TRIANGULATIONS

- Types of triangulations
- Greedy triangulation
- Delaunay triangulation
- Triangulation of polygons

Author: prof. Yevhenii Borodavka

Types of triangulations

In geometry, a triangulation is a subdivision of a planar object into triangles. We will consider point-set triangulation and polygon triangulation.

A point-set triangulation, i.e., a triangulation of a discrete set of points **P**, is a subdivision of the convex hull of the points into simplices such that any two simplices intersect in a common face of any dimension or not at all and such that the set of vertices of the simplices are contained in **P**.

There are many algorithms for point-set triangulation. The result of this type of the triangulation is at least the set of bounded edges or the set of triangles (TIN-model). The maximal number of edges for the set of N points is (3*N-6).

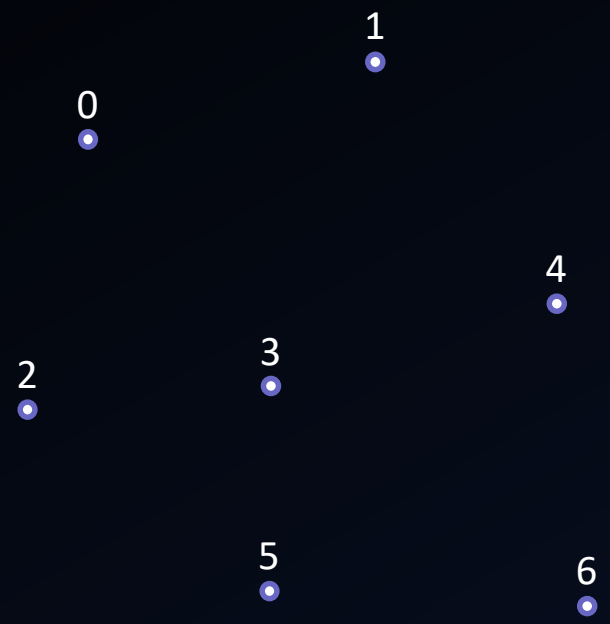
Greedy triangulation

The greedy triangulation is the simplest algorithm of the triangulation with complexity $O(n^2*\log(n))$. During processing we do not reject any action have done before. The algorithm steps are below.

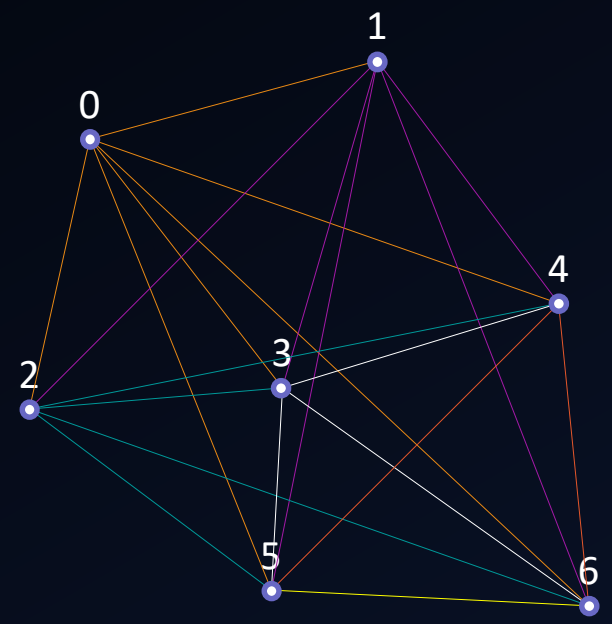
1. The square of the distance to other points is determined for each point.
2. The list of edges created in this way (R_0) is sorted by increasing edge length.
3. The first edge (with the minimum length) is recorded in the list of edges of the TIN model (R_T).
4. For each edge of the set R_0, an intersection test with the edges of the set R_T is performed. If an edge does not intersect with the edges of the set R_T, then it is added to the set R_T. For N points, the process ends when the number of edges in the set R_T is reached (3*N-6) or after testing all edges of the set R_0.
5. Triangles are formed on the set of R_T edges by finding the edges incident to each point of the given set and adjacent edges to them. Each triangle in this model is presented in the form of (i,j,k) — the indices of the initial points, which are the vertices of the triangle.

GREEDY TRIANGULATION

An example. STEP 1. Preparing list of sorted edges R_0.



Set of points



Distances

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
20	1 - 6	6.9
21	0 - 6	7.1

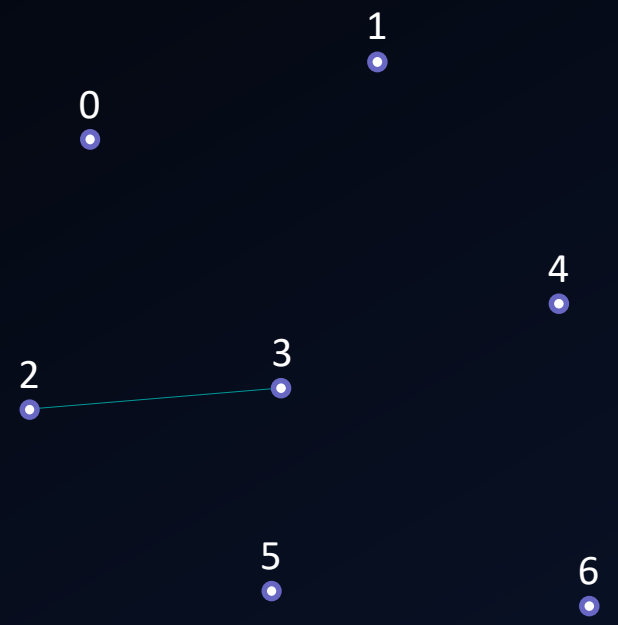
R_0

GREEDY TRIANGULATION

An example. STEP 2. LOOP 1. Selecting edges from the R_0 and adding to the R_T.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
…		
20	1 - 6	6.9
21	0 - 6	7.1

R_0



Checking cross

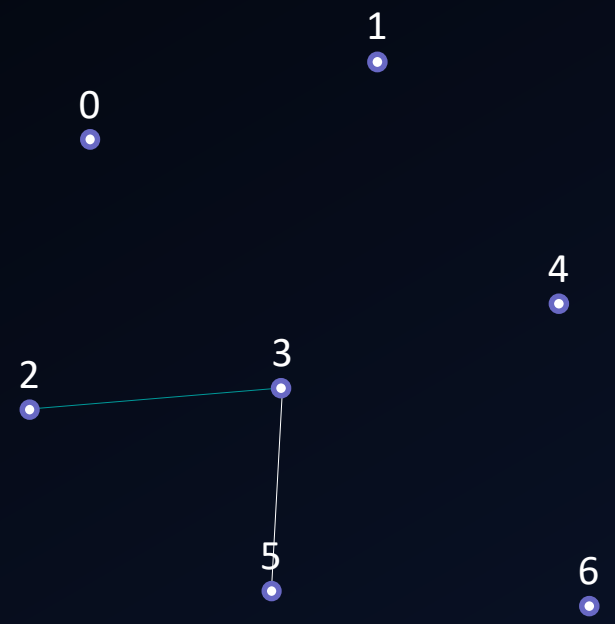
#	EDGE	LENGHT
1	2 - 3	2.1
2		
3		
4		
5		
6		
…		
11		
12		

R_T

GREEDY TRIANGULATION

An example. STEP 2. LOOP 2. Selecting edges from the R_0 and adding to the R_T.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
20	1 - 6	6.9
21	0 - 6	7.1



#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3		
4		
5		
6		
...		
11		
12		

R_0

Checking cross and accept 3-5

R_T

GREEDY TRIANGULATION

An example. STEP 2. LOOP 3. Selecting edges from the R_0 and adding to the R_T.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
20	1 - 6	6.9
21	0 - 6	7.1

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4		
5		
6		
...		
11		
12		

R_0

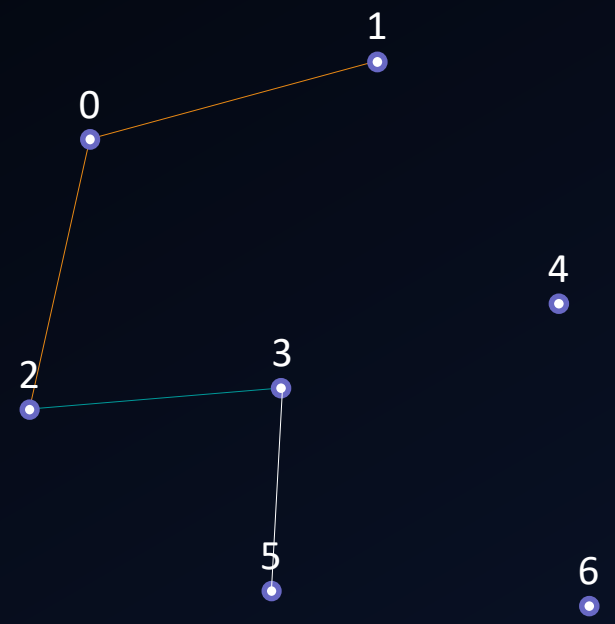
Checking cross and accept 0-2

R_T

GREEDY TRIANGULATION

An example. STEP 2. LOOP 4. Selecting edges from the R_0 and adding to the R_T.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
20	1 - 6	6.9
21	0 - 6	7.1



#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5		
6		
...		
11		
12		

R_0

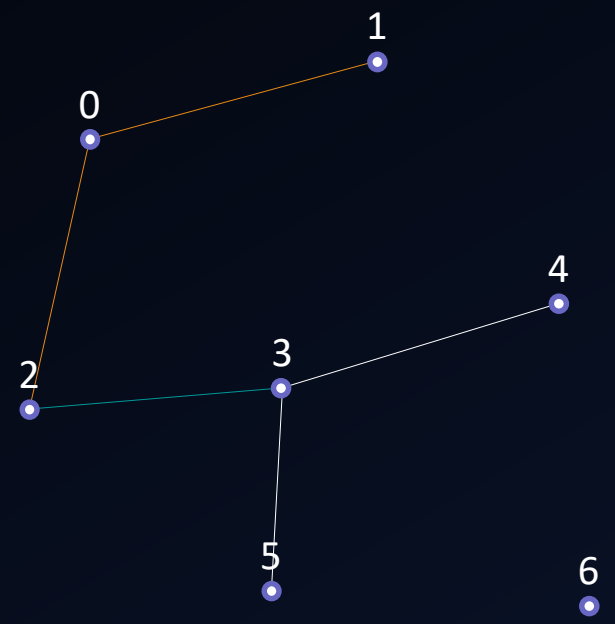
Checking cross and accept 0-1

R_T

GREEDY TRIANGULATION

An example. STEP 2. LOOP 5. Selecting edges from the R_0 and adding to the R_T.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
…		
20	1 - 6	6.9
21	0 - 6	7.1



#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6		
…		
11		
12		

R_0

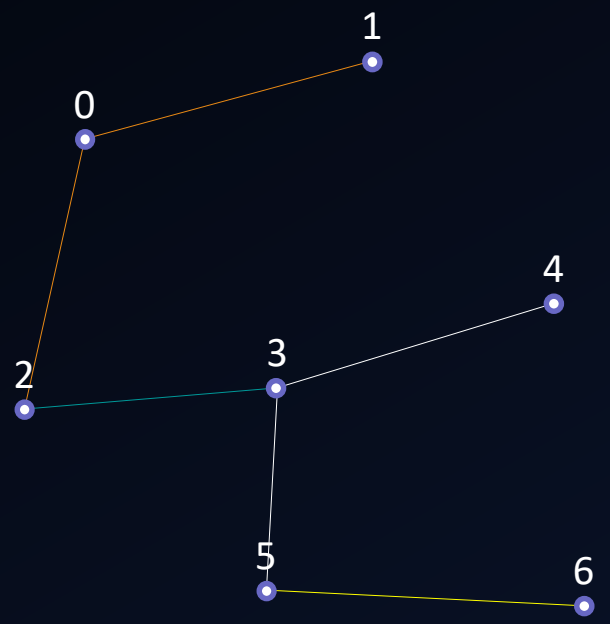
Checking cross an d accept 3-4

R_T

Greedy triangulation

An example. STEP 2. LOOP 6. Selecting edges from the R_0 and adding to the R_T.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
…		
20	1 - 6	6.9
21	0 - 6	7.1



#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
…		
11		
12		

R_0

Checking cross and accept 5-6

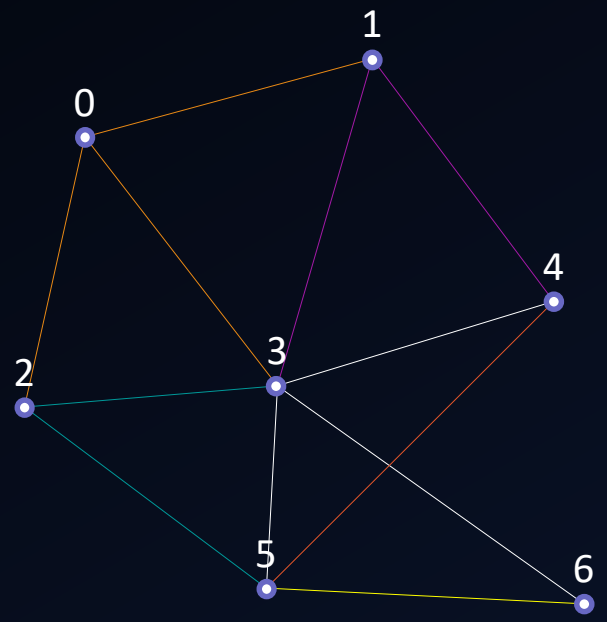
R_T

GREEDY TRIANGULATION

An example. STEP 2. LOOP 12. Selecting edges from the R_0 and adding to the R_T.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
20	1 - 6	6.9
21	0 - 6	7.1

R_0



Checking cross and reject 4-5

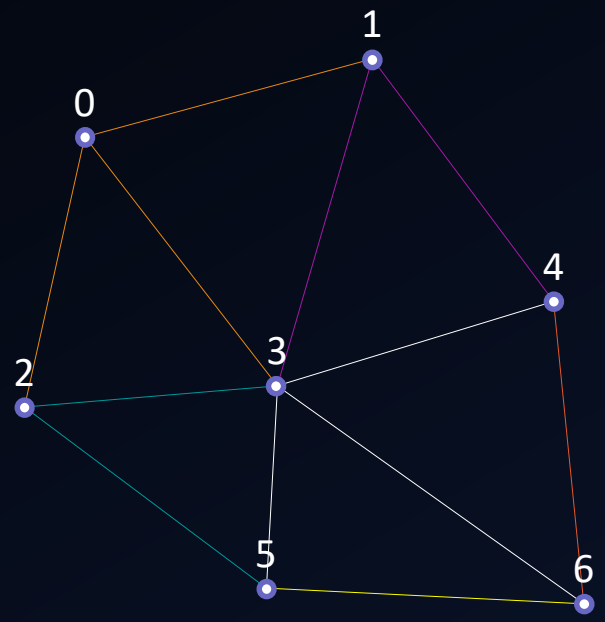
#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12		

R_T

GREEDY TRIANGULATION

An example. STEP 2. LOOP 12. Selecting edges from the R_0 and adding to the R_T.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
20	1 - 6	6.9
21	0 - 6	7.1



#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8

R_0

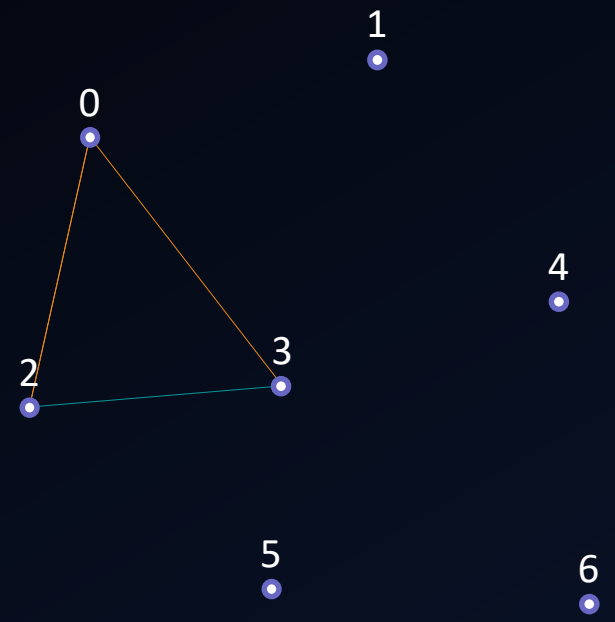
Checking cross and accept 4-6 and we are reached (3*N-6) edges in R_T

R_T

GREEDY TRIANGULATION

An example. STEP 3. LOOP 1. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
…		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2			
3			
4			
5			
6			

R_T

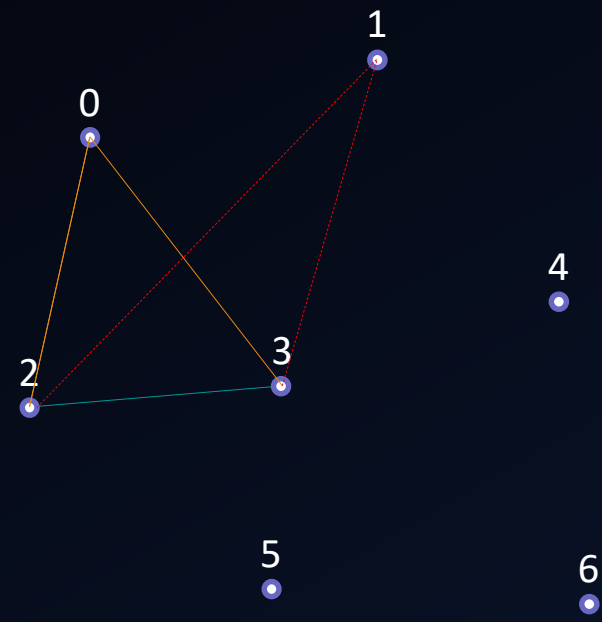
Check point 0 and find edges 0-2 and 0-3

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 2. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



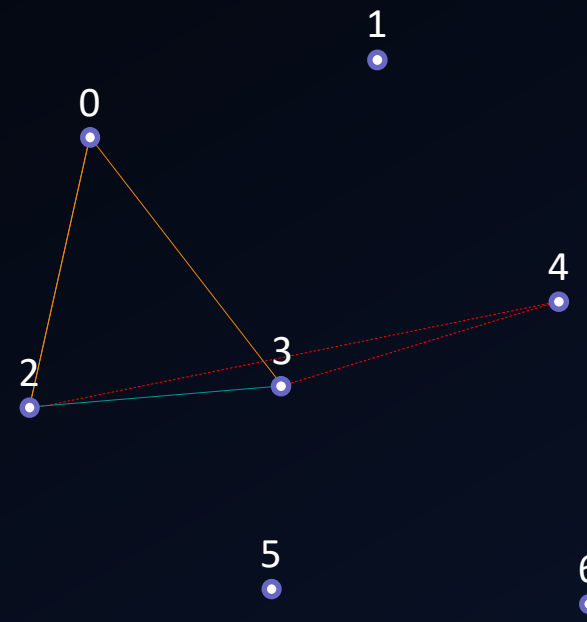
#	I	J	K
1	2	3	0
2			
3			
4			
5			
6			

R_T Check point 1 and didn't find edges 1-2 and 1-3 TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 3. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2			
3			
4			
5			
6			

RT

Check point 4 and didn't find edges 2-4 and 3-4

TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 4. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
…		
11	3 - 6	3.5
12	4 - 6	3.8

#	I	J	K
1	2	3	0
2	2	3	5
3			
4			
5			
6			

RT

Check point 5 and find edges 2-5 and 3-5
Two triangles are built so go to the next edge

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 5. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8

#	I	J	K
1	2	3	0
2	2	3	5
3			
4			
5			
6			

R_T

Check point 0 and didn't find edge 0-5

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 6. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8

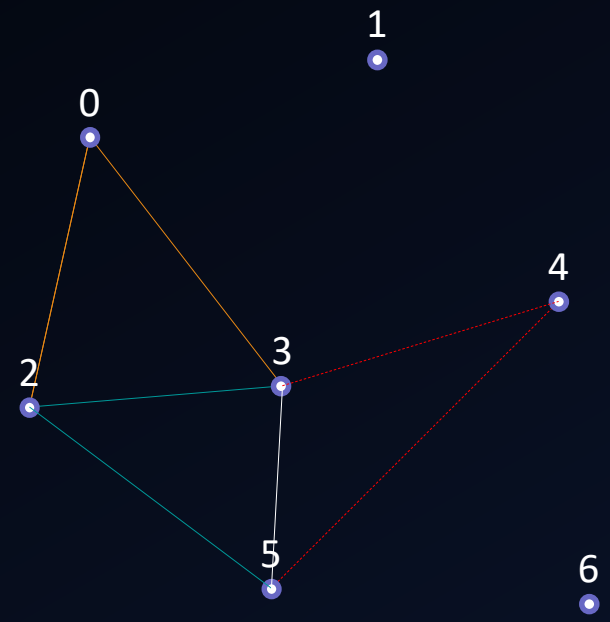
#	I	J	K
1	2	3	0
2	2	3	5
3			
4			
5			
6			

R_T Check point 1 and didn't find edges 1-3 and 1-5 TIN-model

Greedy triangulation

An example. STEP 3. LOOP 7. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3			
4			
5			
6			

R_T

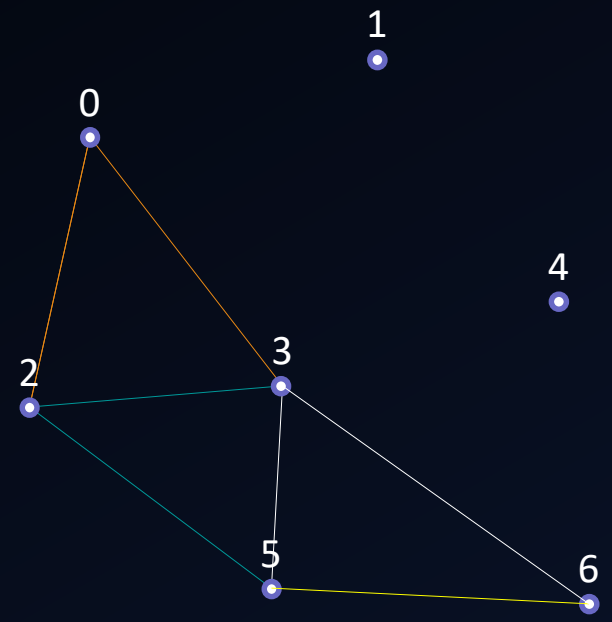
Check point 4 and didn't find edge 4-5

TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 8. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
…		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4			
5			
6			

R_T

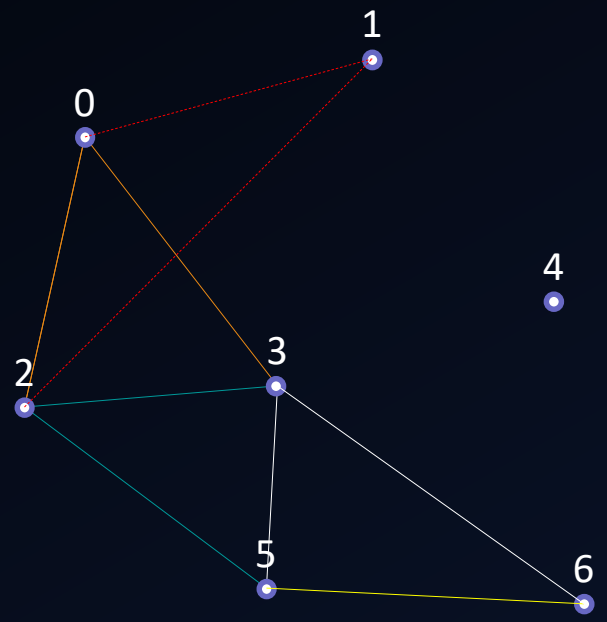
Check point 6 and find edges 3-6 and 5-6
Two triangles are built so go to the next edge

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 9. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4			
5			
6			

R_T

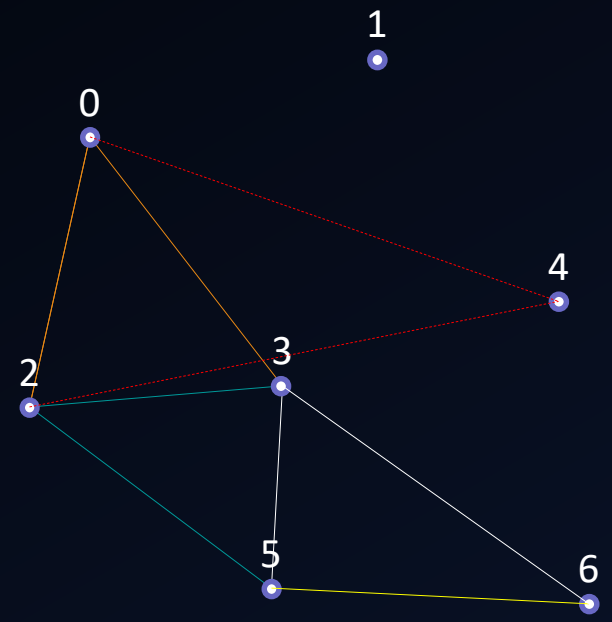
Check point 1 and didn't find edge 1-2

TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 10. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



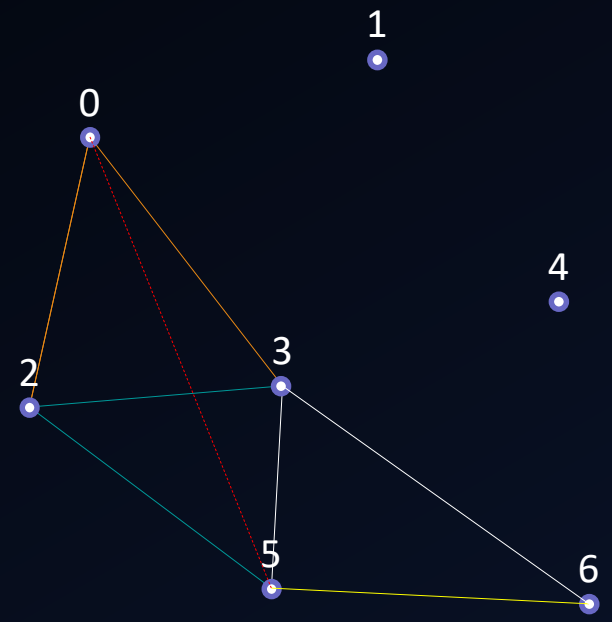
#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4			
5			
6			

R_T Check point 4 and didn't find edges 0-4 and 2-4 TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 11. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4			
5			
6			

R_T

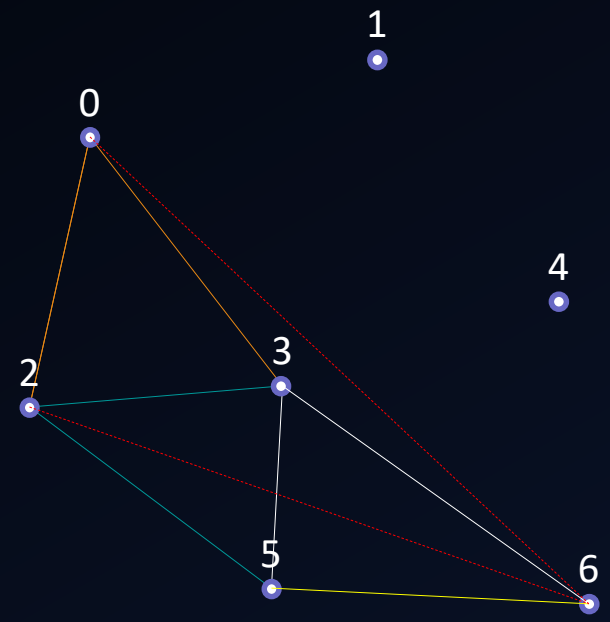
Check point 5 and didn't find edge 0-5

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 12. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4			
5			
6			

R_T

Check point 6 and didn't find edges 0-6 and 2-6

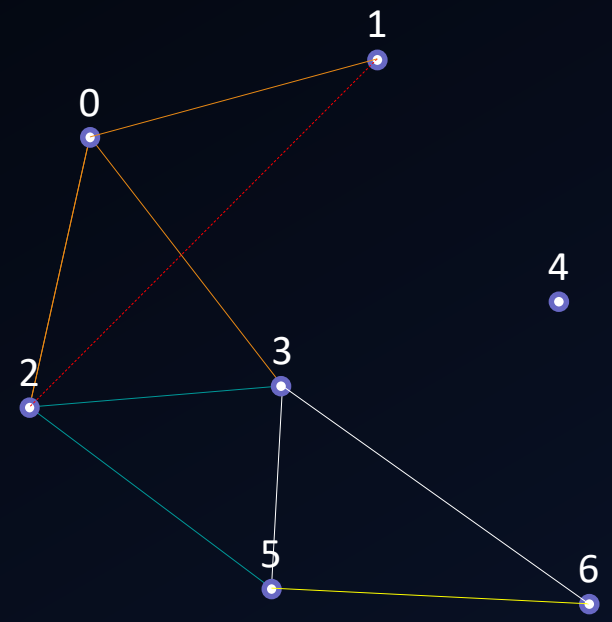
All points are checked so go to the next edge

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 13. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	
5			
6			

R_T

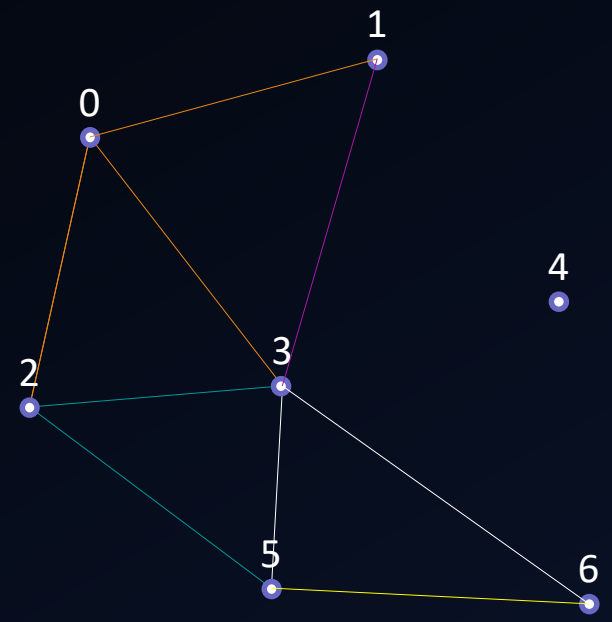
Check point 2 and didn't find edge 1-2

TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 14. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5			
6			

R_T

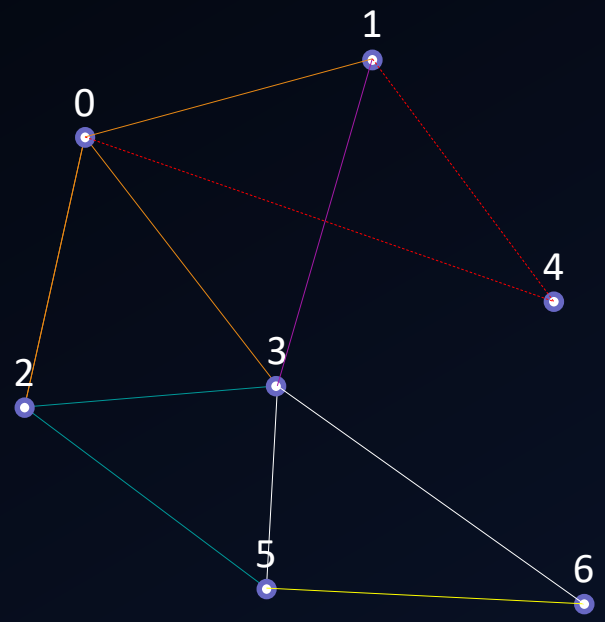
Check point 3 and find edges 0-3 and 1-3

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 15. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



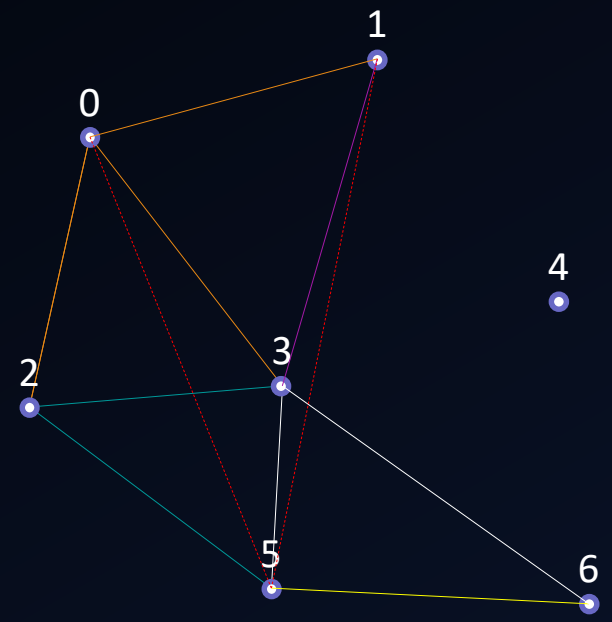
#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5			
6			

R_T Check point 4 and didn't find edges 0-4 and 1-4 TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 16. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



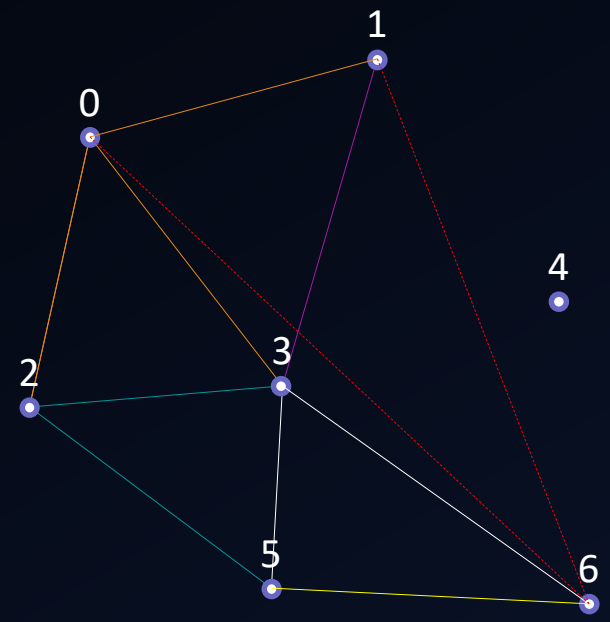
#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5			
6			

R_T Check point 5 and didn't find edges 0-5 and 1-5 TIN-model

Greedy triangulation

An example. STEP 3. LOOP 17. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5			
6			

R_T

Check point 6 and didn't find edges 0-6 and 1-6

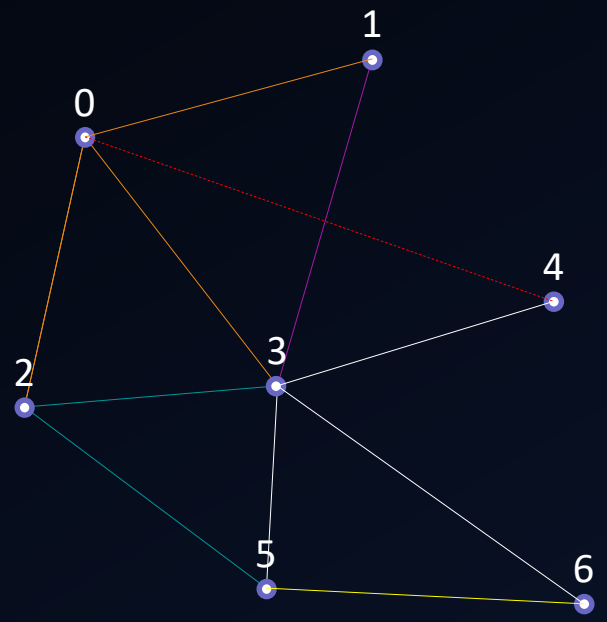
All points are checked so go to the next edge

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 18. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5	3	4	
6			

R_T

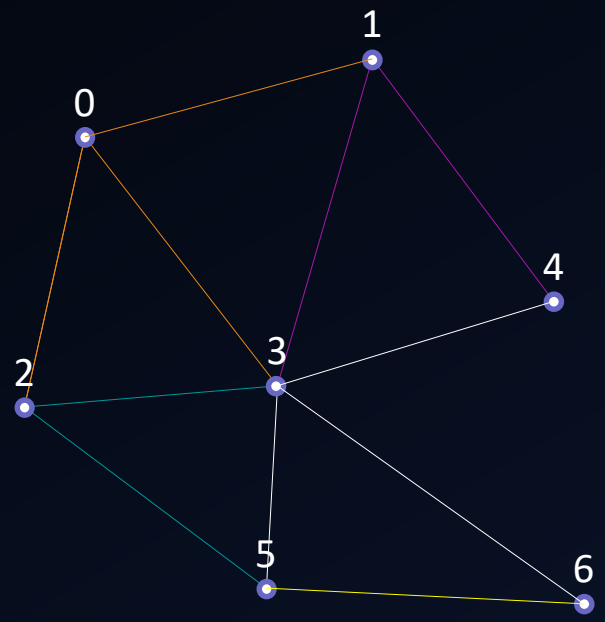
Check point 0 and didn't find edge 0-4

TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 19. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5	3	4	1
6			

R_T

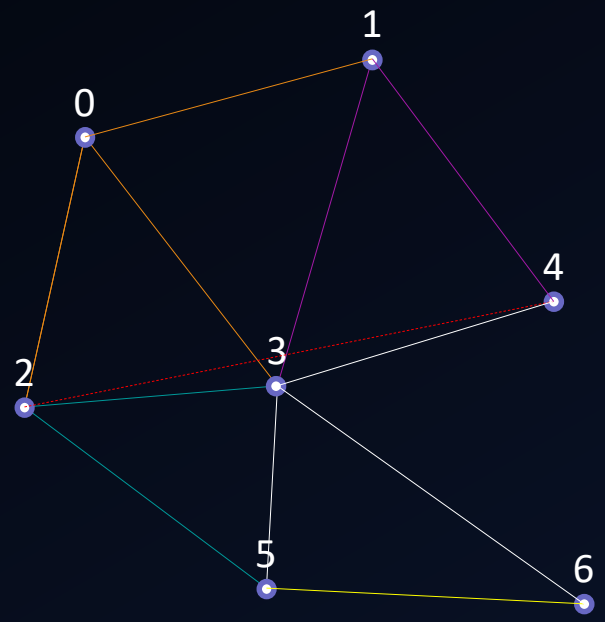
Check point 1 and find edges 1-3 and 1-4

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 20. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5	3	4	1
6			

R_T

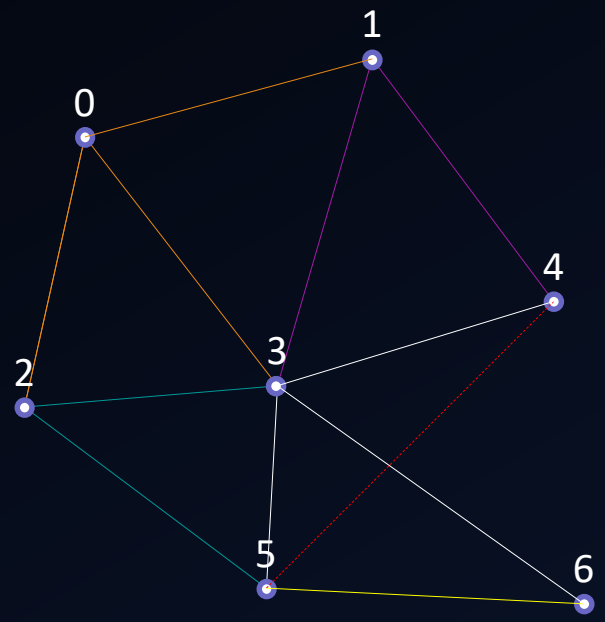
Check point 2 and didn't find edge 2-4

TIN-model

Greedy triangulation

An example. STEP 3. LOOP 21. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5	3	4	1
6			

R_T

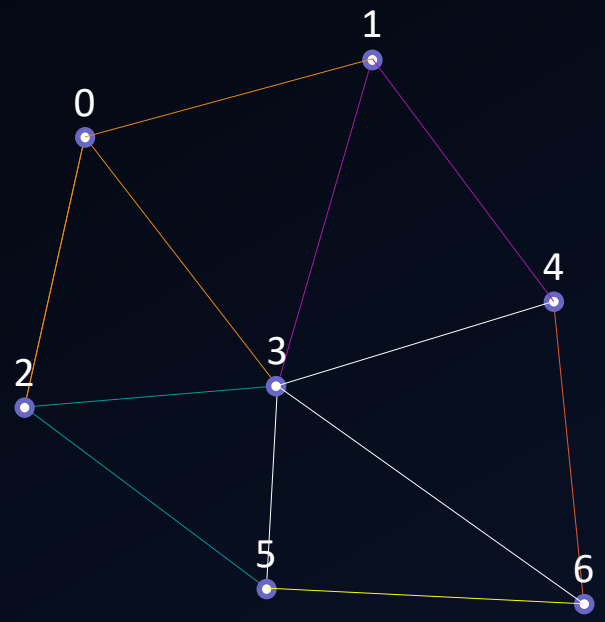
Check point 5 and didn't find edge 4-5

TIN-model

GREEDY TRIANGULATION

An example. STEP 3. LOOP 22. Creating triangles based on points and edges.

#	EDGE	LENGHT
1	2 - 3	2.1
2	3 - 5	2.2
3	0 - 2	2.4
4	0 - 1	2.7
5	3 - 4	2.8
6	5 - 6	3.0
...		
11	3 - 6	3.5
12	4 - 6	3.8



#	I	J	K
1	2	3	0
2	2	3	5
3	3	5	6
4	0	1	3
5	3	4	1
6	3	4	6

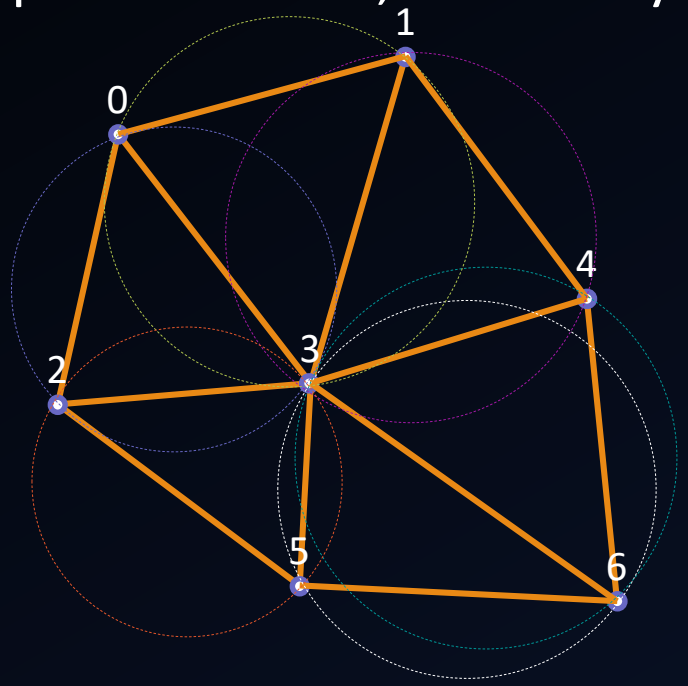
R_T

Check point 6 and find edges 4-6 and 3-6

TIN-model

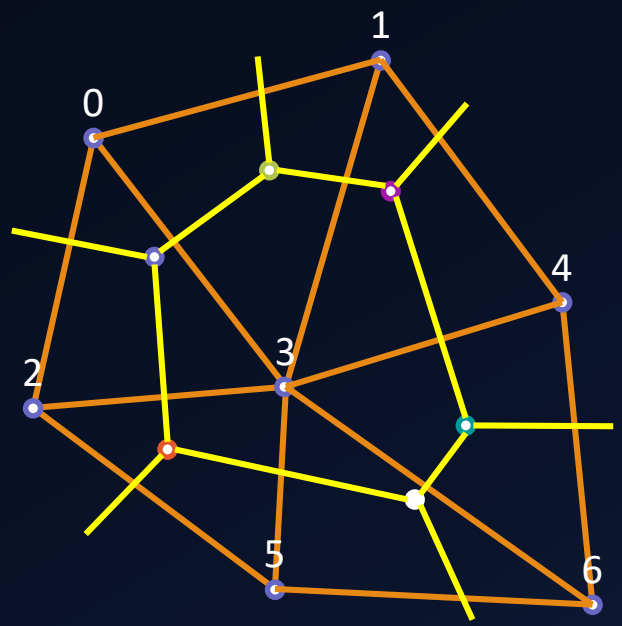
Delaunay triangulation

The Delaunay triangulation is a triangulation which is equivalent to the nerve of the cells in a Voronoi diagram, i.e., that triangulation of the convex hull of the points in the diagram in which every circumcircle of a triangle is an empty circle. Delaunay's triangulation is well balanced — its triangles are directed towards equilateral ones, and the system of triangles always has a convex hull.



Delaunay triangulation

Voronoi diagram



Voronoi + Delaunay

Delaunay triangulation

Many algorithms for computing Delaunay triangulations rely on fast operations for detecting when a point is within a triangle's circumcircle and an efficient data structure for storing triangles and edges. In two dimensions, one way to detect if point **D** lies in the circumcircle of **A**, **B**, **C** is to evaluate the determinant:

$$\Delta = \begin{bmatrix} x_a & y_a & x_a^2 + y_a^2 & 1 \\ x_b & y_b & x_b^2 + y_b^2 & 1 \\ x_c & y_c & x_c^2 + y_c^2 & 1 \\ x_d & y_d & x_d^2 + y_d^2 & 1 \end{bmatrix} = \begin{bmatrix} x_a - x_d & y_a - y_d(x_a - x_d^2) & y_a - y_d^2 \\ x_b - x_d & y_b - y_d(x_b - x_d^2) & y_b - y_d^2 \\ x_c - x_d & y_c - y_d(x_c - x_d^2) & y_c - y_d^2 \end{bmatrix}$$

When **A**, **B**, **C** are sorted in a counterclockwise order, this determinant is positive only if **D** lies inside the circumcircle. The common algorithms are the following:

- Flip algorithms
- Incremental
- Divide and conquer
- Sweephull

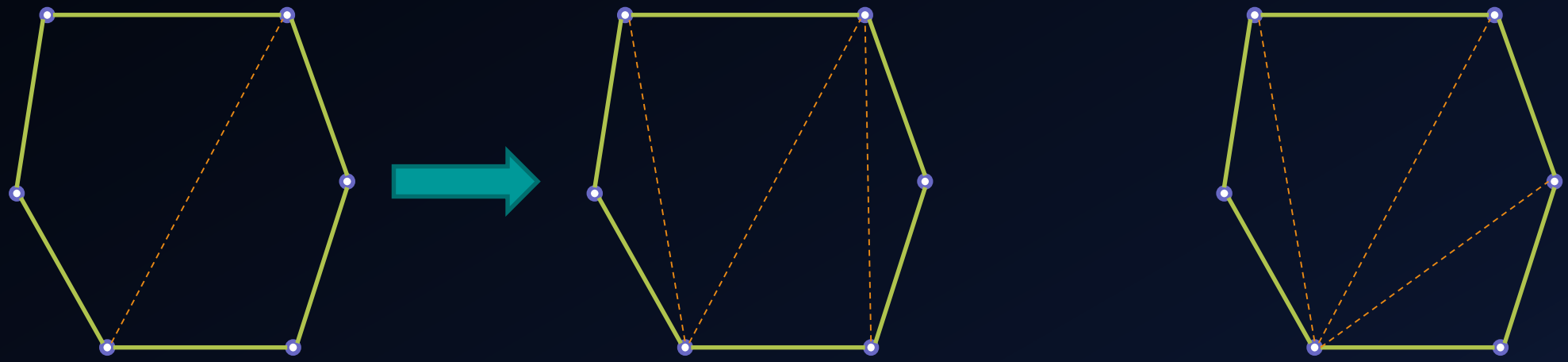
Triangulation of polygons

The task of polygon triangulation can be divided into three categories:

1. Triangulation of convex polygons.
2. Triangulation of non-convex polygons.
3. Triangulation of polygons with holes.

There are two methods for convex polygons triangulation:

- chord splitting into two polygons, and subsequent recursive splitting of each formed polygon until only triangles remain
- consecutive "cutting" of triangles by chords from one vertex



Triangulation of polygons

There are two methods for non-convex polygons triangulation:

- break it into convex polygons and triangulate with one of the algorithms described earlier
- apply the triangulation algorithm to a given non-convex polygon without dividing it into a convex one

The simplest method of dividing a non-convex polygon into a convex one is clipping. Its algorithm is as follows.

1. Move the coordinate system to the first vertex of the polygon.
2. Rotate the coordinate system so that the X axis coincides with the edge of the polygon.
3. Find the sides of the polygon that lie below the X-axis and cut them off.
4. Go to the next vertex and repeat the algorithm recursively until only convex polygons remain.

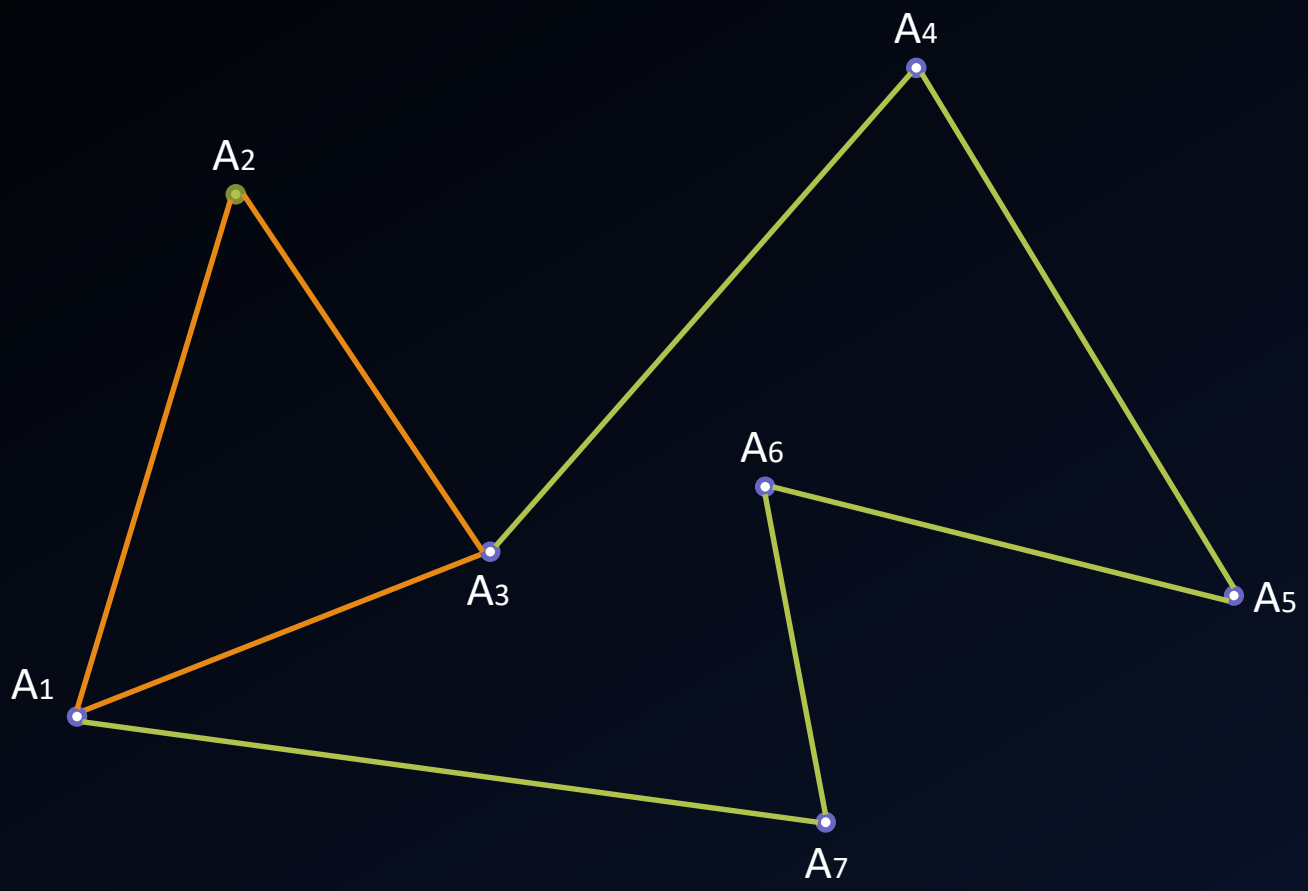
Triangulation of polygons

Another method of triangulation of a non-convex polygon is the algorithm of sequential construction of triangles on three adjacent vertices with verification of the obtained result.
First, we arrange all vertices of the polygon clockwise. Then the following algorithm can be used to construct the triangulation.

1. Take three consecutive vertices A_i, A_{i+1}, A_{i+2}.
2. Check if the vectors A_iA_{i+1} and $A_{i+1}A_{i+2}$ form a right turn. The cross-product of the vectors A_iA_{i+1} and A_iA_{i+2} must be negative.
3. Check if any of the remaining vertices are located in the triangle $A_iA_{i+1}A_{i+2}$.
4. If conditions 2 and 3 are met, we construct a triangle with the points $A_iA_{i+1}A_{i+2}$. We exclude the vertex A_{i+1} from consideration. Next, we consider the triangle $A_iA_{i+2}A_{i+3}$.
5. If at least one of the conditions 2 or 3 is not met, then we proceed to consider vertices A_{i+1}, A_{i+2}, and A_{i+3}.
6. Repeat step 1 until only 3 vertices remain, forming the last triangle.

TRIANGULATION OF POLYGONS

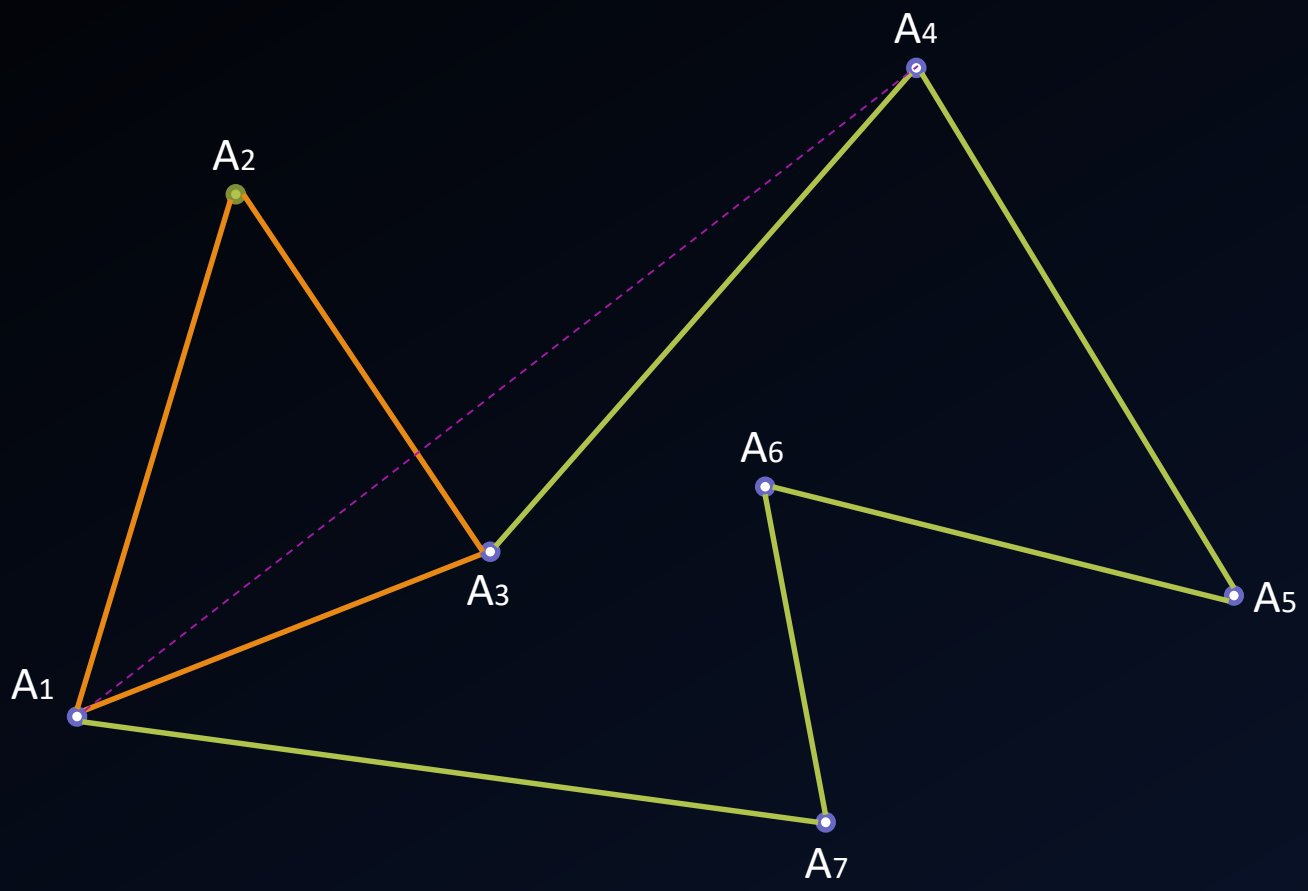
An example. STEP 1.



1. Take the first three points $A_1A_2A_3$.
2. Check the cross-product sign of the vectors A_1A_2 and A_1A_3.
3. Check if any of the remaining vertices are located in the triangle $A_1A_2A_3$.
4. Both conditions are met, so we construct triangle $A_1A_2A_3$. We exclude the vertex A_2 from consideration.

TRIANGULATION OF POLYGONS

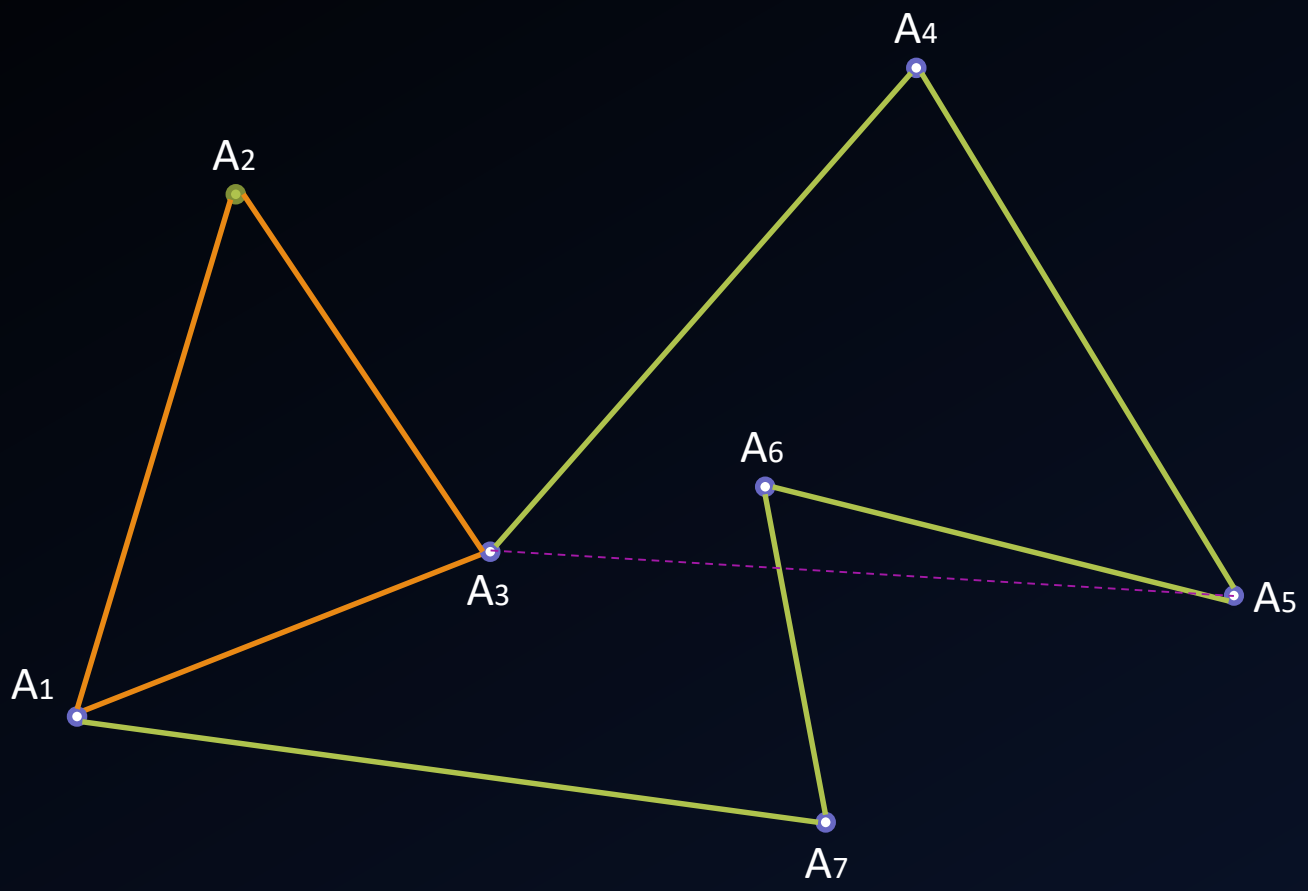
An example. STEP 2.



1. Take the next three points $A_1A_3A_4$.
2. Check the cross-product sign of the vectors A_1A_3 and A_1A_4.
3. Check if any of the remaining vertices are located in the triangle $A_1A_3A_4$.
4. The first condition is not met, so we move to the next point.

Triangulation of polygons

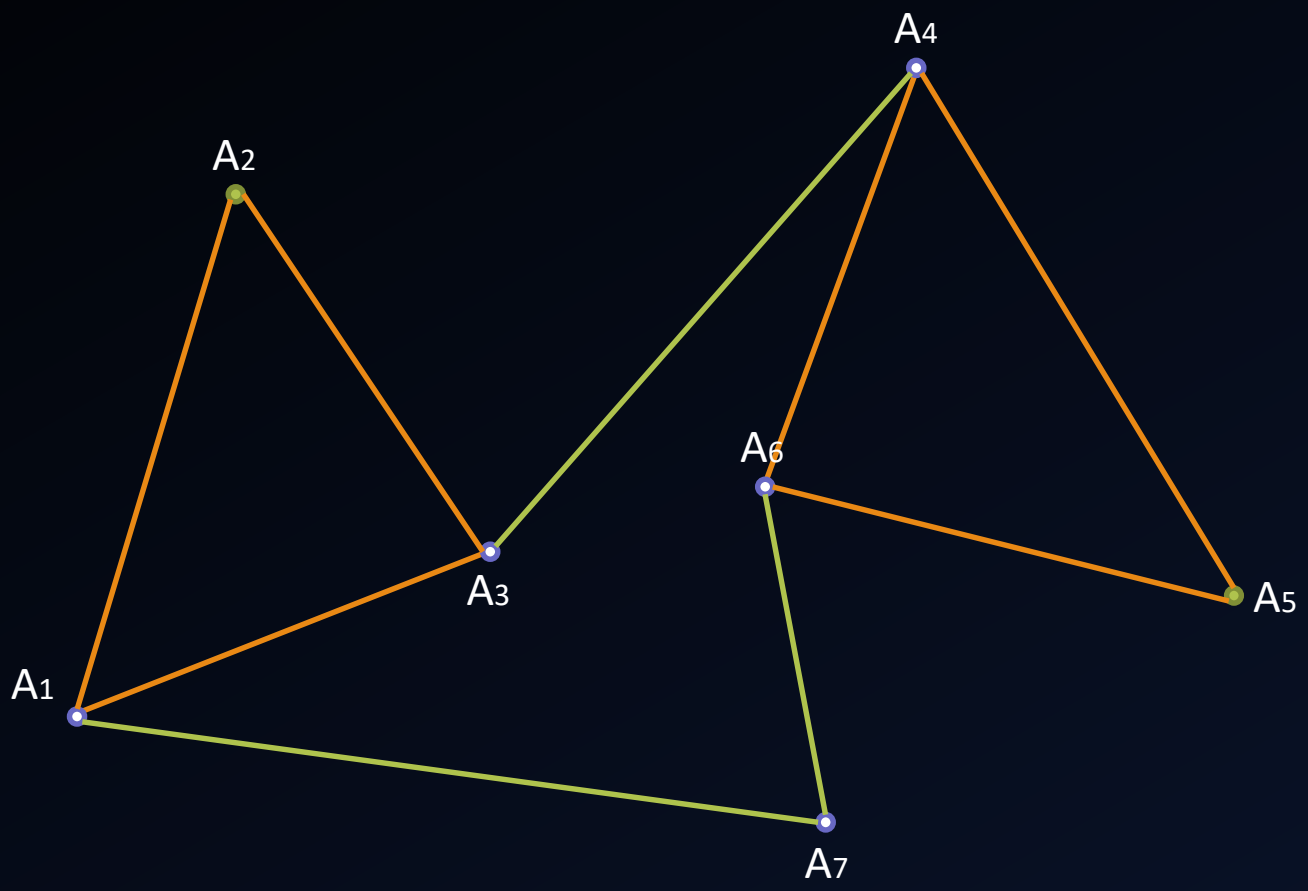
An example. STEP 3.



1. Take the next three points $A_3A_4A_5$.
2. Check the cross-product sign of the vectors A_3A_4 and A_3A_5.
3. Check if any of the remaining vertices are located in the triangle $A_3A_4A_5$.
4. The second condition is not met, so we move to the next point.

Triangulation of polygons

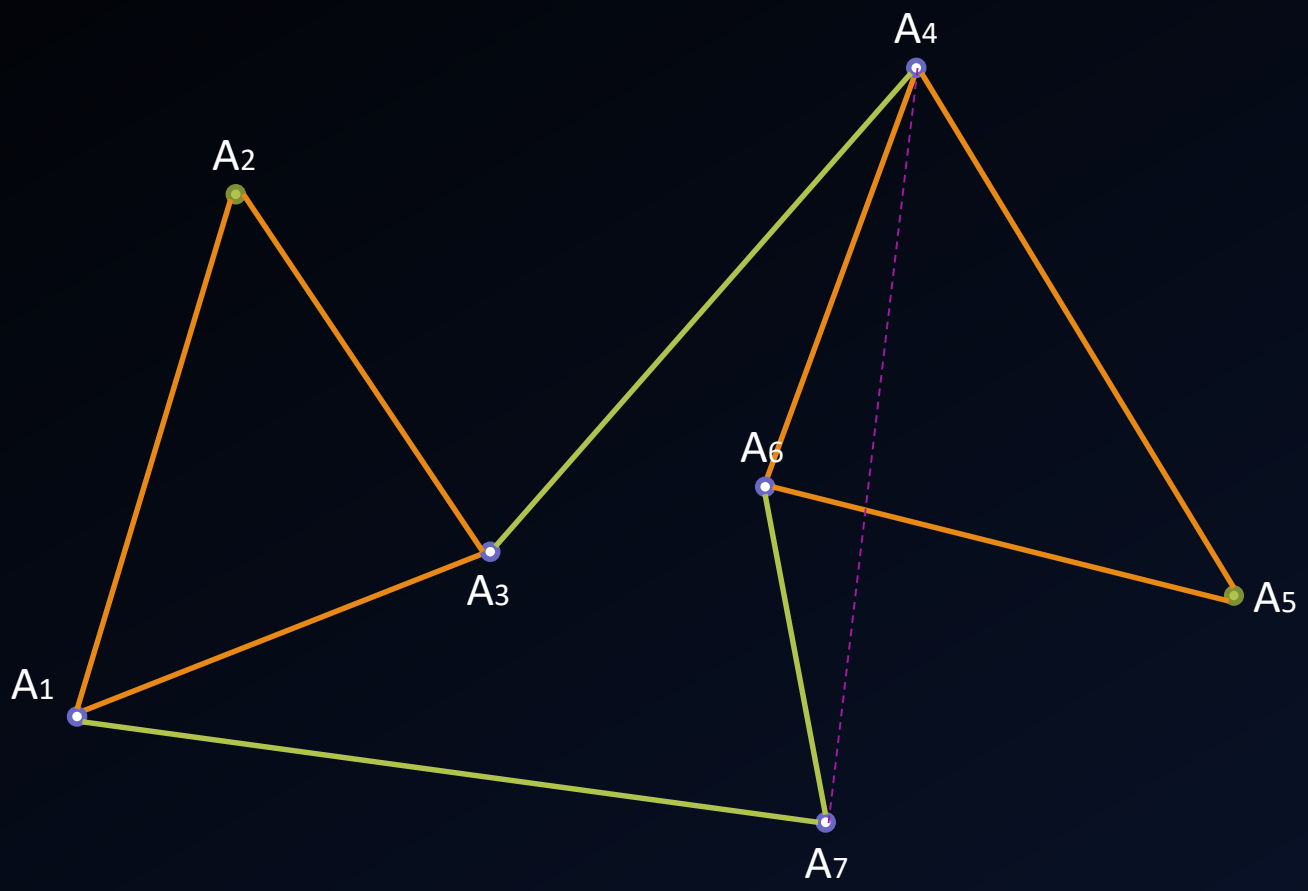
An example. STEP 4.



1. Take the next three points $A_4A_5A_6$.
2. Check the cross-product sign of the vectors A_4A_5 and A_4A_6.
3. Check if any of the remaining vertices are located in the triangle $A_4A_5A_6$.
4. Both conditions are met, so we construct triangle $A_4A_5A_6$. We exclude the vertex A_5 from consideration.

Triangulation of polygons

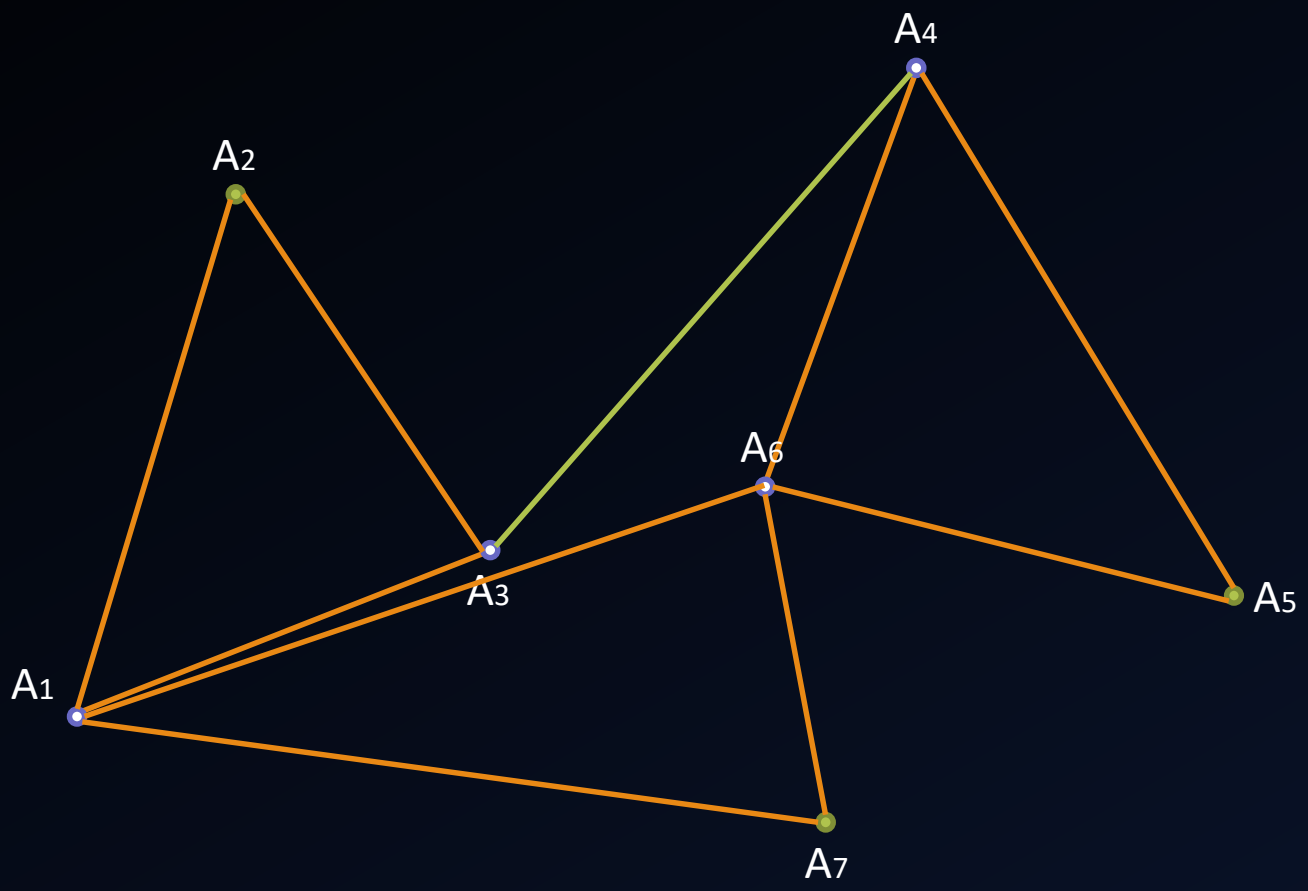
An example. STEP 5.



1. Take the next three points $A_4A_6A_7$.
2. Check the cross-product sign of the vectors A_4A_6 and A_4A_7.
3. Check if any of the remaining vertices are located in the triangle $A_4A_6A_7$.
4. The first condition is not met, so we move to the next point.

Triangulation of polygons

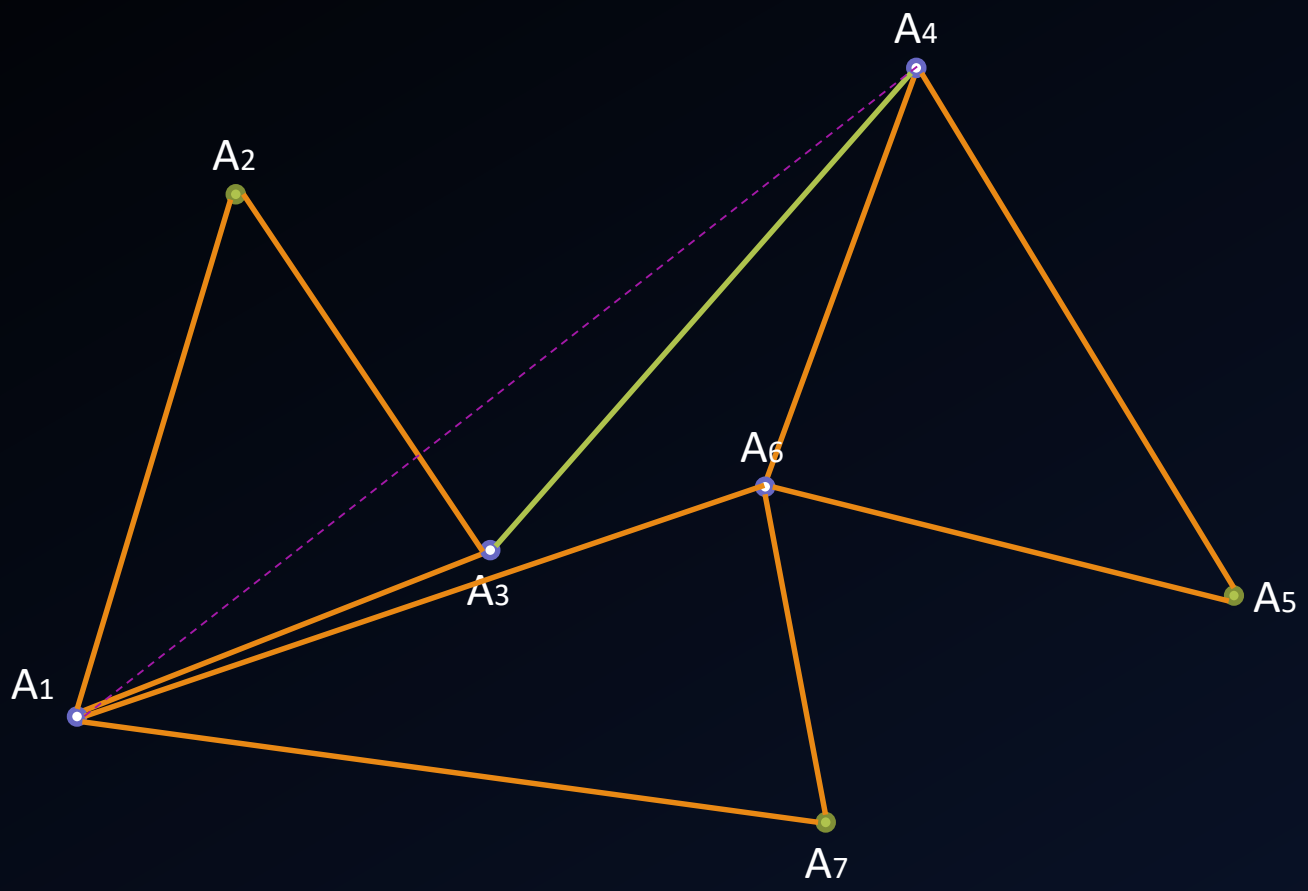
An example. STEP 6.



1. Take the next three points $A_6A_7A_1$.
2. Check the cross-product sign of the vectors A_6A_7 and A_6A_1.
3. Check if any of the remaining vertices are located in the triangle $A_6A_7A_1$.
4. Both conditions are met, so we construct triangle $A_6A_7A_1$. We exclude the vertex A_7 from consideration.

Triangulation of polygons

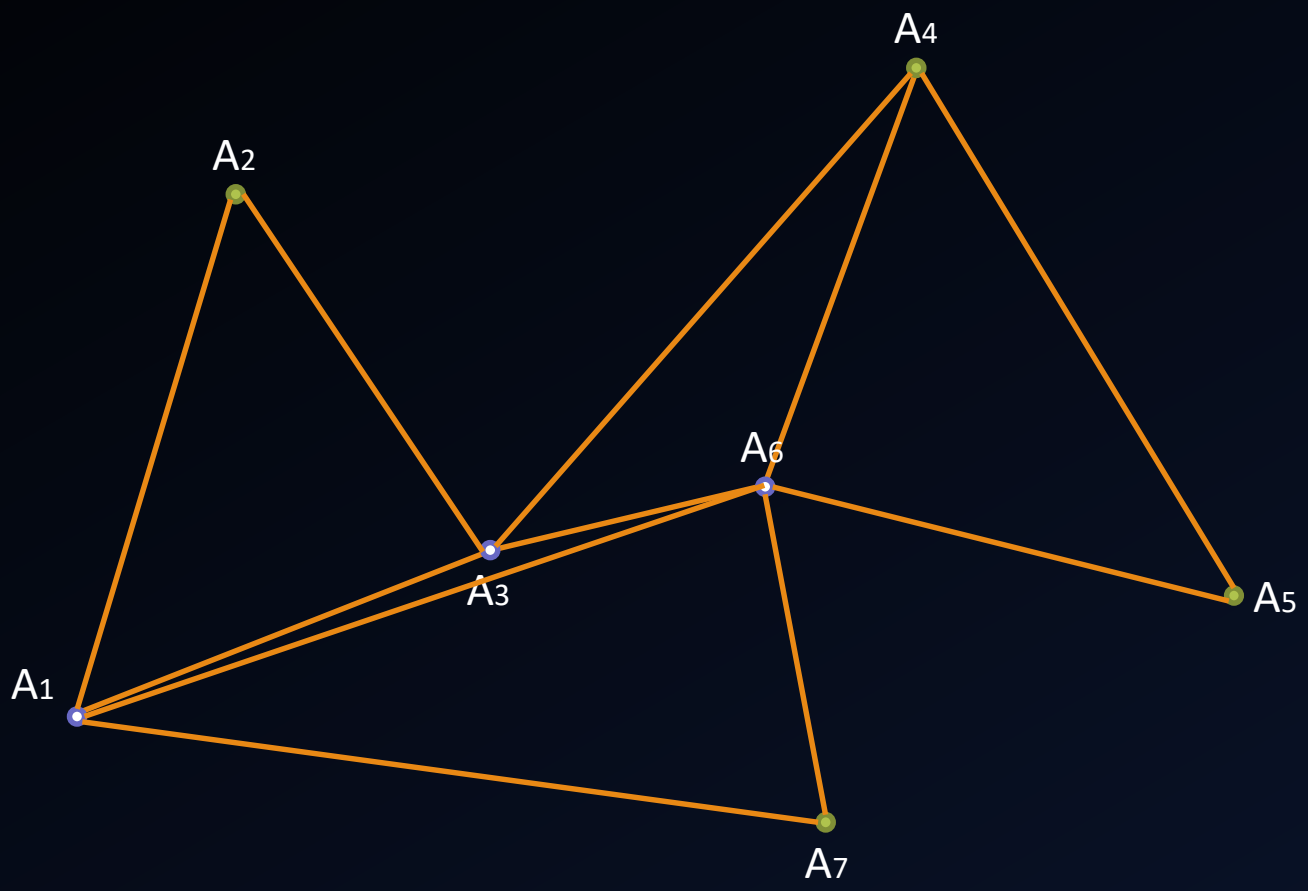
An example. STEP 7.



1. Take the next three points $A_1A_3A_4$.
2. Check the cross-product sign of the vectors A_1A_3 and A_1A_4.
3. Check if any of the remaining vertices are located in the triangle $A_1A_3A_4$.
4. The first condition is not met, so we move to the next point.

TRIANGULATION OF POLYGONS

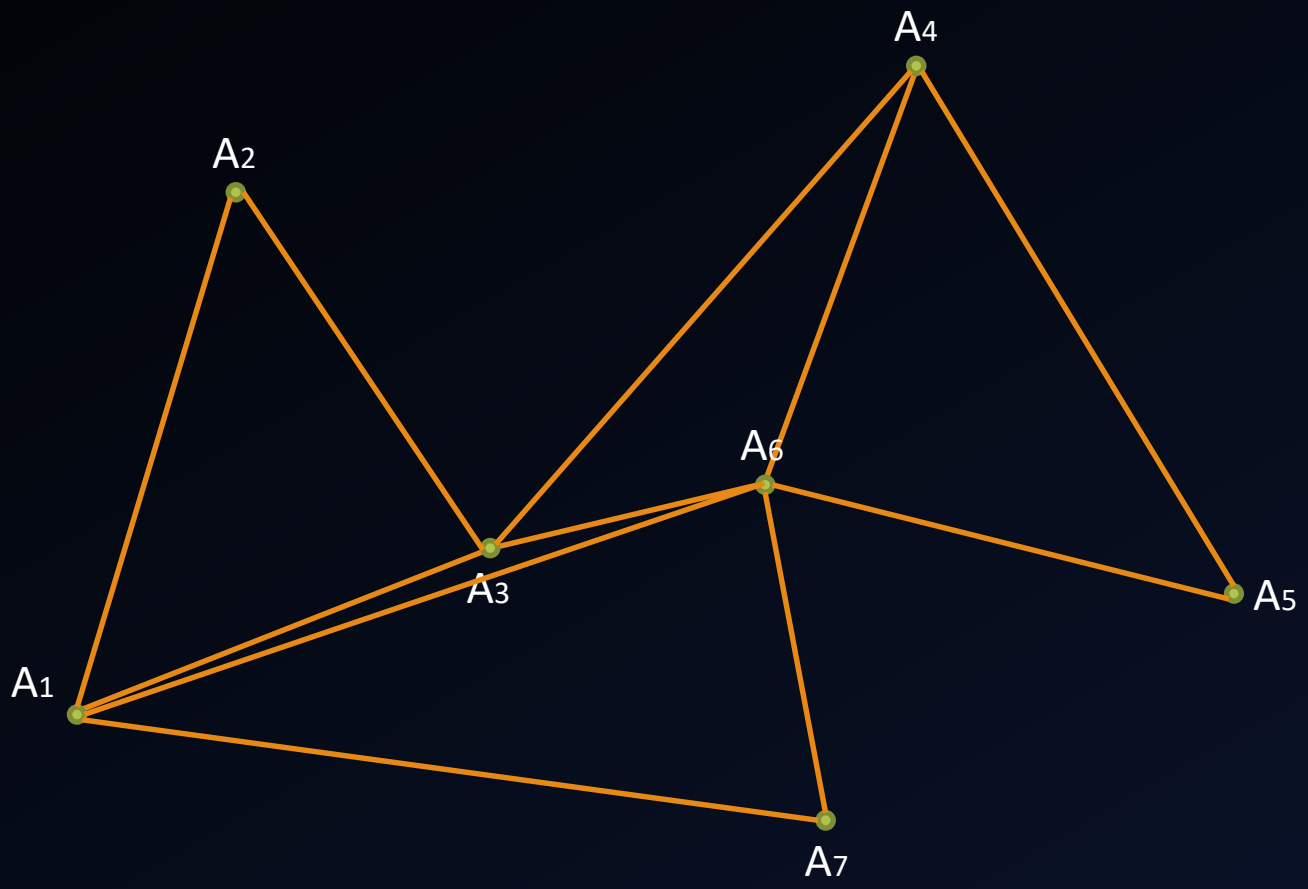
An example. STEP 8.



1. Take the next three points $A_3A_4A_6$.
2. Check the cross-product sign of the vectors A_3A_4 and A_3A_6.
3. Check if any of the remaining vertices are located in the triangle $A_3A_4A_6$.
4. Both conditions are met, so we construct triangle $A_3A_4A_6$. We exclude the vertex A_4 from consideration.

TRIANGULATION OF POLYGONS

An example. STEP 9.



Only three vertexes are left, so we construct the last triangle $A_3A_6A_1$.

Thank you!