, and they are isolated with each other. As a result, the rules used in those fields are Gaussian Rules:

$$\delta(a \pm b) = \sqrt{\delta^2(a) + \delta^2(b)};\ \delta(a \times b) = \sqrt{\delta^2(a) \times b^2 + \delta^2(b) \times a^2};$$

$$\delta\left(\frac{a}{b}\right) = \frac{\sqrt{\delta^2(a)\times b^2 + \delta^2(b)\times a^2}}{b^2}.$$

The Gaussian rules follow the “3θ” principle. It means that if two input errors lie in the regions of $(-3\theta_1, 3\theta_1)$ and $(-3\theta_2, 3\theta_2)$ with probability of 99.97%, then their summation of error lies in the region $(-3\sqrt{\theta_1^2+\theta_2^2}, 3\sqrt{\theta_1^2+\theta_2^2})$ with probability of 99.97%. The summation error is absolute error for add and subtract operations, while it is relative error for multiply and divide operations. Moreover, the tolerance is an absolute error not a relative one and it differs with values of difference regions. So any tolerance is not a static value but a dynamic value related with the value region.

The algorithm’s source of error is the intersection computation, and where we need a tolerance is to judge whether a segment passes a point (the intersection) as mentioned above. In the process, we just assume that the computation value’s order of magnitude is N, i.e. the value region is:$1 \times 10^{n} \sim 9.99 \times 10^{n}$.

Following the Gaussian Error propagation rule, we get the error propagated by each step and get the formula to get the error:$7.8 \times 10^{(2N-15)}$, and use the error as the tolerance. The formula may be a little different according with different computing process of getting the intersection point and getting the point to segment relation.

We use the formula and set a proper tolerance for the plane sweep algorithm. This guarantees to get the segments passing the point correctly. In a test with real world data with magnitude 5, we get the tolerance 7.8×10^{-5}. For three groups of test data, the test result is shown in table 1:

Table 1 Test result 1 of real world data

	NO. of Area Values	NO. of area value greater than 1×10^{-5}	Min value	Max value
1	62488	145	-3.52×10^{-5}	4.13×10^{-5}
2	30228	40	-1.38×10^{-5}	4.13×10^{-5}
3	258288	24	-1.92×10^{-5}	1.82×10^{-5}

As shown in the table, the tolerance 7.8e-5 can guarantee the correctness of the algorithm. Another group of test data is shown in table 2. The data takes attitude and longitude as its units, and generally the values of the data is less than 10. The order of magnitude is 0~1, and we just select 0 as the parameter, and then the tolerance is 7.8×10^{-15}. The tolerance always is greater than the computed absolute values of the triangle area, and can guarantee the correctness, as shown in table 2.

Table 2 Test result 2 of real world data

	NO. of AreaValues	NO. of area values greater than 1×10^{-15}	Min value	Max value
1	266678	81	-2.04×10^{-15}	1.78×10^{-15}
2	4478	46	-2.25×10^{-15}	4.45×10^{-15}
3	8164	13	-1.31×10^{-15}	1.23×10^{-15}

Two groups of tests on different real world data show the correctness of the method on plane sweep algorithm. We have use this method and implement a robust plane sweep algorithm, which is used for intersection-computing for vector map overlay.