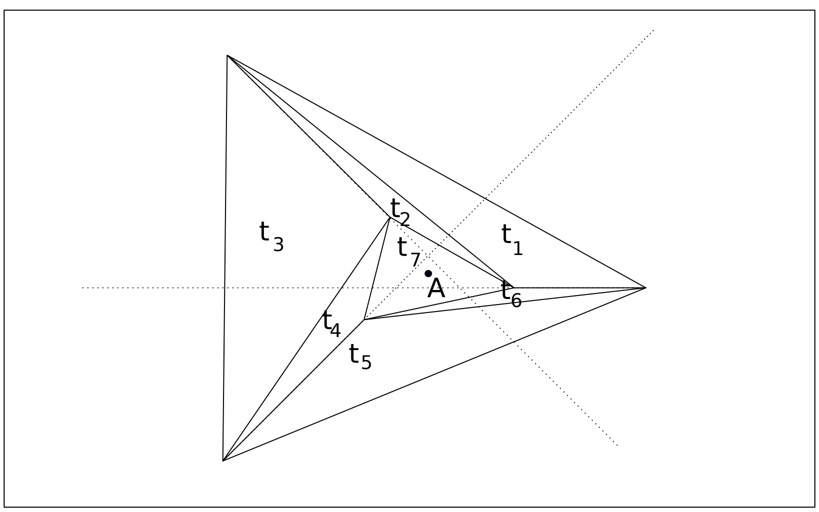
Formal proofs for the convergence of visibility walks in triangulations

Yves Bertot
Yves.Bertot@inria.fr

Given a triangulation of of a convex plane region, there exists a single algorithm to find the triangle that contains a given point:

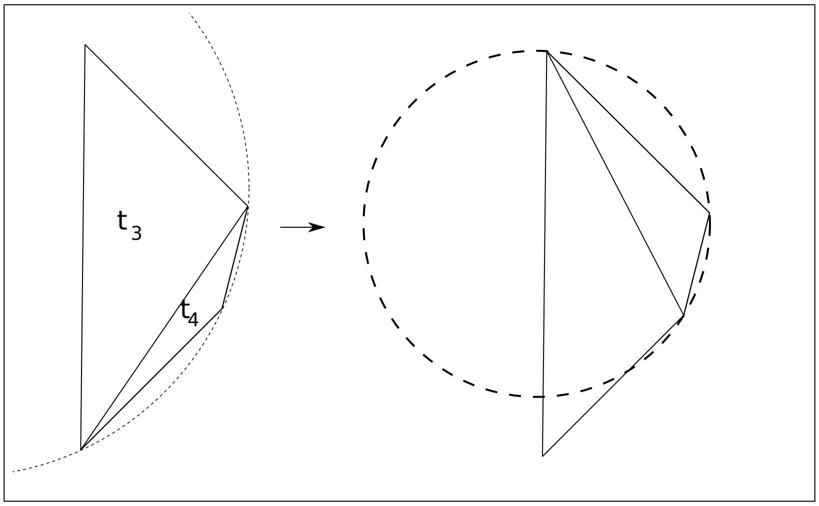
- start with an arbitrary triangle,
- find one of the edges of the triangle that sepates the triangle from the target point (if no such edge exist, the current triangle contains the target point),
- move to the neighbor triangle at that edge and start again.

This algorithm is known as the *visibility walk*. It fails in certain configurations of the triangulation and the target point. The following illustration shows an example where the walk going following triangles $t_1, t_2, t_3, t_4, t_5, t_6, t_1$ loops without reaching the triangle t_7 that contains the target point A. From triangle t_1, the edge between t_1 and t_2 is the only one that has point A on the other side of the edge, so one must move to t_2. From t_2, both the edge between t_2 and t_3 and the edge betwee t_2 and t_7 have point A on the other side. Therefore a move from t_2 to t_3 is legitimate. The same pattern reproduces to move from t_3 to t_4 and from t_4 to t_5. The dashed lines show t_3 is a legitimate move from t_2, t_5 is a legitimate move from t_4, and t_1 is a legitimate move from t_6.



On the other hand, the visibility walk algorithm is guaranteed to succeed when the triangulation satisfies the *Delaunay criterion*. The Delaunay criterion is satisfied when no vertex of a triangle is inside the circumcircle of an adjacent triangle. In the counter example figure, the point of t_4 that is not common with t_3 is inside the circumcircle of triangle t_3.

When the Delaunay criterion is violated, the triangulation can be improved by flipping the edge between the two offending triangles. This is illustrated in the following figure, where triangles t_3 and t_4 are replaced by a new pair of triangles covering the same surface, but with a different common edge. On each side of the transformation, the figure shows the circumcircle of one of the triangles (as a dashed line). On the left hand side, the third point of triangle t_3 is inside the circumcircle of triangle t_4, on the right hand side, the third point of one of the triangles is outside the circumcircle of the other triangle.



This suggests two approaches:

1. Given an arbitrary triangulation, first make it satisfy the Delaunay criterion and then solve the triangle walking problem.
2. Modify only the triangles encountered during the visibility walk.

The goal of this internship is to study these approaches as part of a formal proof performed using the Coq system. The work will be done at Inria Sophia Antipolis, under the supervision of Yves Bertot. Previous work on the topic contains a description of algorithms to produce arbitrary (non-Delaunay) triangulations and a proof that the process of repeatedly flipping offending edges eventually produces a Delaunay triangulation.

Context: Proofs will be performed using the Coq system[1] and the Mathematical Components library[2]. Some experiments may require writing example implementations in Ocaml. The supervisor and his team will provide access to computers with Coq and the relevant libraries installed and training.

Prerequisites: The pre-requisite for this internship is a good knowledge of functional programming.

Tasks: The intern will have to write various implementations of the algorithm, either using plain inductive data structures, or using data-types for finite sets and graphs provided in the Mathematical Components library. They will then have to perform proofs, mostly using the Coq system and the existing theorems of the Mathematical Components library.

[1] `https://coq.inria.fr`
[2] `http://math-comp.github.io/math-comp/`