The Visual Computer (2020) 36:757–765
https://doi.org/10.1007/s00371-019-01657-y

ORIGINAL ARTICLE

Kinetic locally minimal triangulation: theoretical evaluation and combinatorial analysis

Tomáš Vomáčka[1] · Ivana Kolingerová[1] · Martin Maňák[1]

Published online: 3 May 2019

Abstract
Kinetic data structures represent an extension to ordinary data structures, where the underlying data become time dependent (e.g. moving points). In this paper, we define the kinetic locally minimal triangulation (KLMT) as a kinetic data structure extension to the locally minimal triangulation in the Euclidean plane. We explore the general properties of this data structure in order to show what types of events need to be considered during its life cycle; we also describe the predicates associated with these events. To describe the general kinetic features, we prove that KLMT is responsive, compact, efficient and non-local. In the combinatorial analysis of KLMT, we briefly describe the mathematical apparatus commonly used to investigate computational complexity properties of kinetic data structures and use it to establish the bounds on the number of events processed during the life cycle of this data structure. Finally, the obtained results are compared to the kinetic Delaunay triangulation showing that KLMT may provide some benefits over kinetic Delaunay triangulation, namely simplifying the mathematical equations that need to be computed in order to obtain the times of events.

Keywords Kinetic data structures · Delaunay triangulation · Locally minimal triangulation · Computational geometry · Planar graphs

1 Introduction

Kinetic locally minimal triangulation (KLMT) is a type of a kinetic data structure, a special type of data structures that are designed to be able to handle sets of time-dependent data. The field of application of such a tool is very wide—it may include tasks such as motion detection in different areas (air traffic monitoring, marine vessel navigation, robotics)—see [9,10,12,14], simulation of animal groups of various kinds or human crowds as shown in [11,15,17,21,25,26], and might even be exploited for wide range of mathematical problems that may be described with systems of partial differential equations and solved by finite element method such as in [6].

✉ Tomáš Vomáčka
tvomacka@kiv.zcu.cz

Ivana Kolingerová
kolinger@kiv.zcu.cz

Martin Maňák
manak@kiv.zcu.cz

[1] Faculty of Applied Sciences, University of West Bohemia, Pilsen, Czech Republic

It is very common to use kinetic Delaunay triangulation (KDT) or some of its *relatives* such as kinetic Voronoi diagram or kinetic regular triangulation for the aforementioned tasks, because they are closely related to various nearest neighbour graph structures that are commonly used [7,16,27]. However, we noticed that the KLMT has never been considered as an alternative and we think its properties should be explored. In this paper we focus on the theoretical properties of KLMT, expanding our work from [19] where we have discussed various types of proximity graphs for both static and kinetic applications, and we propose an idea that the kinetic applications could benefit from replacing KDT with KLMT. These data structures are very similar, and the replacement may provide some advantages over KDT because KLMT allows us to compute the events using equations of lower degree when compared to KDT, even despite the fact that KLMT management requires us to handle certain types of events that do not need to be considered when using KDT.

1.1 Locally minimal triangulation

We will start by the following definitions:

Definition 1 A triangulation $T(P)$ of a discrete set of points $P \subset \mathbb{R}^2$ is a maximal planar graph. That is, no edge between two points in P can be added to $T(P)$ without it becoming non-planar.

There are many different types of triangulations with different properties over any given set of points; let us now define the locally minimal triangulation:

Definition 2 A triangulation of a set of points $P = \{p_1, p_2, \ldots, p_n\}, n > 2, P \subset \mathbb{R}^2$, is locally minimal if and only if every edge $p_i p_j$ shared by two triangles $p_i p_j p_k$, $p_i p_j p_l$ forming a convex quadrilateral is not longer than the diagonal $p_k p_l$. It will be referred to as $LMT(P)$ [8].

It is worth mentioning that *LMT(P)* is not unique for P and it is dependent on the concrete method of its construction.

1.2 Delaunay triangulation

Another possible triangulation over a given set of points that we will explore in this article is the *Delaunay triangulation*—let us note that it is the most widely used triangulation in the field of kinetic data structures (see further) and we will use it as a base for an evaluation of the features of other kinetic data structures.

Definition 3 Delaunay triangulation $DT(P)$ of a set of points $P = \{p_1, p_2, \ldots, p_n\}, n > 2, P \subset \mathbb{R}^2$, is a triangulation in which no point $p_i \in P$ lies inside a circumcircle of any of the triangles in $T(P)$ [20].

Let $p_i = (x_i, y_i), \forall p_i \in P$, be the coordinates of the points in P. For practical reasons, it is necessary to be able to determine the position of a point $p_q \in P$ against a circumcircle of triangle $p_i p_j p_k \in P$.

The most commonly used way to do this is by using the incircle test [22], described by the following formula:

$$inCircle(p_q, p_i, p_j, p_k) = \det \begin{bmatrix} x_i & y_i & x_i^2 + y_i^2 & 1 \\ x_j & y_j & x_j^2 + y_j^2 & 1 \\ x_k & y_k & x_k^2 + y_k^2 & 1 \\ x_q & y_q & x_q^2 + y_q^2 & 1 \end{bmatrix} \quad (1)$$

If the triangle $p_i p_j p_k$ is oriented counterclockwise, a positive value means that p_q lies inside the circumcircle of $p_i p_j p_k$, a negative value means that it lies outside, and zero means that the point lies directly on the circumcircle.

Only one $DT(P)$ exists for any given set of points P as long as there are no four cocircular points in P, regardless of the construction algorithm used for computing the triangulation.

1.3 Triangulation construction algorithms

Many different types of construction algorithms exist with diverse properties that may be used to create various types of triangulations. These algorithms offer different time and space complexities, extensibility to higher dimensions, the possibility to be parallelized, etc. It is beyond the scope of this text to describe them all; let us just mention that we recommend to use the *incremental insertion*, because it is easy to implement and even though it does not usually provide the optimal time or space complexity, it generally allows us to introduce new points to our pre-existing data structures in run-time and may be easily modified to be able to remove the points in run-time as well. The data structures that allow the user to add and remove data at run-time are often called *dynamic*. More details on the topic of construction algorithms and properties of dynamic triangulations may be found in our previous work—[28].

1.4 Kinetic data structures

As shown in [4], any static data structure constructed over a finite set of primitives (points) may be proven valid by checking a finite number of conditions over these primitives. Let us define a set of basic terms that will help us understand the kinetic data structures in general and will be necessary later to describe their properties:

Primitive A primitive is an elementary object (such as a point) used as a basic input when creating a (kinetic) data structure. The primitive is defined by a set of coordinates and properties.
Predicate A function of a (sub)set of primitives which returns a discrete set of values. The result of this function is often reduced to determine the sign of an expression.
Certificate A check of a predicate made to determine the correctness of the given data structure.

Kinetic data structures (KDS) generalize static data structures by introducing movement to the primitives. In our case, we will focus on triangulations over a set of points. In this particular case, we define the kinetic primitives as "moving points", i.e. their coordinates are functions of time: $\forall p_i \in P : p_i(t) = (x_i(t), y_i(t))$. This may result in a wide variety of forms and applications. Note that the movement may be an arbitrary (continuous) function of time. However, it is common to limit the movement to *polynomial* functions of time since they are easy to solve (albeit not always analytically), simple enough to investigate and can be used to approximate any real-life scenario in the eventual applications. Since the primitives are time dependent, we need to define one more basic term that will allow us to investigate changes in topology. Because both the predicates and certificates for KDS are functions of time, the previously valid certificates may fail eventually with increasing values of time.

Event Time instants determined by a certificate failure are called *events*. The events are commonly divided into two types: *internal* and *external*. The difference is that external events directly affect the topology of the kinetic data structure, whereas internal events need to be processed for the KDS to work properly but do not change its topology.

As it has been shown in [4,23], maintaining a kinetic data structure requires us to compute and process the events in the correct order determined by the time in which they occur. The commonly used way to perform this task is by utilizing a priority queue for the events. The certificate failures are stored in the queue with the priority being the time when they occur. The life cycle of a kinetic data structure then usually consists of storing some events in the queue and then periodically popping some of them from the queue, computing new events, pushing them into the queue and so forth as needed.

1.5 The properties of kinetic data structures

According to [4,5,13], there are four basic properties that may be used to judge the effectiveness of a particular data structure for kinetic data. These are *responsiveness, efficiency, locality and compactness*. Note that these properties are dependent on the exact implementation of the given kinetic data structure, not only on its type and the property it is maintaining. For the purposes of the KDS properties description, let us denote that a given KDS contains n moving primitives. Let us also denote that a function is small if it is bounded by $O(\log^{\epsilon}(n))$ or $O(n^{\epsilon})$, for an arbitrarily small $\epsilon > 0$.

Note that any minor change in the kinetic data structure may influence quite a large number of certificates; it is often convenient to be able to determine the relationship between each primitive and the certificates it may directly affect.

As said before, the kinetic data structures use a priority queue for event management. The length of the queue may be then expressed as a function of the size of the input data n.

Responsiveness A responsive kinetic data structure is such a KDS that takes only a small amount of time to handle every certificate failure (an internal or external event). Handling of the event may includes changing the topology of the data structure, altering its properties, adding new certificate functions or removing old ones, etc. If expressed as a function of n, a KDS is responsive if the time required to handle every certificate failure is small.

(Weak) Efficiency KDS is said to be efficient if the total number of events processed in the worst case is asymptotically the same as (or slightly larger than) the number of external events processed in the worst case. Let us note that these two worst cases do not necessarily need to be the same cases (which is the reason that this property is called *weak* efficiency).

Strong Efficiency KDS is strongly efficient if and only if the ratio of all processed events to processed external events in the worst case is small for any given motion.

Locality A kinetic data structure is called local if the number of directly affected certificates is small for any input data.

Compactness A given data structure is compact if the maximum number of events ever present in the queue is small.

1.6 Kinetic triangulation events and properties

In order to explore the properties of external events in KDS, we need to be able to describe the changes in the topology that are associated with the events. In order to do that, let us extend the idea of graph *isomorphism* for the kinetic environment.

Let $T(P, t)$ denote the state at time t of triangulation $T(P)$, constructed over a set of n time-dependent points $P = \{p_1(t), \ldots, p_n(t)\}$, where $\forall p_i \in P : p_i(t) = (x_i(t), y_i(t))$. Let $E = \{e_1, \ldots, e_m\}$ be a set of m edges in $T(P)$. Let $\forall e_i \in E : e_i = (p_k, p_l)$ if e_i is the edge defined by points $p_k, p_l \in P$. Finally, let $E(t)$ be the set of edges in a kinetic triangulation at time t.

Definition 4 (*Kinetic Equality*): Given two triangulations $T_1(P, t_1)$, $T_2(P, t_2)$ constructed over the same set of (time-dependent) points, the triangulations are kinetically equal iff they are isomorphic.

Note that since the two graphs are constructed over the same set of points, the isomorphism condition is met iff $\forall (u, v) \in E_1 \Leftrightarrow (u, v) \in E_2$. We will denote this relation as $T_1(P, t_1) \cong T_2(P, t_2)$. In other words, we say that two kinetic triangulations constructed over the same set of points are kinetically equal if their topology is identical and the edges in those two triangulations connect the same pairs of points. Note that it is not necessary for the triangulations to have equal values of time in order to be kinetically equal and therefore the (moving) points do not need to be at the same positions.

Let $\mathbb{E} = \left(t_1^{ex}, \ldots, t_n^{ex}\right)$, $t_i^{ex} < t_{i+1}^{ex} \forall i < n$ be the set of all time values for which external events happen in $T(P, t)$. Formally, we can state that:

$$\forall t_i^{ex} \in \mathbb{E}\, \exists \varepsilon > 0 : T(P, t_i^{ex} - \varepsilon) \ncong T(P, t_i^{ex} + \varepsilon) \tag{2}$$

Equation 2 describes the life cycle of a kinetic triangulation over a time interval $[t_1^{ex}; t_n^{ex}]$ and highlights the fact that external events change the topology of the triangulation as the set of edges defined by the triangulation changes with each external event. It is also worth mentioning that some of the states of $T(P, t)$ may be topologically equal to other states that existed in the triangulation for earlier values of t—this is the whole purpose of using the kinetic approach to handle data structures over sets of time-dependent data. By computing the times of external topologic events, we are able to determine every single different state of the data structure during its life cycle without the risk of missing any of them. The internal events are more complicated to describe, and we shall discuss them later in this text with a concrete data structure to provide an example of their effect.

1.7 Combinatorial tools

It has been shown in [4,24] that the combinatorial analysis of the kinetic properties of different kinetic data structures may be converted to the problem of analysing the upper envelope of a set of their predicate functions. The number of changes in the "topmost" function in the upper envelope is then closely connected to the length of the corresponding Davenport–Schinzel sequence. This sequence is defined as follows:

Definition 5 (*Davenport–Schinzel Sequence*) [24]: Let n, s be two positive integers. A sequence $\sigma = (\sigma_1, \ldots, \sigma_m)$ of integers is an (n, s)-Davenport–Schinzel sequence if it is non-repeating and satisfies the following conditions:

1. $\forall i : 1 \leq \sigma_i \leq n$
2. Any non-repeating subsequence of σ made of only two integers is of length at most $s + 1$.

Let us denote the length of the longest (n, s)-Davenport–Schinzel sequence by $\lambda_s(n)$. It has been shown in [24] that the tight bounds on the value of $\lambda_s(n)$ involve the Ackermann function.

Definition 6 (*Ackermann Function*) [24]:

$$A(m, n) = \begin{cases} n + 1 & \text{if } m = 0 \\ A(m-1, 1) & \text{if } n = 0 \\ A(m-1, A(m, n-1)) & \text{otherwise} \end{cases}$$

Let us also denote that $A(n) = A(n, n)$ and that $\alpha(n)$ is the inverse function of $A(n)$. It has been shown in [24] that $\alpha(n) \leq 4$ for any practical values of n. For the purposes of this text, we will need the following values of $\lambda_s(n)$:

Theorem 1 (Values of $\lambda_s(n)$)

$$\begin{aligned}\lambda_1(n) &= n \\ \lambda_2(n) &= 2n - 1 \\ \lambda_3(n) &= \Theta(n\alpha(n)) \\ \lambda_4(n) &= \Theta(n \cdot 2^{\alpha(n)})\end{aligned}$$

More details on Davenport–Schinzel sequences, Ackermann function and its inverse with respect to kinetic data structures can be found in [3,24].

2 Kinetic locally minimal triangulation

2.1 Predicates and certificates

As given in Definition 2, *LMT(P)* is proven valid by comparing the lengths of the two possible diagonals in convex quadrilaterals; this gives us two different predicates that need to be checked—the length comparison itself and the convexity check of the four points. Let $P_Q = \left(p_i, p_j, p_k, p_l\right)$, $P_Q \subseteq P$, be a subset containing four points which create two adjacent triangles. From the kinetic point of view, every such quadrilateral in *LMT(P)* will be checked, thus creating two certificates: one for the local minimum $LM(P_Q)$ and the other for convexity $CH(P_Q)$:

$$LM(P_Q) = |p_i - p_j| - |p_l - p_k| \tag{3}$$

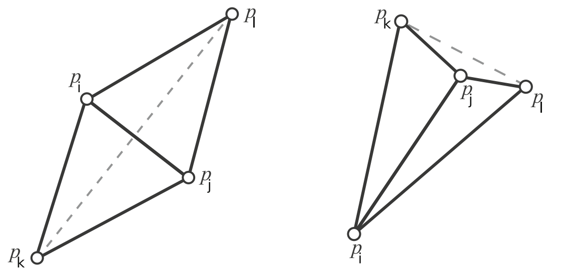
$$O(p_i, p_j, p_k) = \operatorname{sgn}\left(\det \begin{bmatrix} x_i & y_i & 1 \\ x_j & y_j & 1 \\ x_k & y_k & 1 \end{bmatrix}\right) \tag{4}$$

$$\begin{aligned} CH(P_Q) = O(p_i, p_j, p_l) > 0 \wedge O(p_j, p_l, p_k) > 0 \\ \wedge O(p_l, p_k, p_i) > 0 \wedge O(p_k, p_i, p_j) > 0 \end{aligned} \tag{5}$$

The certificate in (3) compares the lengths of the two possible diagonals inside the quadrilateral P_Q and from the practical point of view; it makes perfect sense to simplify it by comparing square lengths, therefore simplifying the computation by removing the square root operations. The second certificate in (5) is necessary because the check for a shorter diagonal in any given quadrilateral is valid only if it is convex. In order to determine if the convexity condition is met, we need to compute the value of (4) for each triplet of vertices in P_Q. This equation determines the orientation (clockwise or counterclockwise) of the vertex and its two neighbours.

Basic examples of the two cases described earlier are provided in Fig. 1 with a convex quadrilateral in Fig. 1a and a non-convex quadrilateral in Fig. 1b. You can see that the locally minimal (i.e. shorter) edge is chosen for the convex configuration of points but not for the non-convex one.

Let us introduce point movement into the data structure by defining that $\forall p_q \in P : \; p_q(t) = (x_q(t), y_q(t))$. We can



(a) Convex quadrilateral **(b)** Nonconvex quadrilateral

Fig. 1 Possible quadrilateral configurations

see that the certificates defined in the previous text are now functions of time. If we limit the point movement so that $x_q(t)$, $y_q(t)$ are polynomials, the certificates will be polynomials of time as well. This restriction is quite common in kinetic data structure research as it provides a reasonably robust way of approximating more sophisticated trajectories and a good platform for further analysis and comparison with other kinetic data structures.

In the following text, we will illustrate the colinear and topologic events in KLMT. Let us take a closer look at a non-convex quadrilateral of four points just like the one we describe in Fig. 1b except that the point p_j will now move along a linear trajectory as shown in Fig. 2. As we can see, the edge $p_k p_l$ is clearly shorter than the edge $p_i p_j$ from the very start of our analysis as shown in Fig. 2a. As described in the previous text, we can see that the common edge shared by the two triangles cannot be swapped because the quadrilateral is non-convex. Due to the movement of p_j, we can see in Fig. 2b that at a certain point in time, points p_j, p_k, p_l become colinear and the quadrilateral becomes convex. We will call this type of event a *colinear event*. However, the edge still cannot be swapped because the quadrilateral is now in a singular state and the edge should formally appear immediately after p_j moves away from the line $p_k p_l$, the result of this edge swap is shown in Fig. 2b, this is called a *topologic event* which is in this particular case represented by a swap of the diagonal in the quadrilateral.

Note that the colinear event is an internal event because it does not change the topology of the data structure but is necessary to be processed in order to schedule and de-schedule the topologic events, which, on the other hand, are *external events* because they result in topology changes represented by the edge swaps as described above. Generally, the two types of events are not directly tied together, and it is not necessary for the edge swap to follow immediately after the colinear event, and in some configurations, it will not happen at all. An example of such a configuration is shown in Fig. 3. The length of the edge $p_i p_j$ is shorter than the length of edge $p_k p_l$ for any value of time. In this case, the colinear events will be scheduled but no topologic change will occur in the triangulation.

Figure 4 shows another significant case. We can see that in the initial state, the length of edge $p_i p_j$ is shorter than $p_k p_l$ and the quadrilateral is non-convex. Due to the movement of p_j, the two edges become of equal length when p_j reaches the position of p'_j and finally $p_i p_j$ becomes longer than $p_k p_l$, here represented by the position of p''_j. During this whole period of time, the quadrilateral remains non-convex and therefore no topology change can be performed on the triangulation even though a topologic event would be scheduled at the time when $p_j = p'_j$.

2.2 Combinatorial analysis

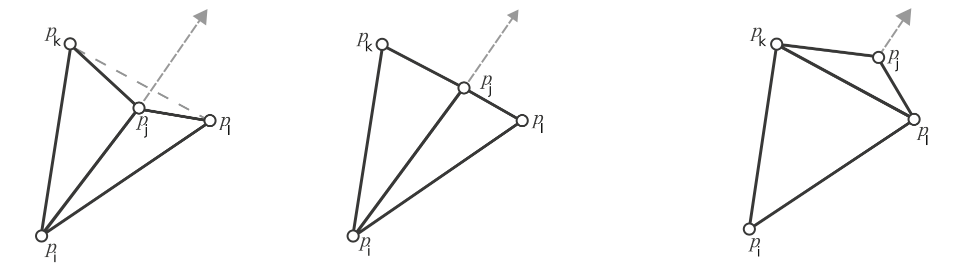
We shall now explore the basic kinetic data structure properties of KLMT as defined in the previous section. First, let us consider the *responsiveness* and *compactness*.

Lemma 1 *KLMT is responsive.*

Proof There are two different types of events in KLMT: *colinear* and *edge swap*. To prove that KLMT is responsive, we need to show that handling them will take only a small amount of time:

Colinear Since this event is internal, it will not directly affect the data structure in any way. However, if the affected quadrilateral was non-convex prior to this event, it becomes convex and therefore valid for topologic

Fig. 2 Events in KLMT



(a) Initial configuration **(b)** Colinear event occurs **(c)** Topologic event occurs

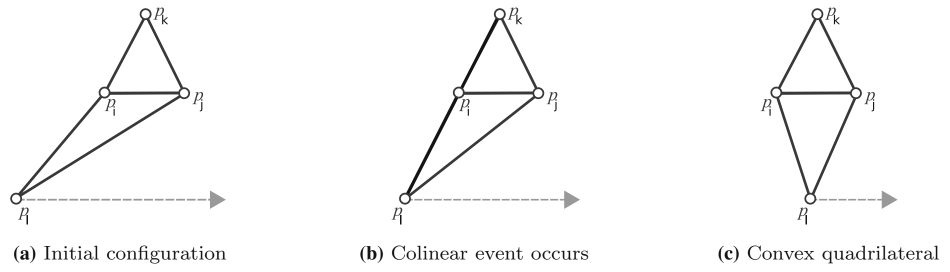


Fig. 3 Non-convex configuration without external events

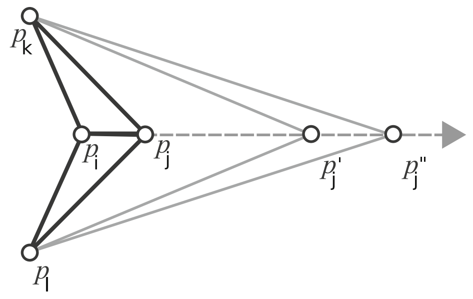


Fig. 4 Non-local case of KLMT

events. Because of that, we need to schedule up to one topologic event for the newly convex quadrilateral. And vice versa if the quadrilateral was convex prior to the colinear event, we need to de-schedule up to one topologic event that might have been scheduled for the quadrilateral.

Edge Swap This is an external event because swapping the inner edge in a convex quadrilateral will change the topology of KLMT. The result of the edge swap is up to 5 new quadrilaterals being formed in the triangulation. (One is created by the edge swap, and the other four may be created from the newly created triangles and the triangles directly adjacent to the quadrilateral.) For each of those quadrilaterals we need to schedule up to one new event (either colinear or topologic, depending on the actual current state of the quadrilateral). Also, up to four events may be de-scheduled, because the quadrilaterals that were used to compute them do not exist any more as a result of the edge swap.

Each of the aforementioned operations is independent on the total number of points in the triangulation; therefore, we can see that the amount of time required to handle the events in KLMT is $O(1)$. Also, the topologic changes performed as a result of handling an external event are local and will only affect the related quadrilateral. □

Note that the movement of points may affect the topology of multiple quadrilaterals at the same time. If such circumstances occur, it is vital to know that every single topologic change that is to happen in the data structure will be accompanied by a corresponding event and every single processed topologic event will result in one change in topology of the data structure.

Lemma 2 *KLMT is compact.*

Proof To prove that KLMT is compact, we need to show that the number of events ever present in the queue is small. There is only a constant number of events that can be scheduled for any given edge present in the triangulation represented by the roots of the certificate function for the given quadrilateral. When an edge is removed from the triangulation as a result of handling a topologic event, the connected events are also removed from the event priority queue. And since the number of edges in $KLMT(P);\ P = \{p_1, p_2, \ldots, p_n\}$ is $O(n)$ at any given time, we can see that KLMT is compact. □

Before we proceed to the evaluation of KLMT efficiency, we need to be able to determine the worst-case number of external events this data structure can process during its lifetime. Let us state that $\lambda_s(n)$ is the maximum length of (n, s)-Davenport–Schinzel sequence (see [2]).

As shown in [2], the number of topologic events in a kinetic Delaunay triangulation in *two*-dimensional Euclidean space is bounded by $O(n^2\lambda_s(n))$, if each subset of P of size 4 generates at most a constant number s of external events. The same proof is also valid for KLMT: since we limited the movement of the points in P to polynomial trajectories, (3) will be a polynomial and therefore provide a constant number of roots (which correspond with the topologic events). We can also see that similarly to KDT, the topologic events are also bound to a single quadrilateral and leave its bounding edges unchanged.

Corollary 1 *The upper bound on the total number of external (edge swap) events in* KLMT(P) *is* $O(n^2\lambda_s(n))$.

Theorem 2 *The number of internal (colinear) events processed by* KLMT(P) *is bounded by* $O(n^2\lambda_s(n))$.

Proof As stated in the previous text, internal events do not change the topology of the kinetic data structure by their definition. Therefore, if the number of possible internal events on each quadrilateral is constant, the total number of internal events is bounded by the number of external events in the data structure because new quadrilaterals can only be formed

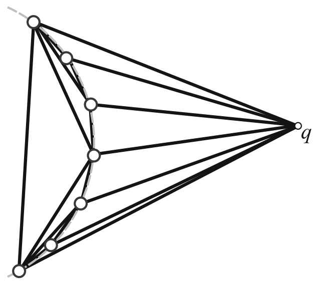


Fig. 5 Non-local case of KLMT

as a result of handling external (edge swap) events. We can see in (5) that the number of internal events for any given quadrilateral is equal to the number of roots of a polynomial equation, therefore independent of the size of P and therefore constant. □

It was also shown in [1] that the convex hull of a moving point set may change $\Theta(n^2)$ times, which implies a lower bound of $\Omega(n^2)$ topologic events in any triangulation constructed over a set of moving points.

Lemma 3 *KLMT is strongly efficient.*

Proof From Theorem 2 we can see that the upper bounds on internal (colinear) and external (topologic) events are $O(n^2\lambda_s(n))$. Corollary 1 shows that the number of external events in *KLMT(P)* is bounded by $O(n^2\lambda_s(n))$. Let us write the ratio of the number of all processed events to the number of processed external events as follows:

$$\frac{O(n^2\lambda_s(n)) + O(n^2\lambda_s(n))}{O(n^2\lambda_s(n))} = \frac{O(n^2\lambda_s(n))}{O(n^2\lambda_s(n))}$$

$$\frac{O(n^2\lambda_s(n))}{O(n^2\lambda_s(n))} \subset O(\log^\epsilon n)$$

We can see that the ratio is *small* as defined in the previous text and therefore *KLMT(P)* is strongly efficient. □

Lemma 4 *KLMT is not local.*

Proof Consider the triangulation in Fig. 5—all the points in KLMT but q are placed on a circular arc. The triangulation is locally minimal, and no edge swaps can be performed because the quadrilaterals containing q are non-convex, yet q is connected to all other points in P, the number of which is bounded by $\Theta(n)$. Therefore, q is included in every single certificate that will occur in *KLMT(P)* and any change of q will affect all the enqueued certificates. □

3 Comparison to kinetic Delaunay triangulation

3.1 Basic properties comparison

As shown in [3], kinetic Delaunay triangulation (KDT) has the following properties:

- KDT is responsive—managing an event takes almost constant time for any event.
- KDT is strongly efficient—only events of one type exist, and thus, all the events are external. Therefore, the ratio of all processed events to processed external events is $\Theta(1)$.
- KDT is not local—each moving point may be connected to a relatively large number of neighbours and therefore participate in a relatively large number of events.
- KDT is compact—although the precise relations between the size of the input set, the allowed type of movement and the size of the event queue are not known, compactness of KDT has been shown—see [2,13].

As we can see, both triangulations are responsive, strongly efficient, compact and non-local. In the case of the strong efficiency, it is worth noting that in the case of KDT, there are only external events of one type (cocircular events), whereas in KLMT there are both external (topologic) and internal (colinear) events, but we have shown that the ratio between internal and external events in this data structure is small.

3.2 Combinatorial comparison

It has been shown in [2] that the maximum number of external events in *KDT(P)* is $O(n^d\lambda_s(n))$ using the same notation we used in the proof of Theorem 2. Let us explore the following lemma, which was first published in [18].

Lemma 5 *The computation of certificates for external events in* KDT(P) *is at least as computationally complex as for* KLMT(P) *constructed over the same set of kinetic points moving along polynomial trajectories of degree up to* $R > 0$, *where* $R \in \mathbb{N}$.

Proof We can see from (1) that in order to compute the predicates for *KDT(P)*, we need to search for the roots of polynomials of time which are of degree up to $4R$. There are two different types of events in *KLMT(P)*; from Eq. (3) we can see that in case of *external* events, we only need to solve polynomials of degree up to $2R$; Eq. (5) shows that the computation of *internal* events requires us to solve polynomials of degree up to $2R$. □

From the practical point of view, it may be worth mentioning that if the points move along linear trajectories as stated before, their time-dependent coordinates are described by polynomials $x_i(t)$ and $y_i(t)$ of degree one. This results in solving quadratic equations for KLMT and fourth-degree polynomials for KDT. Note that solving fourth-degree polynomials usually cannot be done analytically which further increases the overall complexity of maintaining the KDT, especially when compared to KLMT. Generally speaking, for polynomial trajectories of any degree, the resulting certificate computation in KLMT will require us to solve polynomials of lower degree than the same task performed with KDT.

Theorem 3 *Maintaining* KDT(P) *is at least as complex as maintaining* KLMT(P) *over the same set of points moving along linear trajectories in the Euclidean plane.*

Proof Since handling each event takes constant time (topologic and colinear) in both *KDT(P)* and *KLMT(P)*, we need to show that the number of processed events in *KDT(P)* is at least as high as the number of processed events in *KLMT(P)*. We have shown that the maximum processed number of events of any type in both the considered data structures is $O(n^d \lambda_s(n))$. For example, if we limit the movement of the points in P to linear trajectories in the Euclidean plane, we can see that the maximum number of roots of (1) is 4 and because $\lambda_4(n) = \Theta(n \cdot 2^{\alpha(n)})$, the upper bound of topologic events in *KDT(P)* is $O(n^3 \cdot 2^{\alpha(n)})$, where $\alpha(n)$ is the inverse Ackermann function.

Similarly, we can see that the maximum number of roots of (3) and of (5) is 2 and since $\lambda_2(n) = 2n - 1$, the upper bound on both the number of topologic and colinear events in *KLMT(P)* is $O(n^3)$. We can see that the task of maintaining the two compared kinetic data structures is asymptotically the same and equal to $O(n^3)$ in both cases. However, since $n \cdot 2^{\alpha(n)} > 2n - 1; \forall n > 0$, we can see that the upper bound on the number of processed events in *KDT(P)* is in fact higher than the upper bound on the total number of events in *KLMT(P)*. □

4 Conclusion

Using the numerical apparatus published in [24], we have shown that the bounds on the number of events processed during the life cycle of KLMT in two-dimensional Euclidean space are $O(n^2\lambda_s(n))$, which is very similar to the case of KDT. However, the certificates used to compute the events are generally simpler in the case of KLMT which in turn results in tighter bounds on the number of events. Given the example of linear trajectories in the Euclidean plane, the total number of events in KLMT is bounded by $O(n^3)$ which is smaller for any values of n than $O(n^3 \cdot 2^{\alpha(n)})$, the respective bound for the number of events in KDT. All the discussed events can be handled in $O(1)$, and therefore, the management of KLMT should be computationally simpler than the management of KDT. From the practical point of view, it is necessary to note the fact that the management of KDT only requires handling of one type of external events, while the management of KLMT requires us to handle internal events as well and this may lead to increased numerical instability. The practical comparison of the two data structures should be done as part of our future work.

Acknowledgements The authors would like to thank to RNDr. Svetlana Tomiczkova, Ph.D., from the Department of Mathematics of the Faculty of Applied Sciences at the University of West Bohemia for her invaluable help and proofreading of the text.

Funding This work was funded by the Czech Science Foundation (Project Number 17-07690S) and by the Czech Ministry of Education, Youth and Sports (Project Number SGS-2019-016).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Agarwal, P.K., Basch, J., de Berg, M., Guibas, L.J., Hershberger, J.: Lower bounds for kinetic planar subdivisions. Discr. Comput. Geom. **24**(4), 721–733 (2000). https://doi.org/10.1007/s004540010060
2. Albers, G., Guibas, L.J., Mitchell, J.S.B., Roos, T.: Voronoi diagrams of moving points. Int. J. Comput. Geom. Appl. **8**(3), 365–380 (1998). citeseer.ist.psu.edu/albers95voronoi.html
3. Basch, J.: Kinetic data structures. Ph.D. thesis, Stanford University (1999)
4. Basch, J., Guibas, L.J., Hershberger, J.: Data structures for mobile data. J. Algorithms **31**(1), 1–28 (1999)
5. Basch, J., Guibas, L.J., Silverstein, C.D., Zhang, L.: A practical evaluation of kinetic data structures. In: Proc. 13th Annu. ACM Sympos. Comput. Geom, pp. 388–390. ACM Press (1997)
6. Beni, L.H.: Development of a 3D kinetic data structure adapted for a 3D spatial dynamic field simulation. Ph.D. thesis, Université Laval, Québec (2009)
7. Bose, P., Devroye, L., Evans, W.: Diamonds are not a minimum weight triangulation's best friend. Int. J. Comput. Geom. Appl. **12**(06), 445–453 (2002). https://doi.org/10.1142/S0218195902000979
8. Dickerson, M.T., Montague, M.H.: A (usually?) connected subgraph of the minimum weight triangulation. In: Proceedings of the Twelfth Annual Symposium on Computational Geometry, SCG '96, pp. 204–213. ACM, New York (1996). https://doi.org/10.1145/237218.237364
9. Erickson, J., Guibas, L.J., Stolfi, J., Zhang, L.: Separation-sensitive collision detection for convex objects. In: SODA '99: Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms, pp. 327–336. Society for Industrial and Applied Mathematics, Philadelphia (1999)
10. Gavrilova, M., Rokne, J., Gavrilov, D.: Dynamic collision detection in computational geometry. In: 12th European Workshop on Computational Geometry, pp. 103–106. Munster, Germany (1996)
11. Goldenstein, S., Karavelas, M.I., Metaxas, D.N., Guibas, L.J., Aaron, E., Goswami, A.: Scalable nonlinear dynamical systems for agent steering and crowd simulation. Comput. Graph. **25**(6), 983–998 (2001)

12. Goralski, I.R., Gold, C.M.: Maintaining the spatial relationships of marine vessels using the kinetic Voronoi diagram. In: ISVD '07: Proceedings of the 4th International Symposium on Voronoi Diagrams in Science and Engineering, pp. 84–90. IEEE Computer Society, Washington (2007). https://doi.org/10.1109/ISVD.2007.30
13. Guibas, L.J.: Kinetic data structures: a state of the art report. In: WAFR '98: Proceedings of the Third Workshop on the Algorithmic Foundations of Robotics on Robotics: The Algorithmic Perspective, pp. 191–209. A. K. Peters, Ltd., Natick, MA, USA (1998)
14. Guibas, L.J., Xie, F., Zhang, L.: Kinetic collision detection: algorithms and experiments. In: ICRA, pp. 2903–2910 (2001)
15. Heigeas, L., Luciani, A., Thollot, J., Castagné, N.: A physically-based particle model of emergent crowd behaviors. In: Graphicon (2003). http://artis.imag.fr/Publications/2003/HLTC03
16. Jaromczyk, J.W., Toussaint, G.T.: Relative neighborhood graphs and their relatives. Proc. IEEE **80**(9), 1502–1517 (1992). https://doi.org/10.1109/5.163414
17. Kamphuis, A., Overmars, M.H.: Finding paths for coherent groups using clearance. In: SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 19–28. Eurographics Association, Aire-la-Ville, Switzerland (2004). https://doi.org/10.1145/1028523.1028526
18. Kolingerová, I., Ferko, A., Vomáčka, T., Maňák, M.: Nearest Neighbour Graph and Locally Minimal Triangulation, pp. 455–464. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-62395-5_31
19. Kolingerová, I., Vomácka, T., Manak, M., Ferko, A.: Neighbourhood graphs and locally minimal triangulations. Trans. Comput. Sci. **33**, 115–127 (2018). https://doi.org/10.1007/978-3-662-58039-4_7
20. Okabe, A., Boots, B., Sugihara, K., Chiu, S.N.: Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. Probability and Statistics, 2nd edn. Wiley, Hoboken (2000)
21. Overmars, M.: Practical algorithms for path planning and crowd simulation. In: Proceedings of the 25th European Workshop on Computational Geometry (EuroCG), p. 263. Brussels, Belgium (2009). Invited speech
22. Richard Shewchuk, J.: Adaptive precision floating-point arithmetic and fast robust geometric predicates. Discr. Comput. Geom. **18**(3), 305–363 (1997). https://doi.org/10.1007/PL00009321
23. Russel, D.: Kinetic data structures in practice. Ph.D. thesis, Stanford, CA, USA (2007). Adviser-Guibas, Leonidas
24. Sharir, M., Agarwal, P.K.: Davenport–Schinzel Sequences and Their Geometric Applications. Cambridge University Press, Cambridge (1995)
25. Sud, A., Andersen, E., Curtis, S., Lin, M., Manocha, D.: Real-time path planning for virtual agents in dynamic environments. In: SIGGRAPH '08: ACM SIGGRAPH 2008 classes, pp. 1–9. ACM, New York (2008)
26. Treuille, A., Cooper, S., Popović, Z.: Continuum crowds. In: SIGGRAPH '06: ACM SIGGRAPH 2006 Papers, pp. 1160–1168. ACM, New York (2006). https://doi.org/10.1145/1179352.1142008
27. Veltkamp, R.: The gamma-neighborhood graph. Comput. Geom. **1**, 227–246 (1991)
28. Vomáčka, T., Kolingerová, I.: Computation of topologic events in kinetic Delaunay triangulation using sturm sequences of polynomials. In: SIGRAD 2008, pp. 57–64. University Electronic Press, Linköping (2008)

Tomáš Vomáčka is a Ph.D. student at the University of West Bohemia, Czech Republic, and a member of the Centre of Computer Graphics and Visualization at this institution. The main topic of his research is kinetic data structures with special focus on planar triangulations—exploring their properties and potential applications in different areas of industry ranging from VR pedestrian navigation to early warning systems for air traffic control.

Ivana Kolingerová is a full-time professor at the Department of Computer Science and Engineering at the University of West Bohemia in Pilsen, Czech Republic. Her research interests include computer graphics and applied computational geometry, especially algorithms for triangulated models. Her international experience includes Denmark, the USA, Slovenia and Austria.

Martin Maňák is a researcher at the NTIS Research Centre in Pilsen, Czech Republic. He received his M.Sc degree in Computer Science from the Charles University in 2008 and the Ph.D. degree from the University of West Bohemia in 2016. His research interests include nonlinear Voronoi diagrams, techniques for the visualization and analysis of molecular structures, computational geometry and computer graphics.