On the Role of Kinesthetic Thinking in Computational Geometry *

J. Antoni Sellarès † and *Godfried Toussaint* ‡

Abstract

Computational geometry is a new (about 30 years) and rapidly growing branch of knowledge in computer science that deals with the analysis and design of algorithms for solving geometric problems. These problems typically arise in computer graphics, image processing, computer vision, robotics, manufacturing, knot theory, polymer physics and molecular biology. Since its inception many of the algorithms proposed for solving geometric problems, published in the literature, have been found to be incorrect. These incorrect algorithms rather than being "purely mathematical" often contain a strong *kinesthetic* component. This paper explores the relationship between computational geometric thinking and kinesthetic thinking, the effect of the latter on the correctness and efficiency of the resulting algorithms, and their implications for education.

Key Words and Phrases: convex hull problem, diameter, triangulation, kinesthetic heuristics, kinesthetic thinking, mathematical discovery, algorithm design and analysis, computational geometry, artificial intelligence, cognitive science.

CR Categories: 3.36, 3.63, 5.25, 5.32, 5.5

1 Introduction

1.1 Computational geometry

Computational geometry is claimed by many computer scientists to be a new field of computer science which began with the thesis of Michael Shamos [1] in 1978. Since that time, spurred by rapid developments in computer graphics, VLSI (Very Large Scale Integration), computer vision, robotics, manufacturing, knot theory, polymer physics and molecular biology, it has blossomed into a dynamic discipline of its own and has given birth to several books in the area [2], [3], [28], [29], [37], [49], [50]. For two existing surveys of the field relevant to pattern recognition see [4] and [5]. A special issue devoted to computational geometry gives a bird's eye view of the field [6]. As far back as 1981 it has been discovered by several authors [5] that a score of published algorithms, that have been proposed for solving a variety of geometric problems, actually do not work correctly, i.e., they do not always compute what they were designed to compute. Thinking about these algorithms, how they fail, how they compare to correct algorithms and trying to fix them has led to the ideas discussed in this paper.

*A preliminary version of this paper titled *Computational Geometric Thinking as Kinesthetic Thinking* was presented by Godfried Toussaint at the *Conference on Thinking*, Harvard University, August 1984.

†Institut d'Informàtica i Aplicacions, Universitat de Girona, Spain (sellares@ima.udg.es). Partially supported by DURSI 2001SGR-00296.

‡School of Computer Science, McGill University, Montreal, Quebec, Canada (godfried@cs.mcgill.ca). Partially supported by NSERC and FCAR.

1.2 The psychology of discovery in computational geometry

Psychologists, mathematicians and physicists have been interested for a long time in the process of discovery in mathematics [7], [8], [27]. The process of discovery in the design of geometric algorithms offers a rich environment for psychologists and artificial intelligence researchers to study because of the strong visual and kinetic components present in geometry problems. This is even more so due to the coming of the robotics age where vision, movement and their interaction are crucial problems. This paper presents some views on this topic and conclusions regarding both the design and analysis of geometric algorithms as well as implications for education. It is by no means a report of rigorous psychological testing on subjects, but rather an account of a personal study on the subject in the spirit of Hadamard [7], Mach [27] and Lakatos [8].

1.3 Logico-mathematical vs. kinesthetic or body-syntonic heuristics

It is useful to distinguish between various forms of knowledge [9], intelligence [10] and thinking [11]. While it is true that our knowledge, intelligence, and thinking, particularly that concerned with geometry, are originally acquired through our visual, tactual and other sensory interactions with our environment, it is also true that eventually, as we acquire idealized concepts, our logico-mathematical "machine" takes on a "life of its own" and we are therefore capable of doing logico-mathematical thinking with a minimum of "interference" from the "real world." This is a "double-edged sword": it has its advantages and disadvantages. In this paper we will distinguish between *logico-mathematical* thinking and *kinesthetic* thinking. The term kinesthetic thinking follows the terminology of Johnson [47] and Svanaes [48] and signifies direct cognitive operations on tactile kinesthetic sense experiences. A kinesthetic heuristic is any heuristic in which cognition, understanding and learning takes place through perceptible results of dynamic manipulation of objects to support useful insights on the problem been studied. Kinesthetic heuristics emphasize the experimental aspects related to logico-mathematical deductions as a primary tool of understanding. Similarly we may speak of kinesthetic versus logico-mathematical knowledge or information. While the boundary between the two forms is necessarily fuzzy there are nevertheless instances where they are quite distinct as will become evident when we consider the design of geometric algorithms. It is also useful to distinguish between kinesthetic and visual thinking (knowledge, intelligence) [12], [31] - [36], [38] although the boundary here is also fuzzy and even motor theories of vision have been proposed [24]. That tactual kinesthetic information exists quite apart from visual information was dramatically demonstrated by Louise Pelland [13], [14]. Pelland carried out a study in which two groups of students had to draw an object (an artichoke cut in half). One group spent some time manipulating, touching, and feeling the object before drawing it. The other group spent the same amount of time, not touching the object, but looking at it from different points of view and imagining that they were touching it. The drawings were then ranked by professional artists according to various criteria. With significant results of statistical tests Pelland demonstrated that the "touchers" had a considerably improved drawing skill acquisition over the "non-touchers."

In this paper we review some algorithms from the computational geometry literature with the aim of classifying them according to whether they were designed using logico-mathematical heuristics or visual-kinesthetic heuristics. The latter type of heuristic will also be referred to as body-syntonic, a term used by Papert [15] for a similar notion.

It will be argued that kinesthetic heuristics lead to faster geometric algorithms but are usually more difficult to prove correct. On the other hand some kinesthetic heuristics are intuitively extremely convincing. This has implications for what constitutes the notion of a "proof" of correctness. The implications of kinesthetic heuristics for education will also be discussed.

By way of introduction, and before delving into the computational geometric examples, the difference between logico-mathematical and kinesthetic proofs of geometric theorems is illustrated with a very simple and old theorem that appears in Euclid's elements as Proposition 32. This proposition states that in any triangle the sum of the three interior angles sums to two right angles (180 degrees). Euclid's *Elements* is the epitome of the logico-mathematical approach to geometry in which proofs and constructions are built on logical arguments from previous propositions, postulates and axioms. His proof of Proposition 32 is no exception and goes as follows (refer to Figure 1).

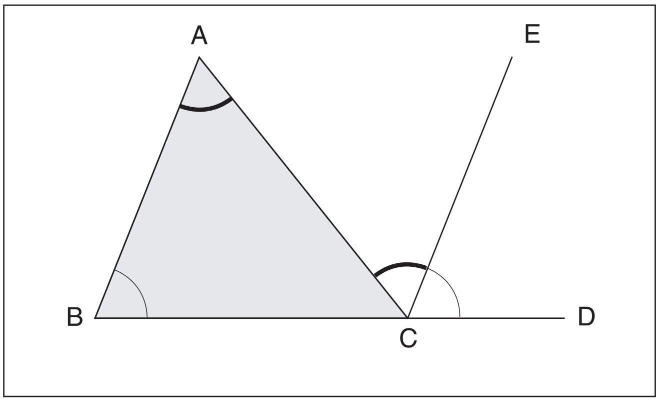

Figure 1: A logico-mathematical proof of Proposition 32 in Euclid's *Elements*.

Let ABC be a triangle, and let side BC be produced to D. Draw CE through point C parallel to the straight line AB. Since AB is parallel to CE, and AC falls upon them, the alternate angles BAC and ACE are equal to one another. Again, since AB is parallel to CE, and the straight line BD falls upon them, the exterior angle ECD equals the interior and opposite angle ABC. But the angle ACE was proved to be equal to the angle BAC. Therefore the whole angle ACD equals the sum of the two interior and opposite angles BAC and ABC. Now add the angle ACB to each. Then the sum of the angles ACD and ACB equals the sum of the three angles ABC, BCA, and CAB. But the sum of the angles ACD and ACB equals one half-plane or two right angles. Therefore the sum of the three interior angles of the triangle equals two right angles.

One of the earliest discussions on the role that kinesthetic information in general, and movement in particular, have on the development of geometry is the delightful book *Space and Geometry* by Ernst Mach [27] published in 1906. It is concerned with the evolution of geometry from the psychological and physiological points of view. Mach discusses two kinesthetic (body-syntonic) heuristics to deduce Proposition 32. The first he attributes to Thibaut [44], a contemporary of Gauss (refer to Figure 2).

Imagine the vertices of the triangle to be nails protruding from the plane. Consider a long bar initially touching the two nails at B and A on the outside of the triangle. Let the bar have an orientation, i.e., a head and a tail. The orientation is from the tail to the head and the head is identified by the arrow in Figure 2. In step 1 rotate the bar in a clockwise direction until it hits the nail at C. The bar has been rotated by an angle α, the marked exterior angle at A. In step 2 we rotate the bar again in a clockwise direction until it hits the nail at B. Now the bar has been rotated by an angle γ, the marked exterior angle at C. Finally we repeat the same procedure rotating about B. At the end of the three rotations the bar reaches its original position for the first time and therefore has been rotated by a total of 360 degrees or four right angles. Therefore $\alpha + \beta + \gamma = 4$ right angles. However, the sum of the three interior angles at A, B, and C, plus $\alpha + \beta + \gamma = 3$ half-planes or 6 right angles. Therefore the sum of the interior angles $= 6 - 4 = 2$ right angles.

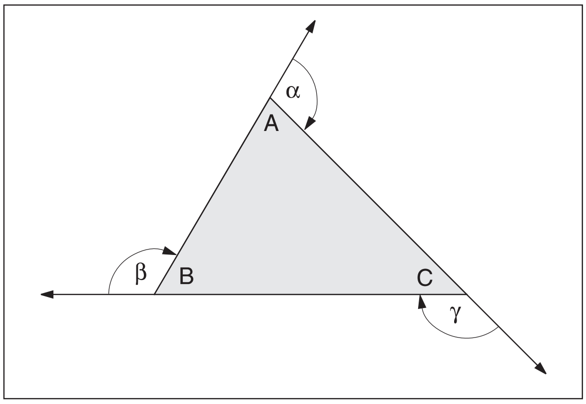

Figure 2: A kinesthetic proof of Proposition 32 due to Thibaut [44].

The second kinesthetic approach to deducing Euclid's Proposition 32 was noticed by Mach himself. This approach also rotates a bar but is simpler and more direct than Thibaut's method (refer to Figure 3). Mach attributes this approach to the way a draftsman draws a triangle with a straight edge when given three points that mark the vertices of the triangle. Accordingly, place the bar initially collinear with points A and C pointing in the direction from A towards C. First rotate the bar in a clockwise direction holding the bar fixed at point A until it becomes collinear with point B. It has then rotated by an angle α. Then rotate the bar again in a clockwise direction holding the bar fixed at point B until it becomes collinear with point C. Finally, repeat the procedure about point C until the bar is again collinear with points A and C for the first time. Therefore the total angle of rotation of the bar is the sum of the 3 interior angles $= \alpha + \beta + \gamma$. But note that the orientation of the final position of the bar is now opposite to what it was at the start; the bar points from C towards A. Therefore the bar has rotated by one half-plane or 2 right angles Therefore $\alpha + \beta + \gamma = 2$ right angles.

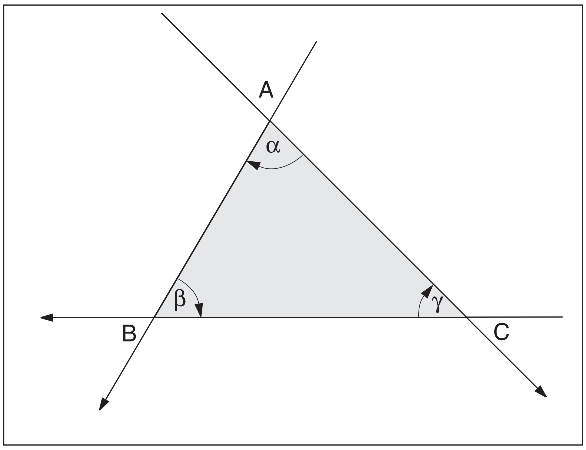

Figure 3: A kinesthetic proof of Proposition 32 due to Mach [27].

The two kinesthetic heuristics described above are both "good" heuristics in the sense that they lead to correct theorems (Proposition 32) in euclidean geometry. They are convincing algorithmic strategies for computing the sum of the internal angles of a triangle. However, one must be careful about what is expected from them. In the context of more general axiomatic geometries the procedures still require more proof. The gap lies in the formula for computing the final angle and can be filled by invoking the theorem (valid in euclidean geometry) on the composition of multiple rotations [46]. Interestingly, if Proposition 32 itself is taken as a hypothesis then it implies Euclid's fifth postulate (the parallel postulate) whereas Thibaut's procedure does not [45].

2 The Convex Hull Problem

2.1 Logico-mathematical heuristics

The convex hull of a set of n points on the plane $P = \{p_1, p_2, \cdots, p_n\}$ is defined as the minimum area convex polygon $CH(P)$ enclosing the set (see Figure 4). We are interested in finding this convex polygon for a given set P.

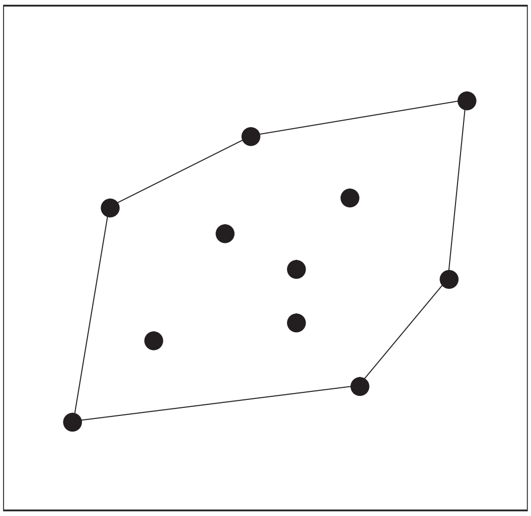

Figure 4: The convex hull of a set of points.

A Logico-mathematical heuristic:

If two points p_i, p_j are such that all the n-2 remaining points in P lie on the same halfplane determined by the line through p_i and p_j then $[p_i, p_j]$ forms an edge of the convex hull of P. This property leads to the following algorithm due to Rosenberg and Landgridge [16].

Algorithm CHl

Begin

Step 1: Through each pair of points draw a line.

Step 2: For each line drawn in step 1 test to see if the remaining points all lie to one side. If they do then add the pair to a list containing the edges of the convex hull.

End

This algorithm is typical of the logico-mathematical type. It is easy to prove its correctness and easy to program. The heuristic or property is clearly a logico-mathematical (in fact it is combinatoric) rather than kinesthetic. However the algorithm is very slow. Because the number of lines drawn in step 1 is proportional to n^2 and for each of these the number of points tested in step 2 is proportional to n, the resulting total time taken by the algorithm is proportional to n^3. Several other algorithms exist with this type of heuristic that are even slower (see Shamos [1]).

2.2 Kinesthetic heuristics

In contrast to the "combinatorial" type of convex hull algorithms in section 2.1, several algorithms exist that use strong kinesthetic heuristics.

The Gift-wrapping heuristic:

Here we think of wrapping a very long string around the set of points (see Figure 5). This idea gives rise to the following algorithm due to Jarvis [17].

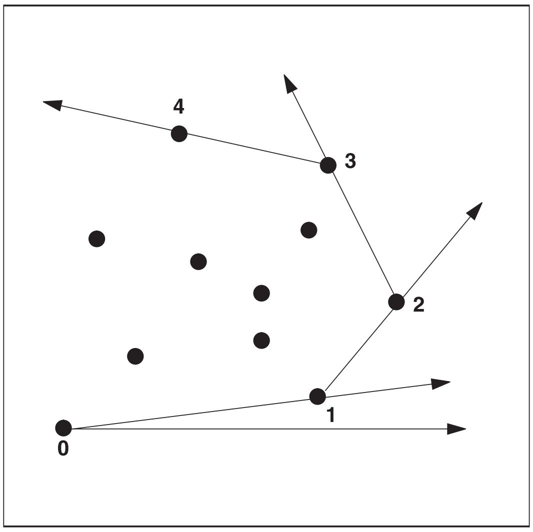

Figure 5: Wrapping a string around a set of points.

Algorithm CH2

Begin

Step 1: Find the point with minimum y-coordinate. Put it on the list of convex hull vertices.

Step 2: To determine the next point "touched by the string" compute all angles that the x-axis makes with the string through all other points and select the point corresponding to the minimum counter-clockwise angle. Add this point to the list of convex hull vertices.

Step 3: Repeat step 2 with respect to the newly found point until the starting point is encountered again.

End

The idea of "swinging" a taut string or stick around the set until it "hits" a point is of course, a strong kinesthetic heuristic and we "know" from our experience in the real world that this procedure will identify the first, and therefore "outermost", point in some direction. To provide a formal proof of correctness for this algorithm requires more effort than for algorithm CHl although we are so convinced of its correctness from the kinesthetic point of view that no proof seems required. In fact, in [17] the author gave no proof of correctness for this algorithm. A "good" kinesthetic heuristic will often have this apparent self-evident property. However, some kinesthetic heuristics are either incorrect or not self-evident and beg us for formal confirmation. Recognizing the "good" kinesthetic heuristics seems to be part of the art of good algorithm design.

On the other hand, CH2 runs much faster than CHl. If there are H points on the convex hull then for each such point the algorithm takes time proportional to n due to step 2. Therefore the total time taken is proportional to Hn. Since H is less than n the algorithm never takes more than n^2 units of time which is less than the n^3 units taken by algorithm CHl.

<u>The elastic band heuristic</u>:

Imagine that an elastic band is stretched out into the form of a large circle that contains all the points of Figure 4. Furthermore, think of the points as nails sticking out of the plane. We let go off the elastic band. The band shrinks until it comes into contact with some nails: the convex hull vertices. Finally at rest the elastic band defines the convex hull.

Again, this is a strong kinesthetic heuristic evoking in us such body-syntonic notions as encircling, shrinking and stretching, coming into contact with, blocking, etc. Of course, the heuristic is only the germ of the algorithm. There is sometimes a non-trivial gap which must be bridged when we jump from the heuristic to the precise mathematical description of the algorithm. For example, in algorithm CH2 "swinging the string until it hits a point" was described mathematically as "computing angles and selecting the minimum angle." Similarly, with the elastic band we must find methods for implementing the heuristic. Several possibilities exist. The shrinking of the elastic is a parallel analog operation and in algorithm design we usually look for a sequential digital procedure. We now describe an algorithm very frequently proposed by students upon first encountering this heuristic. The algorithm uses the kinesthetic heuristic of taking an un-stretched elastic band and enlarging it step by step until it encloses all the points. No one has any difficulty performing this operation with a real elastic band on a board with pegs sticking out of it. However, the students' formal descriptions of what they do physically often betray their actions. It seems here that "an action is worth a thousand pictures".

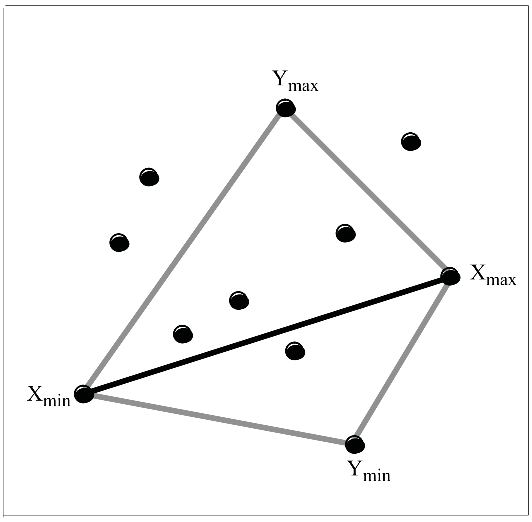

Figure 6: Progressively stretching an elastic band until it encompasses all the points.

Algorithm CH3

Begin

Step 1: Place the elastic band stretched between the two points with maximum and minimum x-coordinates. (see Figure 6)

Step 2: Take the upper section of the elastic and connect it onto the point with maximum

y-coordinate. Similarly, take the lower section to y_{min} (see Figure 6). This leaves us with four corners to work with.

Step 3: Consider the northeast corner or triangle y_{max}, z, x_{max}. For all points in this triangle find the two points with maximum x and y coordinates ($y_{max(2)}$ and $x_{max(2)}$) again and connect the elastic band from y_{max} to x_{max} via these new points (see Figure 7).

Step 4: Repeat step 3 for points left "outside" the elastic band until none remain.

Step 5: Repeat steps 3 and 4 for the three remaining corners.

End

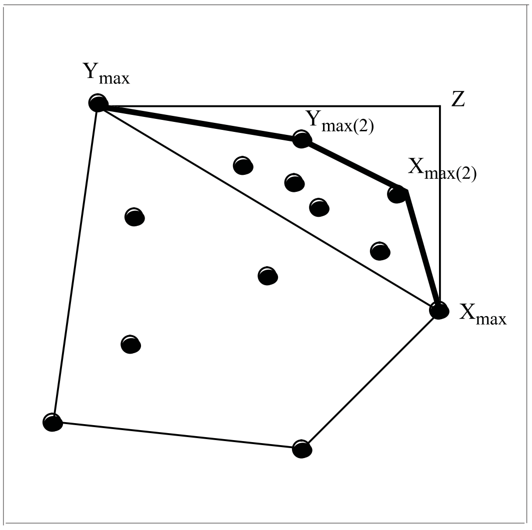

Figure 7: An incorrect way to recursively stretch the elastic band.

While this algorithm often works correctly it is in fact incorrect and it is easy to find sets of points for which the algorithm fails. This is left as an exercise for the reader.

The reason the algorithm can fail is that Step 3 violates the notion of global exterior of the set by repeatedly searching for points with maximum x and y directions. Since these directions have in a sense already "been used up" in steps 1 and 2 they cannot be used again. Therefore this implementation is not the correct kinesthetic inverse of the shrinking procedure. We can remedy this by modifying Step 3 to search for the points furthest in a direction perpendicular to the line through y_{max} and x_{max}. In this way we always search in new directions and we guarantee that points are global rather than local extrema. This latter algorithm has been independently discovered by several researchers (see for example [18], [19]). As with algorithm CH2 it is more difficult than CHl to prove correct but it is extremely fast. In fact, it was proven in [19] and [20] that in many situations it executes in time proportional to n, the number of input points.

3 The Diameter Problem

The diameter of a set of points $P = \{p_1, p_2, \cdots, p_n\}$ on the plane, denoted by $D(P)$ is the distance between the two furthest points of P. The diameter problem is to identify these points (points p_i and p_j for example in Figure 8) given a set P.

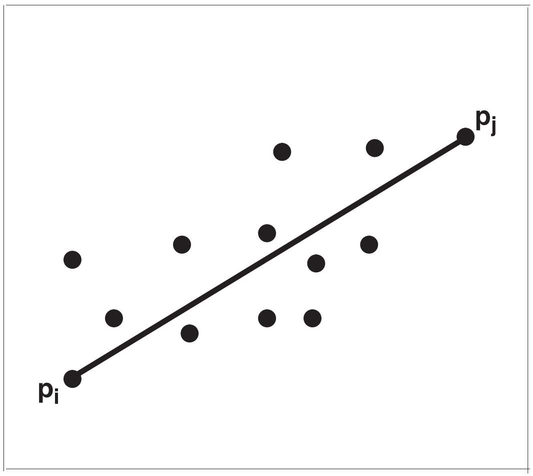

Figure 8: The diameter of a set of points.

We will consider the special problem of finding the diameter of a convex polygon, i.e., the points that form the vertices of a convex polygon P which we are given as an ordered list, say, in a clockwise direction (see Figure 9).

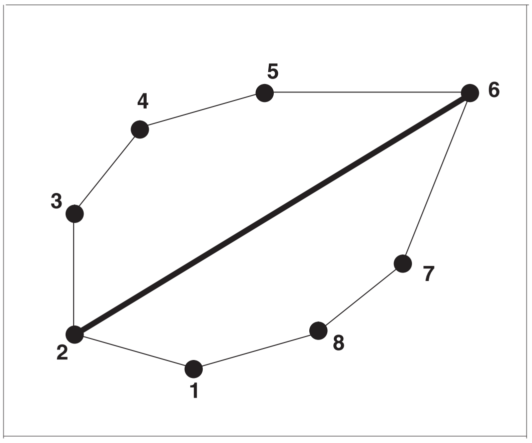

Figure 9: The diameter of a convex polygon.

3.1 The Logico-Mathematical Approach

The most obvious way to solve this problem is to consider all pairs of points, compute their distance, and select the pair that yields the maximum distance. The proof of correctness of this procedure is trivial but the algorithm is slow. The time taken by this algorithm is proportional to the number of pairs of points, that is, proportional to n^2.

3.2 Kinesthetic Heuristics

One published algorithm which almost every student of computational geometry rediscovers is based on the growing elastic heuristic. This approach is similar to the convex hull approach in algorithm CH3 (the first step is identical in fact) except that we always keep the elastic around two points of the set only. We think of the ends of the stretched elastic as the *head* and *tail*.

Algorithm Dl

Begin

Step 1: Find the vertices of P with the minimum and maximum x-coordinate and place the elastic band around these two points. Let x_{max} be the head, x_{min} the tail. (see Figure 10)

Step 2: With the tail fixed proceed to move the head to the adjacent clockwise or counterclockwise point if the elastic gets longer. Continue doing this until it does not grow any longer. (In Figure 10 the head will stop at p_i)

Step 3: Now with the head fixed at p_i proceed to move the tail in the same way. (In Figure 10 it will stop at p_j)

Step 4: Keep alternating the role of head and tail as in steps 2 and 3 until the elastic does not grow any longer. The final position of the elastic identifies the diameter $D(P)$ and the diametrical pair of points. (In Figure 10 the elastic will settle on p_i and p_j which is the true diameter of the set).

End

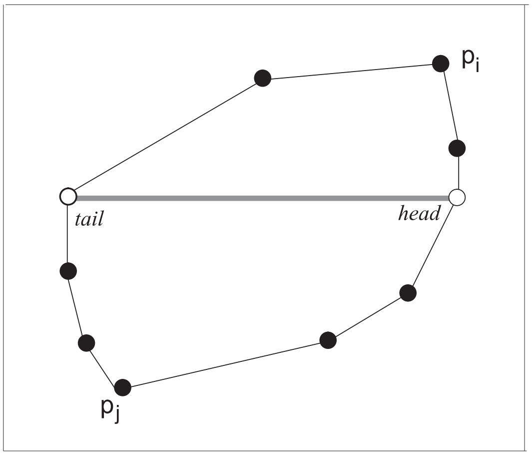

Figure 10: The stretching elastic heuristic for searching the diameter.

This algorithm provides another example of a strong kinesthetic heuristic which is so convincing it would appear not to require a proof. Again, no proof of correctness was published with this algorithm. While the algorithm is very fast indeed and runs in time proportional to n, it is in fact incorrect. A counter-example is given in [21] but the readers are invited to discover one on their own before checking [21]. The reason for failure is similar to that for algorithm CH3. As is often the case in so-called hill-climbing procedures such as these, algorithm Dl can get stuck in a local rather than global maximum.

<u>The rotating calipers heuristic:</u>

This is a dynamic version of the sandwich heuristic. One way to measure the width of an object, such as a hamburger patty, is to place it in between two slices of bread and measure the distance between the slices, i.e., the height of the sandwich not counting the thickness of the slices of bread. In a mechanics laboratory one does this with a set of *calipers* to ensure that "the two slices of bread remain parallel to each other." The rotating-calipers heuristic is based on the intuition that if we "rotate the two slices of bread" simultaneously keeping them parallel then the greatest distance encountered between points of contact should be the diameter we are looking for. In our two-dimensional example this is illustrated in Figure 11. This idea leads to the following algorithm [1], [22].

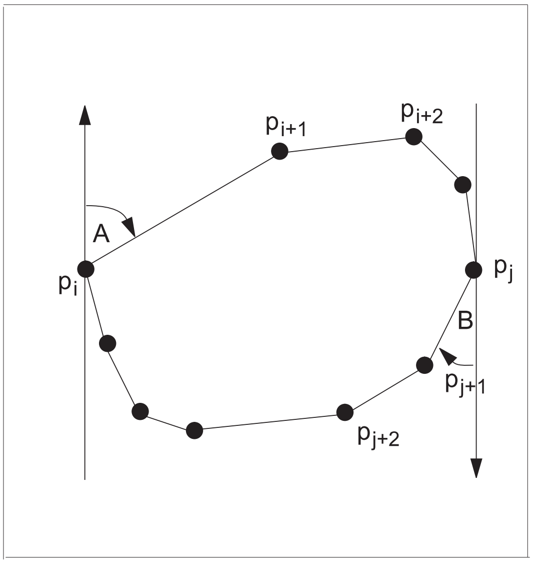

Figure 11: The rotating calipers heuristic.

Algorithm D2

Begin

Step 1: Find the two points with minimum and maximum x-coordinates. Call them p_i and p_j (see Figure 11).

Step 2: Place the calipers vertically on p_i and p_j. (They will be rotated in a clockwise direction.)

Step 3: Measure the two angles A and B. Compute the distance between p_i and p_j as a candidate for the diameter $D(P)$.

Step 4: Rotate the calipers by the smaller of the two angles (in Figure 11 it is B). This gives a new pair of contact points (in Figure 11 we now have p_i and p_{j+1}). Measure their distance and store it in a list of candidates.

Step 5: Repeat steps 3 and 4 until the calipers return to their original position. The largest distance encountered among the candidates is the diameter $D(P)$.

End

Algorithm D2, like algorithm Dl, runs very fast and in time proportional to n. Unlike algorithm Dl, algorithm D2 is always guaranteed to give the correct solution but the proof of correctness is longer and more difficult than that for the logico-mathematical algorithm.

We note here that the rotating caliper heuristic can be used to solve a variety of computational geometric problems besides the diameter problem and it is thus quite a powerful general tool [22]. For a web page with a description of various algorithms and applications as well as interactive JAVA applets that the user can play with see [39].

4 The Polygon Triangulation Problem

One of the fundamental theorems in geometry is that every simple (non self-crossing) polygon can be triangulated, i.e., can be decomposed into triangles by adding straight line segments (diagonals) in the interior of the polygon between pairs of vertices such that no two diagonals intersect each other except at their endpoints [40]. A polygon and one of its triangulations is shown in Figure 12.

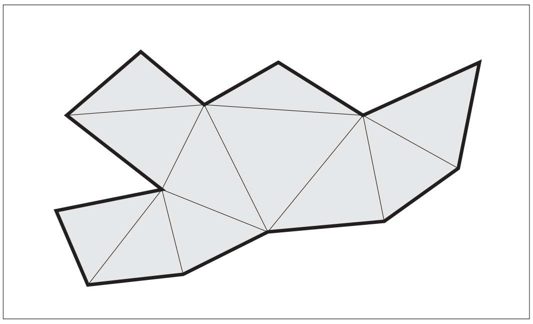

Figure 12: A triangulation of a simple polygon.

A typical algorithm to triangulate a polygon starts by looking for a valid location in which to insert a diagonal. Inserting this diagonal divides the original polygon into two smaller polygons. The same procedure is then applied to the smaller polygons recursively until each remaining polygon is itself a triangle. The union of these triangles then forms a triangulation of the polygon. Therefore the crucial part of such a procedure is to find a diagonal in a simple polygon.

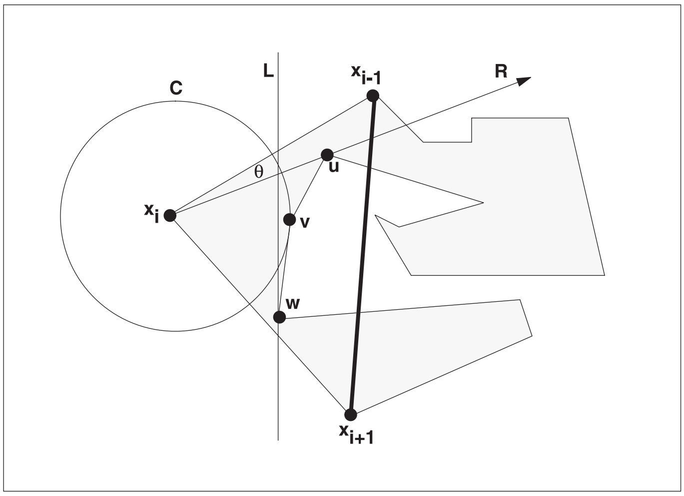

Figure 13: Illustrating three heuristics for inserting a diagonal between x_i and another vertex.

When students of computational geometry are first given the task of designing an algorithm for inserting a diagonal in a simple polygon P they often propose the following (and in fact published) procedure (refer to Figure 13).

Algorithm ID1

Begin

Step 1: Find the vertex of P which has minimum x-coordinate and call it x_i.

Step 2: If the interior of triangle $[x_{i-1}, x_i, x_{i+1}]$ contains no other vertices of P then insert the diagonal $[x_{i-1}, x_{i+1}]$, in which case we are done. Such an empty triangle is called an *ear* and in fact there is a theorem that states that every simple polygon of more than four vertices has at least two non- overlapping ears [42].

Step 3: Otherwise find the vertex in the interior of triangle $[x_{i-1}, x_i, x_{i+1}]$ that is closest to x_i (vertex v in Figure 13) and insert diagonal $[x_i, v]$. One can think of growing a circular disk C centered at x_i until it collides with a vertex of P in the triangle $[x_{i-1}, x_i, x_{i+1}]$.

End

It turns out that this definition of closeness is not a good kinesthetic heuristic for this problem. Indeed, a counter-example to this algorithm is given in Figure 14.

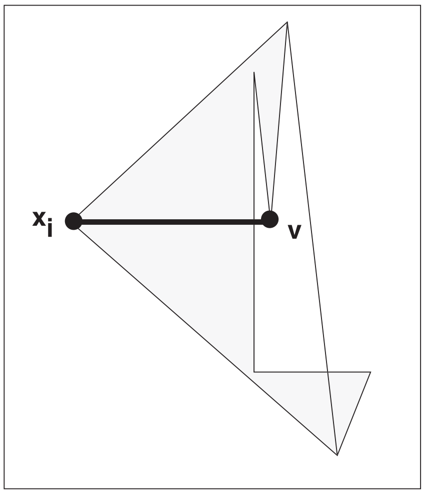

Figure 14: The closet vertex v to x_i need not form a diagonal of P.

Another kinesthetic heuristic frequently discovered by students (and also published) involves rotating a ray.

Algorithm ID2

Begin

Step 1: Find the vertex of P which has minimum x-coordinate and call it x_i.

Step 2: Consider a ray R anchored at vertex x_i initially pointing in the direction of x_{i-1}.

Step 3: Rotate R in a clockwise manner sweeping through the triangle $[x_{i-1}, x_i, x_{i+1}]$ until the ray collides with a vertex of P contained in this triangle (vertex u in Figure 13 making an angle θ) and insert diagonal $[x_i, u]$.

End

It turns out that this kinesthetic heuristic is also incorrect. A counterexample to this proof is given in Figure 15.

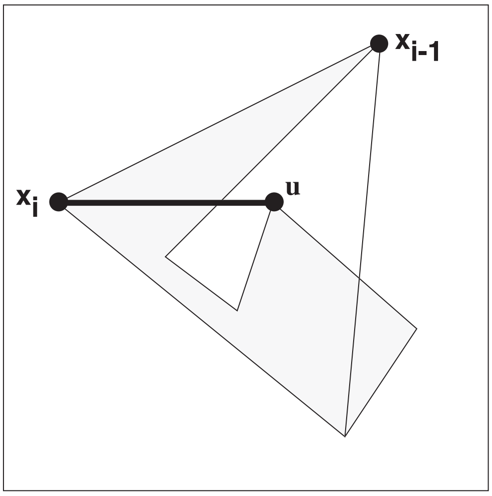

Figure 15: The vertex u that has smallest angle $x_{i-1}x_iu$ need not form a diagonal of P.

A third heuristic frequently discovered by students (and also published) sweeps a vertical line.

Algorithm ID3

Begin

Step 1: Choose the initial vertex x_i so that it has the minimum x-coordinate among all vertices in the polygon.

Step 2: Sweep a vertical line L from left to right starting at x_i until it collides with the first vertex of P contained in the triangle $[x_{i-1}, x_i, x_{i+1}]$ (vertex w in Figure 13) and insert diagonal $[x_i, w]$.

End

Once again, this kinesthetic heuristic is incorrect and a counter-example is given in Figure 16.

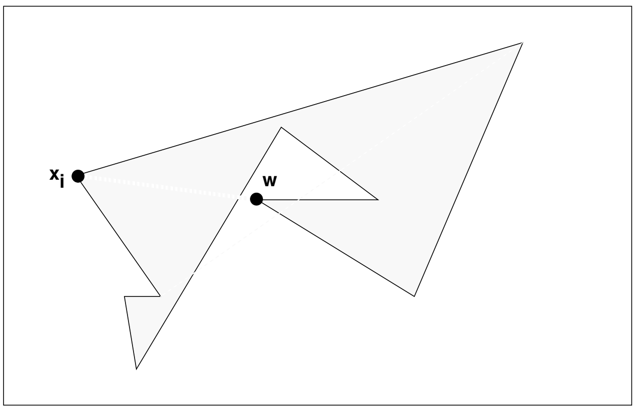

Figure 16: The vertex w with the second least abscissa after x_i need not form a diagonal of P.

To finish this section on a more positive note, a "good" kinesthetic heuristic is given for finding a diagonal between x_i and some other vertex contained in triangle $[x_{i-1}, x_i, x_{i+1}]$, followed by a proof of its correctness (refer to Figure 17).

Algorithm ID4

Begin

Step 1: Find the vertex of P which has minimum x-coordinate and call it x_i.

Step 2: Consider a ray R anchored at x_{i-1} initially pointing in the direction of x_i.

Step 3: Rotate this ray in a counter-clockwise sense until it first hits a vertex of the polygon in triangle $[x_{i-1}, x_i, x_{i+1}]$ (for example vertex y in Figure 17). The desired diagonal is given by $x_i y$.

End

To see that this heuristic is correct let the ray R intersect the segment x_i, x_{i+1} at point z throughout its rotation. Until the ray hits a vertex, the segment $x_{i-1}z$ and therefore the triangle $[x_{i-1}, x_i, z]$ lie in the interior of the polygon. Therefore $x_i y$ is a valid diagonal of the polygon.

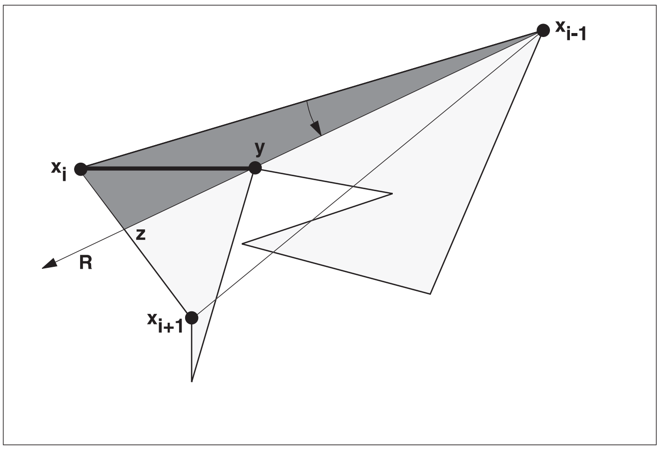

Figure 17: A correct heuristic for inserting a diagonal.

For additional details on incorrect and correct algorithms for this problem the reader is referred to [41] and [43].

5 Conclusions

5.1 Design and analysis of geometric algorithms

In section one it was pointed out that many recently published algorithms in computational geometry have been discovered to fail [5], [21], [41], [43]. One thing that all these algorithms have in common is that they are based on kinesthetic heuristics rather than logico-mathematical heuristics. A second thing that they all have in common is that they are usually much faster than their logico-mathematical counterparts, even when they do work correctly. This suggests that kinesthetic heuristics are better than logico-mathematical ones from the computational efficiency point of view and therefore that the kinesthetic approach is a good pedagogical strategy for the design of fast geometric algorithms. On the other hand, the fact that many algorithms based on kinesthetic heuristics fail is disconcerting and suggests that we have a "double edged sword". More powerful kinesthetic heuristics tend to create stronger geometric illusions and therefore call for extra care in logical thinking to determine if the resulting algorithms are correct.

There are at least two reasons why many of these algorithms fail. The first is that since a proof of correctness is traditionally a logico-mathematical exercise it "gets along well" with logico-mathematical heuristics. On the other hand there is an extra-logical gap between a logico-mathematical proof and a kinesthetic type of proof or algorithm. This gap must be bridged successfully for the algorithm to work correctly. This is often the first trap the algorithm designer must avoid. Of course with rigorous logical thinking one should be able to construct this bridge perhaps with added difficulty. The other side of this sword however is that often the kinesthetic heuristics are so convincing that either a proof seems unnecessary and is not given, or as is often the case an incorrect proof is outlined. It is conjectured that when an incorrect proof is given it is because the convincing nature of the heuristic causes the proof to be merely a rationalization (rather than a critical analysis) of something we are already convinced must be correct. This is the second trap facing the algorithm designer. A good strategy is therefore to harness the kinesthetic heuristics, but also mistrust them no matter how convincing they may appear to be, and subject them to rigorous logico-mathematical analysis.

5.2 Implications for education

Traditionally, in the educational system, mathematics in general, and proofs in particular, are presented as dry, purely logical, step by step procedures that lead in a "vertical" way to the "goal". This is of course only the finished product of the "living" mathematics which is a more passionate and semi-blind road to discovery [8]. In the words of Seymour Papert [15].

> "Mathematical work does not proceed along the narrow logical path of truth to truth to truth, but bravely or gropingly follows deviations through the surrounding marshland of propositions which are neither simply and wholly true nor simply and wholly false."

We have seen in this paper that there is a gap that must be bridged between kinesthetic heuristics for solving a problem and the finished algorithm with its proof of correctness. This is of course only an exaggerated form of a situation that exists in scientific discovery in general [8] because of the inherent gap between "reality" and mathematics. Albert Einstein [23] put it this way:

> "As far as the laws of mathematics refer to reality, they are not certain; and as far as they are certain, they do not refer to reality."

Computational geometry with kinesthetic heuristics offers a rich and more relevant manner of teaching a "living" mathematics. It offers the student a "truer" picture of reality by combining mathematics not only with physics but also with the psychology of visual perception. It illustrates the dangers involved in bridging the gap between the physical and Platonic worlds and makes mathematics a more "dangerous" undertaking. Computational geometry should thus be a good antidote for mathophobia. Readers familiar with Turtle Geometry [25] may have noticed its similarity to the computational geometry with kinesthetic heuristics discussed in this paper. In fact Turtle Geometry is one type of *computational* geometry and the moving turtle is quite an elegant and simple fundamental kinesthetic operator. Some of the algorithms in computational geometry such as those using the rotating calipers can be easily placed in the turtle framework [25]. The main difference is that the computational geometry presented here is more advanced than turtle geometry. While at present this material forms the content of graduate courses at many of the world's leading universities, this is not because the material is that difficult, but rather because it lies at the frontiers of knowledge. Much of the material in [1] - [3] is already creeping down to the undergraduate curriculum and is eminently suited to high school students with personal computers at their disposal. Therefore this material would appear to be a natural continuation of the turtle geometry program envisioned by Papert [25] for children, into the high school context. Recently dynamic geometry [51], a new emergent conception of computer assisted geometry, has joined turtle geometry and in part has moved off it. Dynamic geometry is based on interactive software programs such as Geometer's Sketchpad, Cabri-Géomètre and Cinderella. Although these programs emphasize the construction and exploration of planar configurations and the detection of invariants by means of observation, it is possible, with them, to implement most of the kinesthetic algorithms described here, thus allowing us to use the computer as a tool for a type of simulation intermediate between purely mental simulation or immagination and physically attainable simulation.

It has been argued that there are "good" and "bad" kinesthetic heuristics in the sense that some yield correct algorithms whereas others yield incorrect ones. It would appear then that a good pedagogical strategy for the successful algorithm designer in computational geometry (particularly that aspect of it dealing with dynamic problems such as collision avoidance and spatial planning in robotics) is to train the kinesthetic sense as well as the logical thinking skills. In fact Svanaes [48] argues that kinesthetic thinking is more important than logico-mathematical thinking for the successful design of software such as graphical user interfaces. Turtle geometry is a step in the right direction but restricted to the visual domain since students are "stuck" to the computer screen. More teaching tools are needed that provide for students to develop kinesthetic thinking through interaction with the "real" three dimensional world of motion. Perhaps dance is but one of these techniques. It has been found that dance training helps students to improve their drawing skills [14]. Perhaps it will also help students to design better geometric algorithms. Finally, it should be pointed out that computational geometric algorithms are usually taught as finished, logical, correct products even when they are discovered with kinesthetic heuristics that first yield incorrect algorithms that are later fixed. Students need to be taught to bridge this gap themselves. In the words of Ernst Mach:

> "An idea is best made the possession of the learner by the method by which it has been found."

References

[1] M.I. Shamos, "Computational Geometry," Ph.D. thesis, Yale University, 1978.

[2] F.P. Preparata, Ed., *Computational Geometry*, JAI Press, Inc., 1983.

[3] G.T. Toussaint, Ed., *Computational Geometry*, North Holland Publishing Co., 1984.

[4] G.T. Toussaint, "Pattern recognition and geometrical complexity," *Proc. 5th Int. Conf. on Pattern Recognition*, Miami Beach, 1980, pp. 1324-1347.

[5] G.T. Toussaint, "Computational geometric problems in pattern recognition," *Pattern Recognition Theory and Applications*, J. Kittler, ed., NATO Advanced Study Institute, Oxford University, April 1981.

[6] G. T. Toussaint, "Computational geometry," *Proceedings of the IEEE*, vol. 80, 1992, No. 9, pp. 1347-1363.

[7] J. Hadamard, *The Psychology of Invention in the Mathematical Field*, Princeton University Press, 1945.

[8] I. Lakatos, *Proofs and Refutations: The Logic of Mathematical Discovery*, Cambridge University Press, 1976.

[9] J. Pascual-Leone, "The forms of knowing in the psychological organism: reflections on Royce and Roseboom (eds.) Psychology of Knowing," *Phil. Soc. Sci.*, Vol. 6, 1976, pp. 175-181.

[10] H. Gardner, *Frames of Mind: The theory of Multiple Intelligences*, Basic Books Inc., 1983.

[11] P.N. Johnson-Laird and P.C. Wason, Eds., *Thinking*, Cambridge University Press, 1977.

[12] R. Arnheim, *Visual Thinking*, University of California Press, 1969.

[13] L. Pelland, "Kinesthetic stimulation as a method for improved drawing skill acquisition," Masters thesis, Concordia University, Montreal, 1980.

[14] L. Pelland, "La danse et le dessin," *Re-flex*, Vol. 3, No. 3, 1983, pp. 52-53.

[15] S. Papert, *MindStorms*, Basic Books Inc., 1980.

[16] B. Rosenberg and D. Landgridge, "A computational view of perception," *Perception*, Vol. 2, 1973, pp. 415-424.

[17] R.A. Jarvis, "On the identification of the convex hull of a finite set of points in the plane," *Infor. Proc. Letters*, Vol. 2, 1973, pp. 18-21.

[18] W.F. Eddy, "A new convex hull algorithm for planar sets," *ACM Trans. Math. Software*, Vol. 3, No. 4, 1977, pp. 398403 and 411-412.

[19] S.G. Akl and G.T. Toussaint, "Efficient convex hull algorithms for pattern recognition applications," *Proc. 4th Int. Joint Conf. on Pattern Recognition*, Kyoto, Japan, 1978, pp. 483-487.

[20] L. Devroye and G.T. Toussaint, "A note on linear expected time algorithms for finding convex hulls," *Computing*, Vol. 26, 1981, pp. 361-366.

[21] B.K. Bhattacharya and G.T. Toussaint, "A counter example to a diameter algorithm for convex polygons," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. PAMI4, May 1982 pp. 306-309.

[22] G.T. Toussaint, "Solving geometric problems with the rotating calipers," *Proceedings MELECON'83*, Athens, Greece, June 1983.

[23] A. Einstein, "Geometry and experience," Prussian Academy of Sciences, Berlin, January 27, 1921; in*Sidelights on Relativity*, Dover, 1983.

[24] W.B. Weimer, "A conceptual framework for cognitive psychology: motor theories of the mind," in *Perceiving, Acting and Knowing*, Eds., R. Shaw and J. Bransford, John Wiley 1977, pp. 267-311.

[25] S. Papert, *Turtle Geometry*, MIT Press, 1980.

[26] L. Guibas, L. Ramshaw, and J. Stolfi, "A kinetic framework for computational geometry," tech. rept., Xerox Park and Stanford University, May, 1983.

[27] E. Mach, *Space and Geometry*, The Open Court Publishing Co., Chicago, 1906.

[28] J. O'Rourke, *Art Gallery Theorems and Algorithms*, Oxford University Press, 1987.

[29] J. O'Rourke, *Computational geometry in C*, Cambridge University Press, 1994.

[30] G. T. Toussaint, Ed., *Computational Morphology*, North Holland, 1988.

[31] R.E. Mueller, *Iventivity*, E.M. Hale & Co., 1966. North Holland, 1988.

[32] J.L. Adams, *Conceptual Blockbusting*,San Francisco Book Co. Inc., 1976.

[33] A. Koestler, *The Act of Creation*, Picador, 1975.

[34] R. Sommer, *The Mind's Eye*, Dell Publishing Co., New York, 1978.

[35] M. Samuels and N. Samuels, *Seeing with the Mind's Eye*, Random House, 1975.

[36] D.A. Dondis, *A Primer of Visual Literacy*, M.I.T. Press, Cambridge, 1973.

[37] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, *Computational Geometry. Algorithms and applications*, Springer, 1997.

[38] D.D. Hoffman, *Visual Intelligence: How We Create What We See*, W. W. Norton and Co., 1998.

[39] Rotating Calipers page on the Web. http://www-cgrl.cs.mcgill.ca/ godfried/research/ calipers.html

[40] Lennes, N. J., "Theorems on the simple finite polygon and polyhedron," *American Journal of Mathematics*, vol. 33, 1911, pp.37-62.

[41] Ho, W.-C., "Decomposition of a polygon into triangles," *The Mathematical Gazette*, vol. 59, 1975, pp.132-134.

[42] Meisters, G. H., "Polygons have ears," *American Mathematical Monthly*, June/July 1975, pp. 648-651.

[43] Toussaint, G. T., "Efficient triangulation of simple polygons," *The Visual Computer*, Vol. 7, 1991, pp. 280-295.

[44] Thibaut, B. F., *Grundris der reinen Mathematik*, Gottingen, 1809.

[45] D. A. Singer, *Geometry: Plane and Fancy*, Springer-Verlag New York, Inc., 1998.

[46] S. G. Hoggar, *Mathematics for Computer Graphics*, Cambridge University Press, 1994.

[47] M. Johnson, *The Body in the Mind*, University of Chicago Press, 1987.

[48] D. Svanaes, "Kinesthetic thinking: the tacit dimension of interactive design," *Computers in Human Behavior*, Vol. 13, Bo. 4, Elsevier 1997, pp. 443-463.

[49] J.E. Goodman and J. O'Rourke, Ed., *Discrete and Computational Geometry*, CRC Press, 1997.

[50] J.R. Sack and J. Urrutia, Ed., *Handbook on Computational Geometry*, Elsevier 2000.

[51] U. Kortenkamp, "Foundations of Dynamic Geometry," Ph. D. Thesis, Swiss Federal Institute of Technology, Zurich, 1999