Rovinné triangulace a jejich využití.

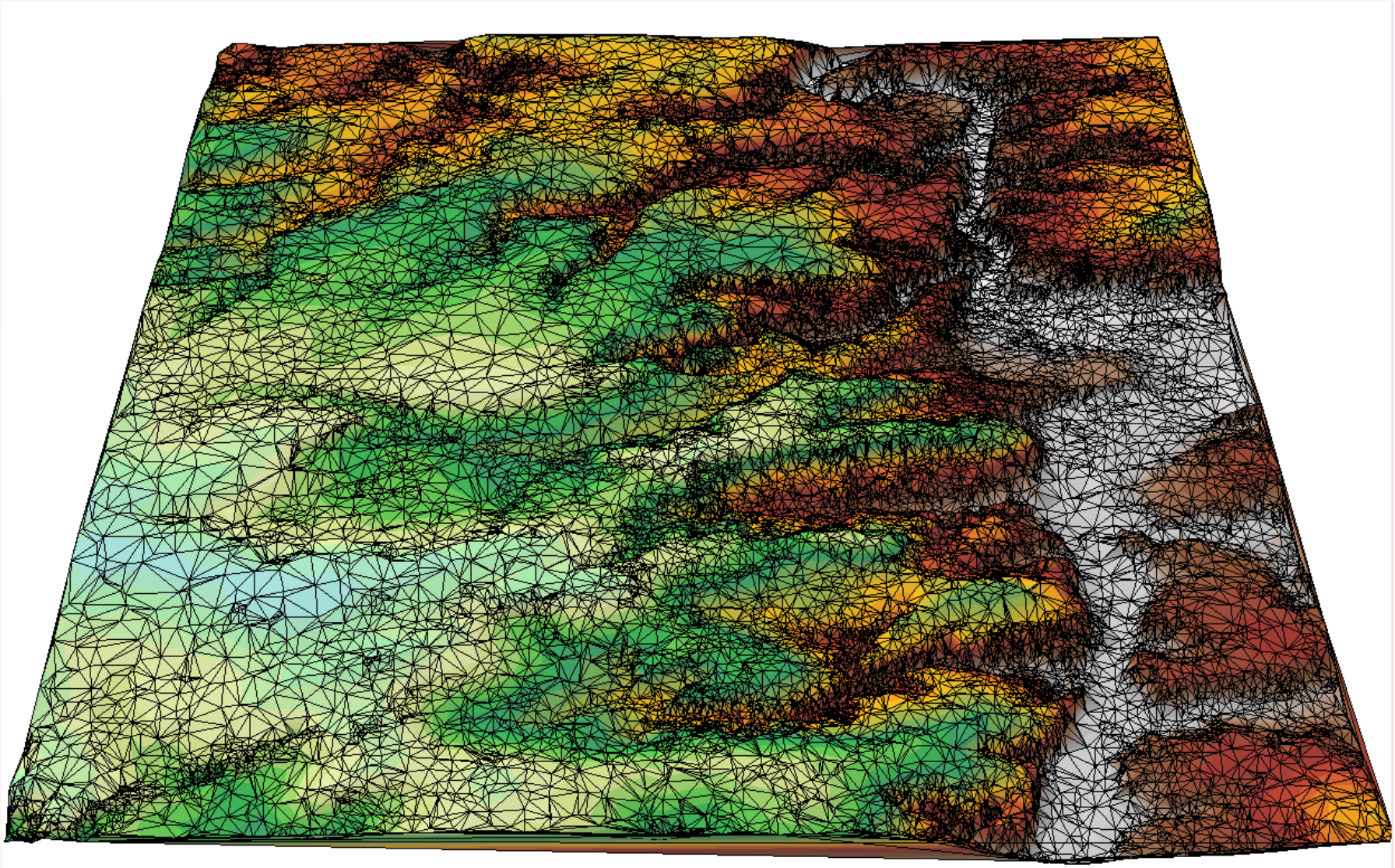
Greedy Triangulation. Delaunay Triangulation. Constrained Delaunay Triangulation. Data Dependent Triangulation. DMT.

Tomáš Bayer | bayertom@natur.cuni.cz

Katedra aplikované geoinformatiky a kartografie. Přírodovědecká fakulta UK.

Obsah přednášky

1. Tvorba digitálního modelu terénu



2. Formulace problému

Dáno:	Množina bodů $P = \{p_1, p_2, ..., p_n\}$ v $\mathbb{R}^2$.
Hledáme:	Triangulaci T nad množinou P.

Definice:
Triangulace $\mathcal{T}$ nad množinou bodů P představuje takové planární rozdělení, které vytvoří soubor m trojúhelníků $t = \{t_1, t_2, ..., t_m\}$ a hran tak, aby platilo:

- *Libovolné dva trojúhelníky $t_i, t_j \in \mathcal{T}, (i \neq j)$, mají společnou nejvýše hranu.*
- *Sjednocení všech trojúhelníků $t \in \mathcal{T}$ tvoří $\mathcal{H}(P)$.*
- *Uvnitř žádného trojúhelníku neleží žádný další bod z P.*

Vztah mezi počtem bodů n, hran n_h a trojúhelníků n_t v rovině pro $\mathcal{T}$

$$\begin{aligned} n_h &= 3n - 3 - k, \\ n_t &= 2n - 2 - k, \end{aligned}$$

k počet bodů ležících na $\mathcal{H}(P)$:
Odhad pouze funkcí n

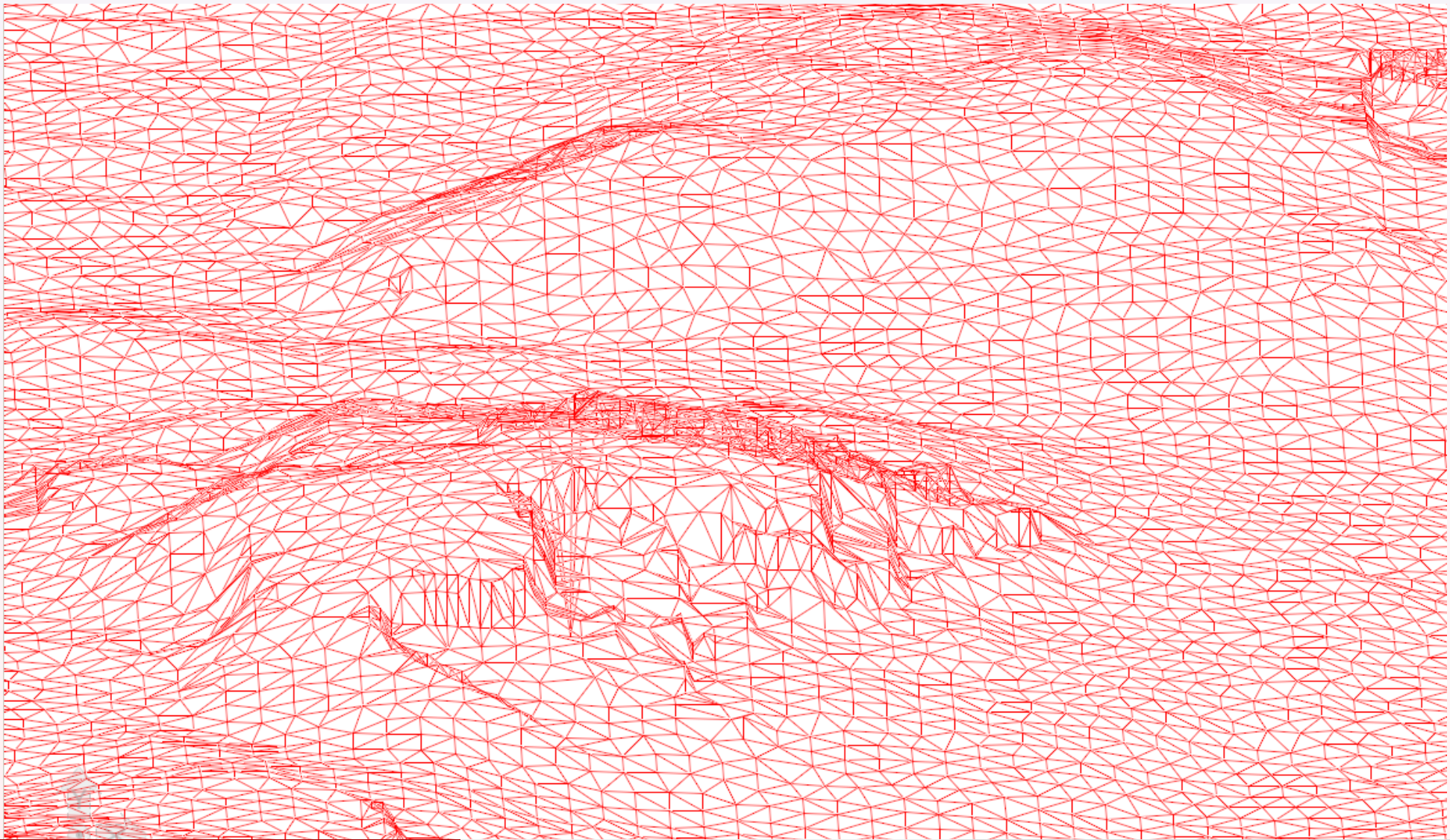
$$\begin{aligned} n_h &\leq 3n - 6, \\ n_t &\leq 2n - 5. \end{aligned}$$

3. Použití triangulací

Nejčastější aplikace triangulací:

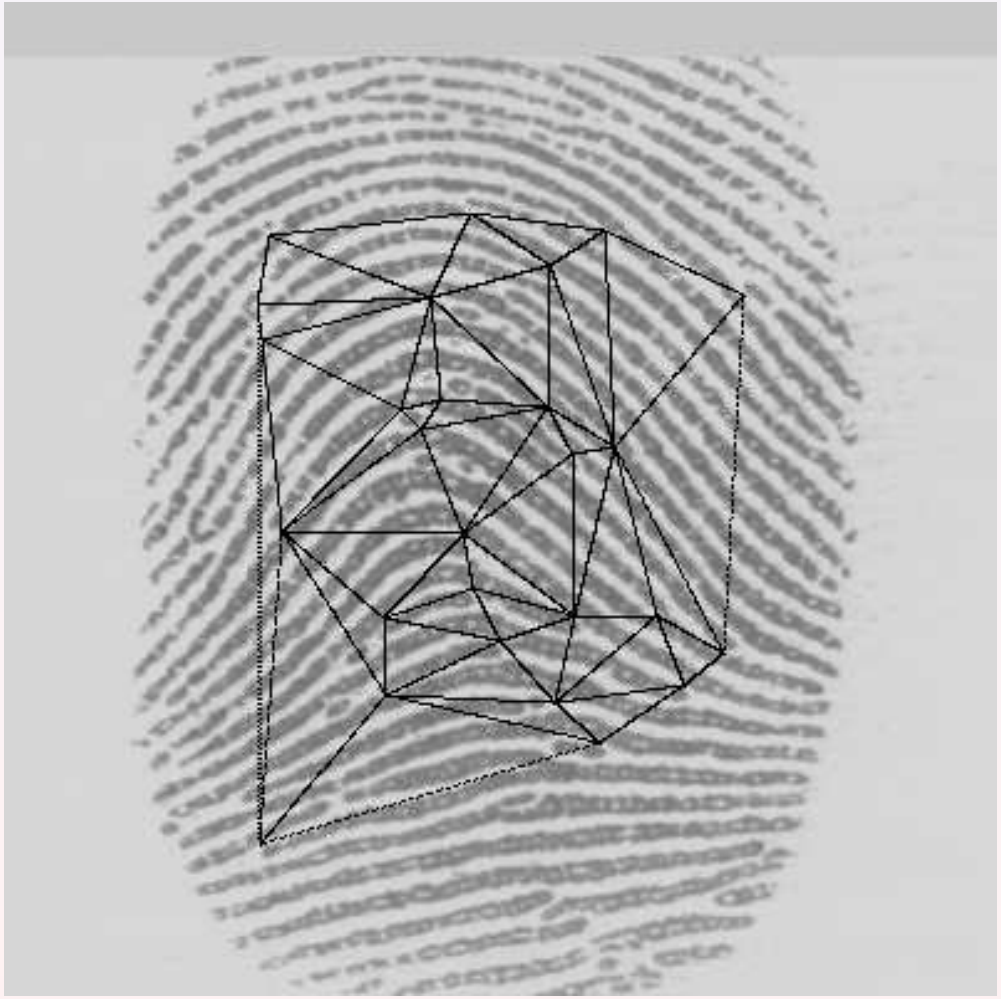
- Kartografie & GIS: tvorba digitálních modelů terénu (DMT).
- Zpracování obrazu: segmentace, rozpoznávání vzorů.
- DPZ: tvorba prostorových modelů z dat laserového skenování.
- Počítačová grafika: vizualizace prostorových dat ve scénách.
- FEM (tzv. metoda konečných prvků): analýza vlastností a struktury materiálů, simulace.
- Kartografická generalizace.
- Plánování pohybu robotů: vozítka na Marsu.
- Modelování přírodních jevů: eroze.
- Interpolační techniky: převod bodových jevů na plošné
- Biometrie: detekce otisků prstů.

4. Rekonstrukce terénu z dat leteckého laserového skenování



5. Aplikace triangulací v biometrii

Detekce otisků prstů (Bebis et al., 2000)



6. Požadavky na triangulaci $\mathcal{T}$

Požadavky na triangulační algoritmus:

- Jednoduchost algoritmu, snadná implementace.
- Dostatečná rychlost pro velká P $(n > 1E6)$ bodů, požadavek na $O(n \cdot \log(n))$ algoritmus.
- Malá citlivost na singulární případy, kdy $\mathcal{T}$ není jednoznačná (popř. ji nelze provést).
- Převod do vyšších dimenzí.
- Schopnost paralelizace algoritmu.
- Optimální tvar trojúhelníkové sítě.

Některé body v kontrastu: jednoduchost implementace x rychlost.

Triangulační algoritmy patří mezi jedny z nejvíce teoreticky rozpracovaných postupů.

7. Volba triangulace a jejich dělení

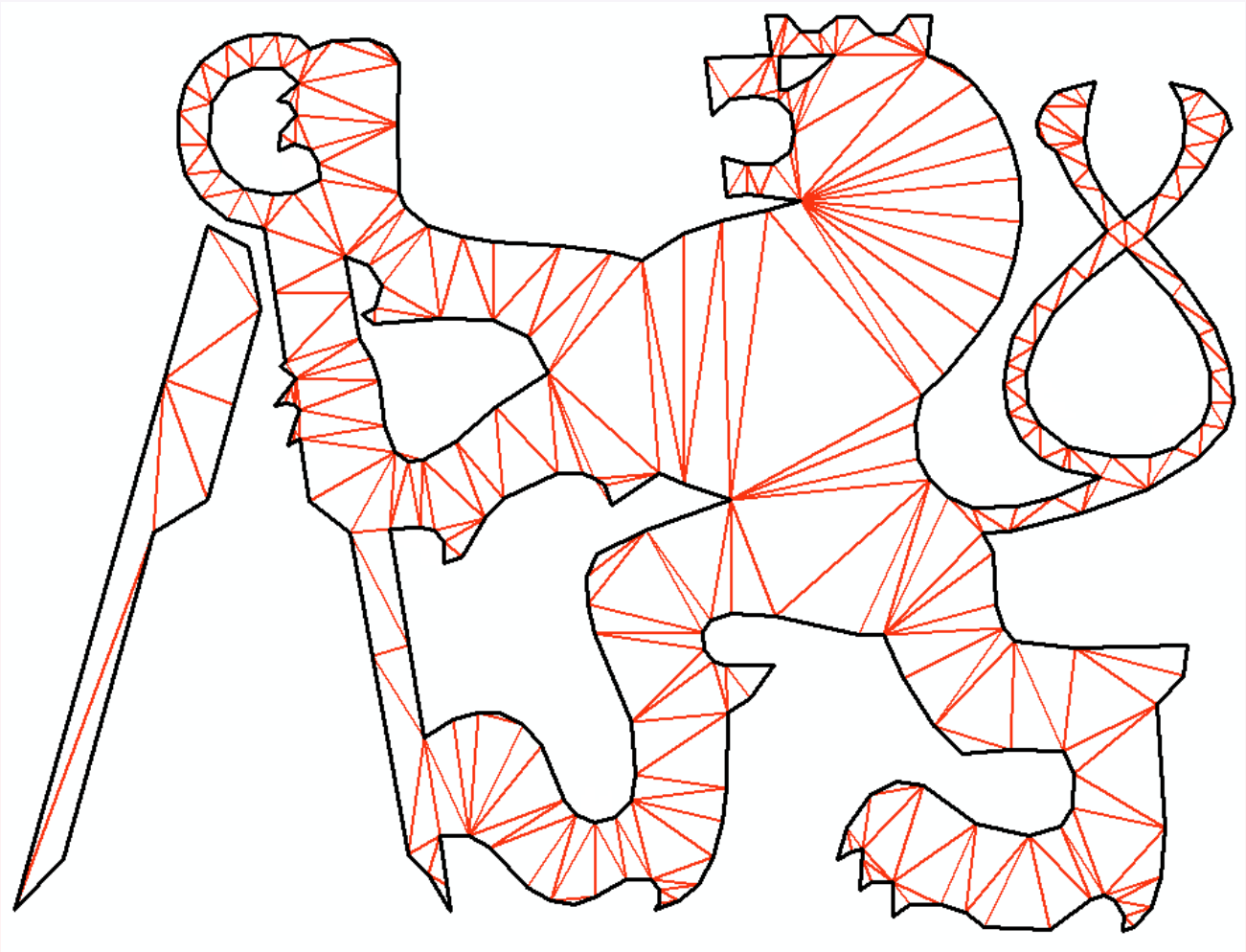
Při výběru triangulace $\mathcal{T}$ nutno zohlednit:

Tvar trojúhelníků:
Triangulace by měla produkovat pravidelné trojúhelníky vhodných tvarů (blížící se rovnostranným). Kritérium je důležité při tvorbě DMT, trojúhelníková síť se musí co nejvíce přimykat k terénu.

Povinné hrany:
Schopnost vkládat povinné hrany a modifikovat tvar triangulace. Ovlivnění tvaru terénu, vkládání kosterních čar, tj. hřbetnic, údolnic, spádnic.

Triangulace nekonvexní oblasti:
Schopnost triangulace nekonvexní oblasti či oblasti obsahující díry.
V mapách nejsou triangularizovány některé oblasti, např. vodní plochy, budovy.

8. Ukázka triangulace nekonvexní oblasti obsahující díry



9. Dělení triangulací

Dělení triangulací dle geometrické konstrukce:

- Greedy triangulace.
- Delaunay triangulace.
- MWT (Minimum Weight Triangulation).
- Constrained triangulace (triangulace s povinnými hranami).
- Datově závislé triangulace.

Dělení triangulací dle použitých kritérií:

- Lokálně optimální triangulace.
- Globálně optimální triangulace.
- Multikriteriálně optimalizované triangulace.

Vlastnosti triangulace $\mathcal{T}$ se posuzují ve vztahu k těmto kritériím.

10. Lokálně vs. globálně optimální triangulace

Lokálně optimální triangulace $\mathcal{T}$:
Každý čtyřúhelník tvořený dvojicí trojúhelníků se společnou stranou triangularizován optimálně vzhledem k zadanému kritériu.
Pro množinu P existuje více lokálně optimálních triangulací, každá z nich optimalizuje jiné kritérium.

Globálně optimální triangulace $\mathcal{T}$
Všechny trojúhelníky triangulace $\mathcal{T}$ optimální vzhledem k zadanému kritériu.
Neexistuje jiná triangulace $\mathcal{T}'$, která by dosáhla alespoň u jednoho trojúhelníku lepší hodnoty posuzovaného kritéria.
Globálně optimální triangulace je současně lokálně optimální.

Multikriteriálně optimalizované triangulace $\mathcal{T}$:
Kombinace několika lokálních či globálních kritérií.
Vycházejí z Delaunay triangulace, která je optimalizována k těmto kritériím.
Dlouhé výpočetní časy, doposud nejsou známy efektivní algoritmy, použití genetických algoritmů.

11. Hodnocení triangulace

Množina bodů $P = \{p_1, p_2, p_3, p_4\}$.
$\forall p_i \in \mathcal{H}$, konvexní čtyřúhelník.

Pak dle Eulerovy věty pro $n = 4$, $k = 4$ platí:$n_h = 5$, $n_t = 2$.

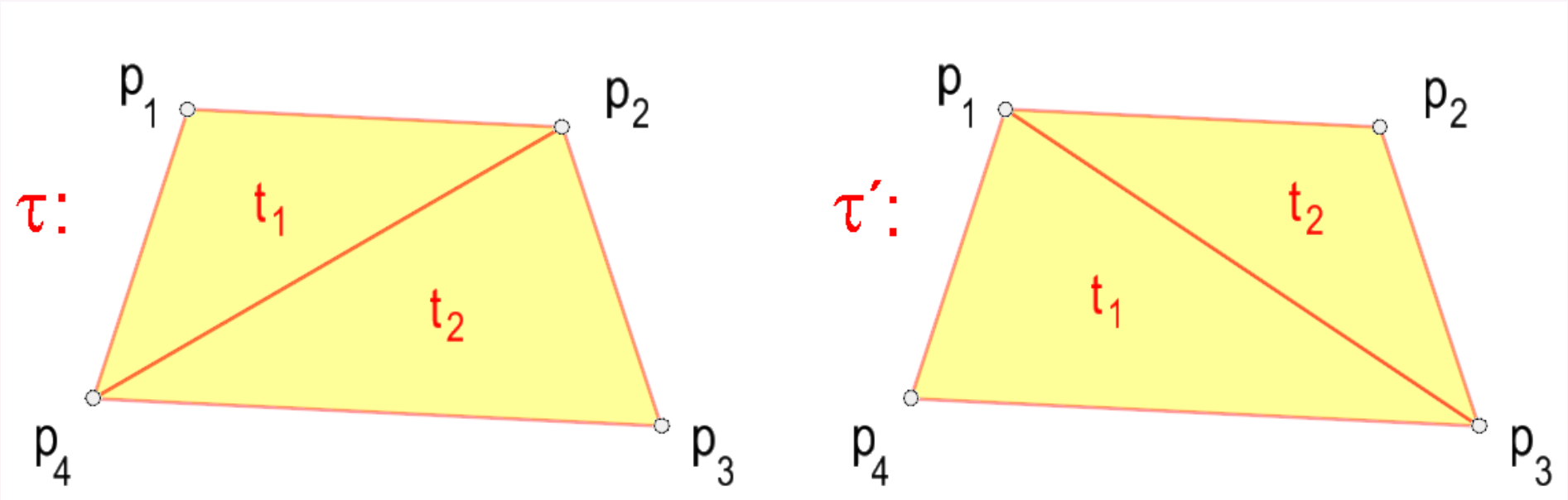
Důsledek: existují dvě různé triangulace $\mathcal{T}(P)$ a $\mathcal{T}'(P)$:

$$\mathcal{T}(P) = \{t_1(p_1, p_2, p_3), t_2(p_1, p_3, p_4)\},$$
$$\mathcal{T}'(P) = \{t'_1(p_1, p_2, p_4), t'_2(p_2, p_3, p_4)\}.$$

Vhledem k posuzovanému kritériu je jedna z triangulací *optimální*, tj. minimalizuje ho.

Tato pravidla lze zobecnit pro $n > 4$, triangulace se tak ke každé dvojici trojúhelníků.

12. Ilustrace $\mathcal{T}(P)$ a $\mathcal{T}'(P)$



13. Lokální kritéria

Mají geometrický podtext, snaha o generování trojúhelníků “rozumných” tvarů.

Přehled nejčastěji používaných lokálních kritérií:

- Minimální/maximální úhel v trojúhelníku α.
- Minimální/maximální výška v trojúhelníku v.
- Minimální/maximální poloměr vesané kružnice r.
- Minimální/maximální poloměr opsané kružnice R.
- Minimální/maximální plocha trojúhelníku S.
- Úhel mezi normálami sousedních trojúhelníků.

Nejčastěji používáno první kritérium (Delaunay triangulace maximalizuje minimální úhel).
Kritérium úhlu mezi normálami používáno u datově závislých triangulací.

Od každého kritéria dispozici min-max varianta či max-min varianta.

14. MIN/MAX strategie

Vrcholy konvexního 4-úhelníku $P = \{p_i, p_j, p_k, p_l\}$

$$\mathcal{T}(P) = \{t_1(p_i, p_l, p_j), t_1(p_i, p_j, p_k)\}, \qquad \mathcal{T}'(P) = \{t_1'(p_i, p_l, p_k), t_2'(p_l, p_j, p_k)\}.$$

Min-max kritérium:

Výpočet maximální hodnoty c nad $\mathcal{T}(P)$ i $\mathcal{T}'(P)$

$$\overline{c} = \max_{\mathcal{T}}(c), \qquad \overline{c}' = \max_{\mathcal{T}'}(c').$$

Minimalizace maximální hodnoty kritéria c v triangulaci

$$\mathcal{T} = \begin{cases} \mathcal{T}(P), & \overline{c} \leq \overline{c}', \\ \mathcal{T}'(P), & \overline{c} > \overline{c}'. \end{cases}$$

Max-min kritérium:

Výpočet minimální hodnoty c nad $\mathcal{T}(P)$ i $\mathcal{T}'(P)$

$$\underline{c} = \min_{\mathcal{T}}(c), \qquad \underline{c}' = \min_{\mathcal{T}'}(c').$$

Minimalizace maximální hodnoty kritéria c v triangulaci

$$\mathcal{T} = \begin{cases} \mathcal{T}(P), & \underline{c} \geq \underline{c}', \\ \mathcal{T}'(P), & \underline{c} < \underline{c}'. \end{cases}$$

15. MIN/MAX strategie, úhel v trojúhelníku

Odstranění trojúhelníků s příliš ostrými/tupými úhly $\Rightarrow$ tvarově nevhodné.

Min-max kritérium:

Eliminace trojúhelníků s příliš tupými úhly

$$\overline{\alpha} = \max_{\mathcal{T}}(\alpha), \qquad \overline{\alpha}' = \max_{\mathcal{T}'}(\alpha').$$

Minimalizace maximálního úhlu v triangulaci

$$\mathcal{T} = \begin{cases} \mathcal{T}(P), & \overline{\alpha} \leq \overline{\alpha}', \\ \mathcal{T}'(P), & \overline{\alpha} > \overline{\alpha}'. \end{cases}$$

Max-min kritérium:

Eliminace trojúhelníků s příliš ostrými úhly

$$\underline{\alpha} = \min_{\mathcal{T}}(\alpha), \qquad \underline{\alpha}' = \min_{\mathcal{T}'}(\alpha').$$

Minimalizace maximálního úhlu v triangulaci

$$\mathcal{T} = \begin{cases} \mathcal{T}(P), & \underline{\alpha} \geq \underline{\alpha}', \\ \mathcal{T}'(P), & \underline{\alpha} < \underline{\alpha}'. \end{cases}$$

16. Globální kritéria

Optimalizují geometrické parametry všech trojúhelníků v triangulaci $\mathcal{T}(P)$.

Nejčastěji používaná kritéria:

Suma délek stran:
Minimalizace celkovou délky hran h_i triangulace $\mathcal{T}(P)$

$$\sum_{i=1}^{n_h} h_i = min.$$

MWT (Minimum Weight Triangulation), NP problém.
Přibližné řešení: genetické algoritmy, Greedy triangulace.

Povinné hrany:
Předem definované hrany uvnitř triangulace, tzv. *Constrained Triangulation*.
$\mathcal{T}(P)$ není lokálně optimální.

Zadání charakteristických hran terénních tvarů, lepší aproximace terénu.

17. Greedy triangulace

Patří do skupiny hladových algoritmů (Greedy Algorithms)

$$W(\mathcal{T}(P)) = \sum_{i=1}^{n_h} \|h_i\|_2 = \min,$$

kde

$$\mathcal{T}(P(n)) = \mathcal{T}(P(n-1)) + \Delta\mathcal{T}, \qquad h_{[n]} = \arg \min_{\forall h_i \in h \setminus h_{[1]}, \ldots, \setminus h_{[n-1]}} \left(\|h_i\|_2\right).$$

Vlastnosti triangulace:
Pokud se v P nevyskytují hrany se stejnou délkou, je triangulace jednoznačná.
Snaží se však vytvářet trojúhelníky s nejkratšími stranami.
Ttrojúhelníky nemusí splňovat žádnou speciální geometrickou podmínku.
Jednoduchá implementace.
Složitost je $O(n^3)$, lze optimalizovat na $O(n^2 \cdot \log(n))$.

Důsledek:
Síť trojúhelníků není z tvarového hlediska optimalizována,
Do triangulace tak mohou být přidány tvarově nevhodné trojúhelníky.
V kartografii není příliš často používána.

Výsledná triangulace se blíží MWT.

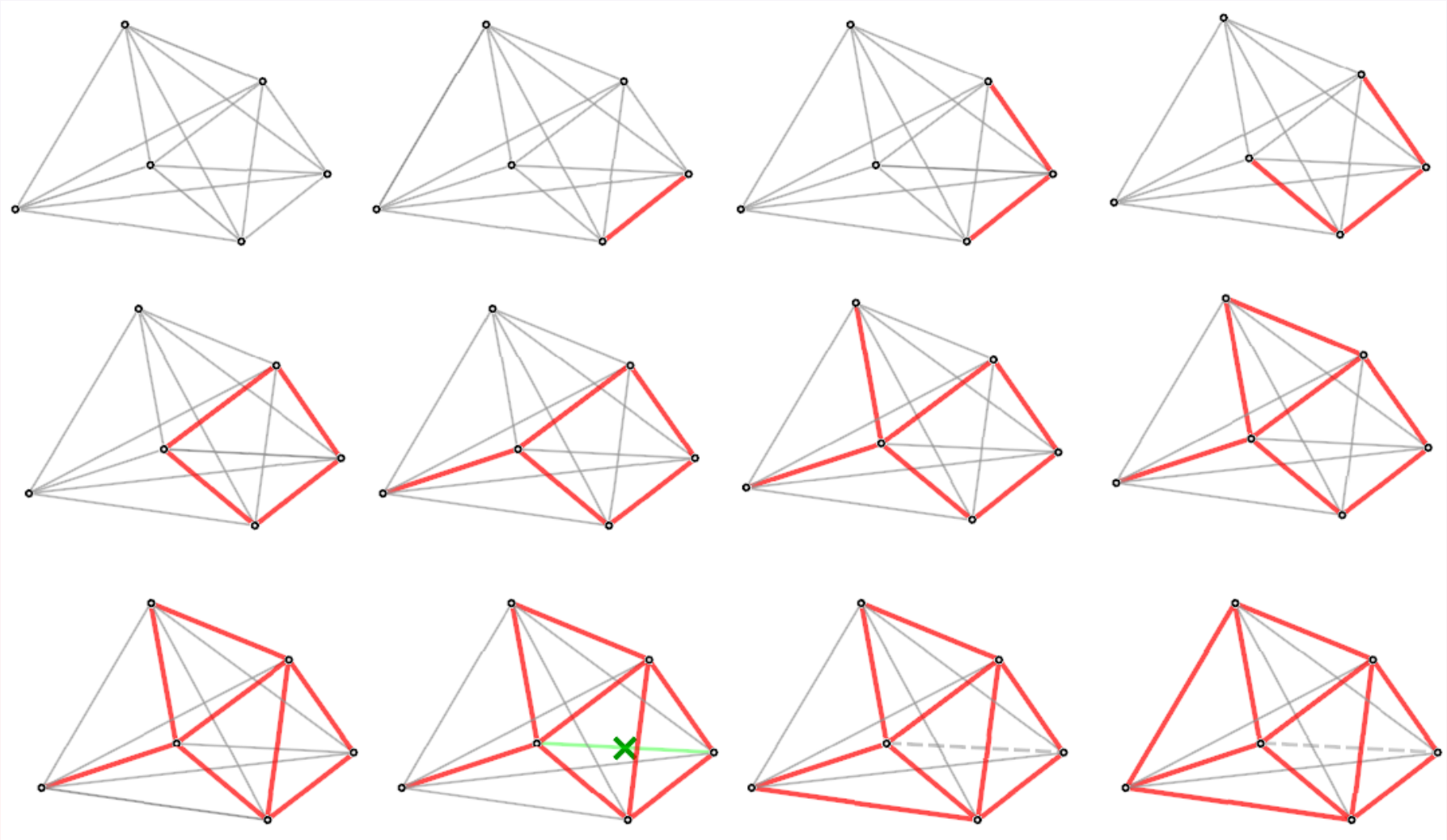
18. Algoritmus Greedy triangulace

Využití prioritní fronty PQ.

Algoritmus 1: Greedy Triangulation (S, $\mathcal{T}$)

1: $PQ = \emptyset$
2: for $\forall p_i$, $i \in \langle 1, n \rangle$:
3: for $j \in \langle i+1, n \rangle$:
4: Vytvoř hranu $h = (p_i, p_j)$.
5: $PQ \leftarrow (\|h\|, h)$.
6: $\mathcal{T} \leftarrow Q.pop()$.
7: while PQ not empty:
8: $h = PQ.pop()$
9: $intersect = false$
10: for $\forall h_i \in \mathcal{T}$:
11: if $(h \cap h_i \neq \emptyset)$
12: $intersect = true$
13: break:
14: if (!$intersect$) $\mathcal{T} \leftarrow h$.

19. Grafické znázornění Greedy triangulace



20. Delaunay triangulace $\mathcal{DT}$ a její vlastnosti

Nejčastěji používaná triangulace, v oblasti GIS de-facto standart.
Existuje v $\mathbb{R}^2$ i v $\mathbb{R}^3$.

V1:
Uvnitř kružnice k opsané libovolnému trojúhelníku $t_j \in \mathcal{DT}$ neleží žádný jiný bod množiny P.

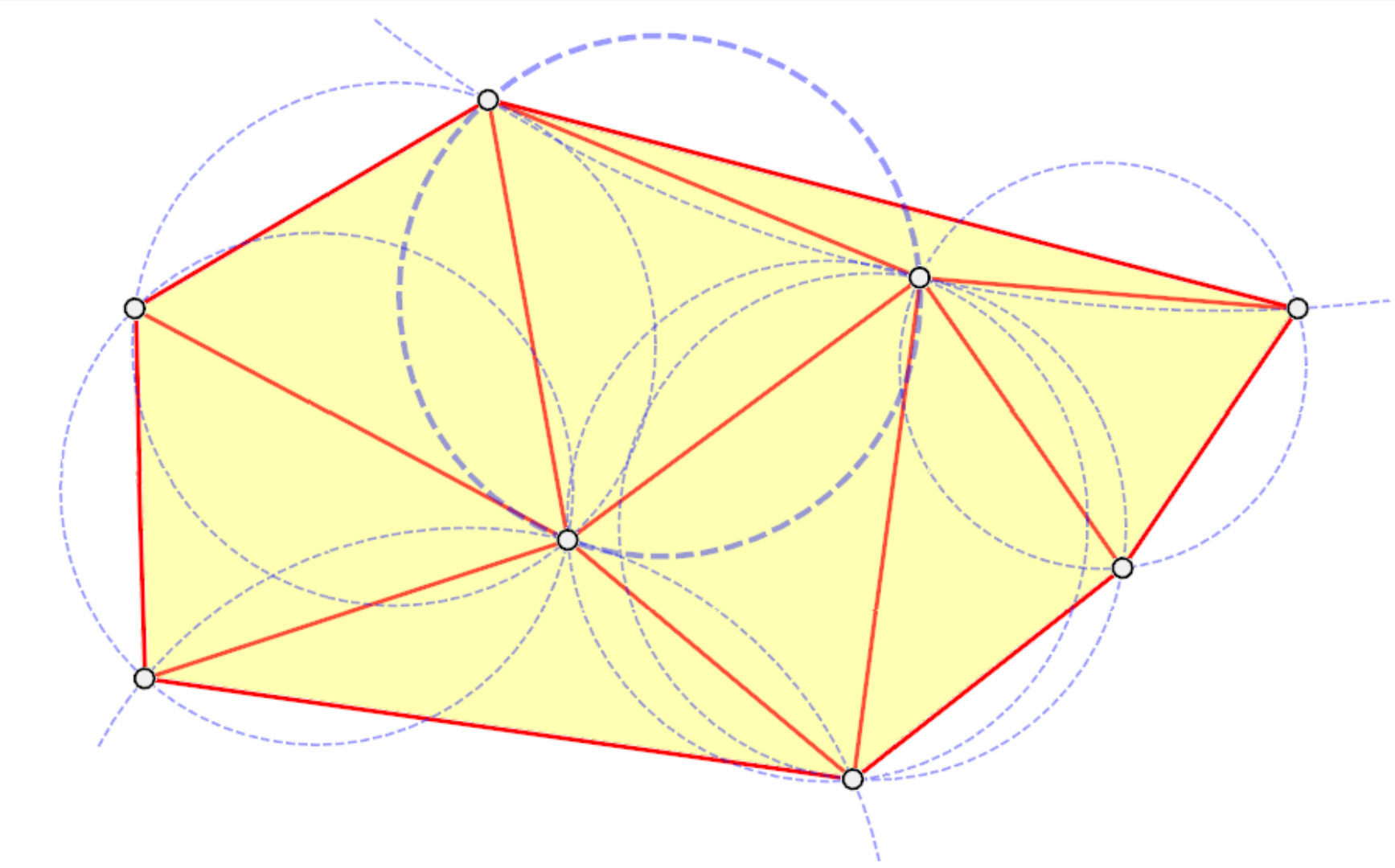
V2:
$\mathcal{DT}$ maximalizuje minimální úhel v $\forall t$, avšak $\mathcal{DT}$ neminimalizuje maximální úhel v t.

V3:
$\mathcal{DT}$ je lokálně optimální i globálně optimální vůči kritériu mnimálního úhlu.

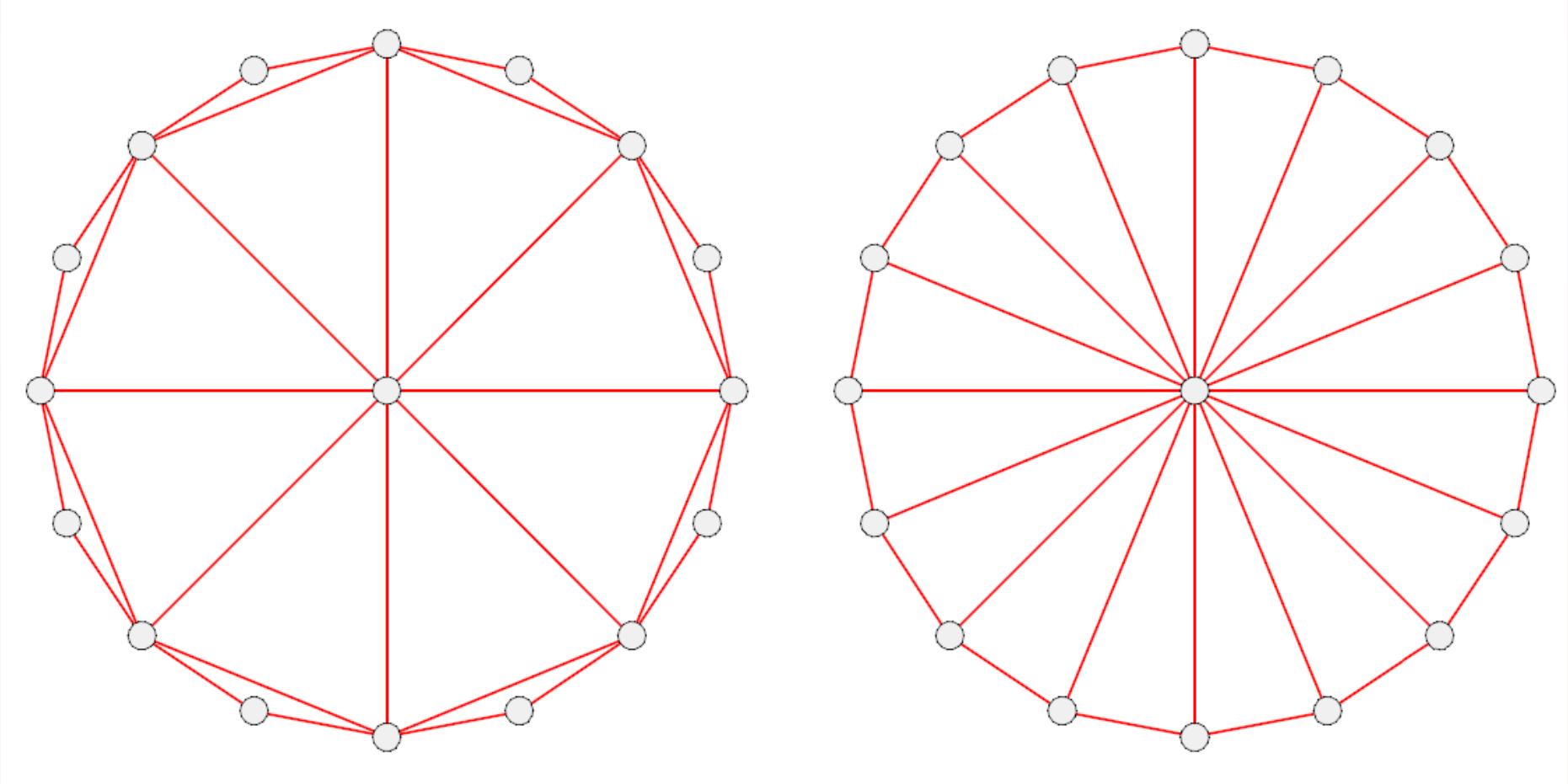
V4:
$\mathcal{DT}$ je jednoznačná, pokud žádné čtyři body neleží na kružnici.

Výsledné trojúhelníky se při porovnání ze všemi známými triangulacemi nejvíce blíží rovnostranným trojúhelníkům.

21. Ukázka $\mathcal{DT}$



22. Srovnání $\mathcal{GT}$ a $\mathcal{DT}$

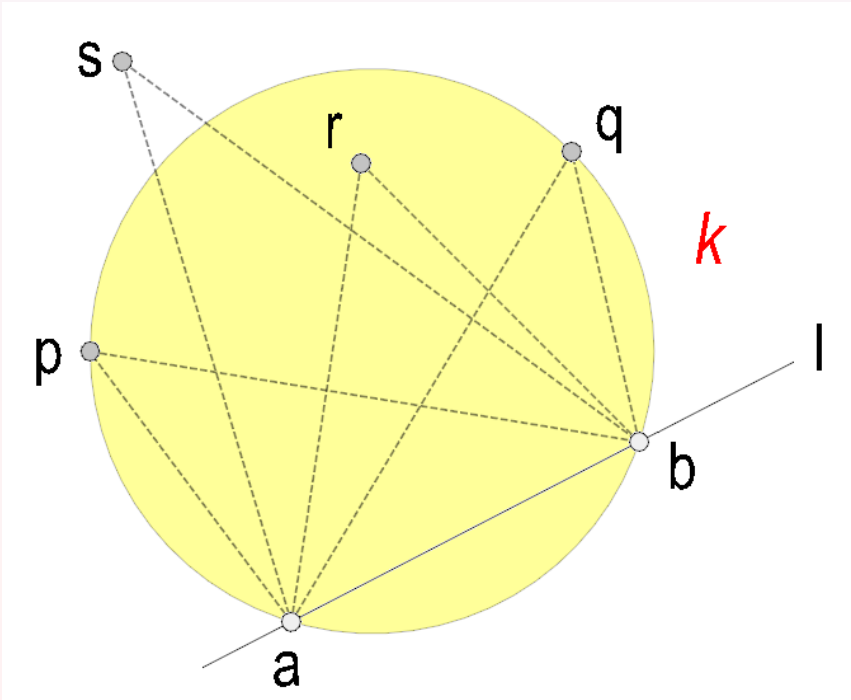


23. Geometrické vlastnosti $\mathcal{DT}$

Nechť k je kružnice l přímka protínající k v bodech a, b, a p, q, r, s jsou body ležící na stejné straně od l.
Platí:

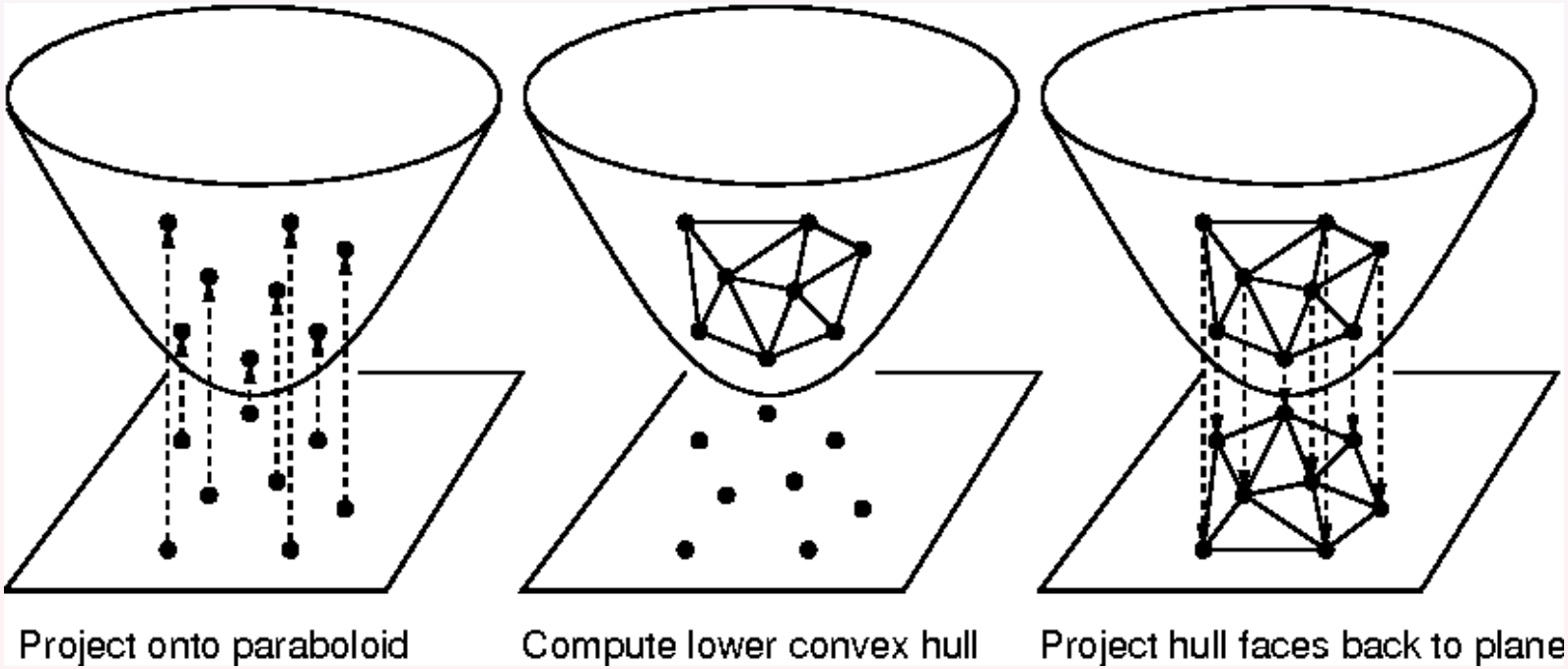
$$\angle arb > \angle apb = \angle aqb > asb.$$

Důsledek Tháletovy věty.



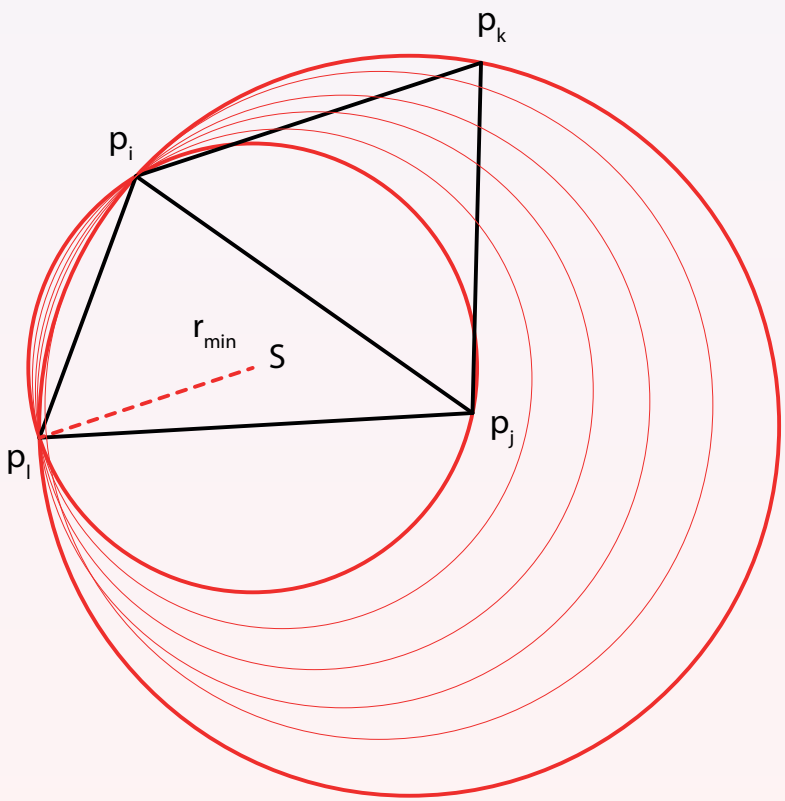
23. Souvislost $\mathcal{DT}$ a $\mathcal{H}$

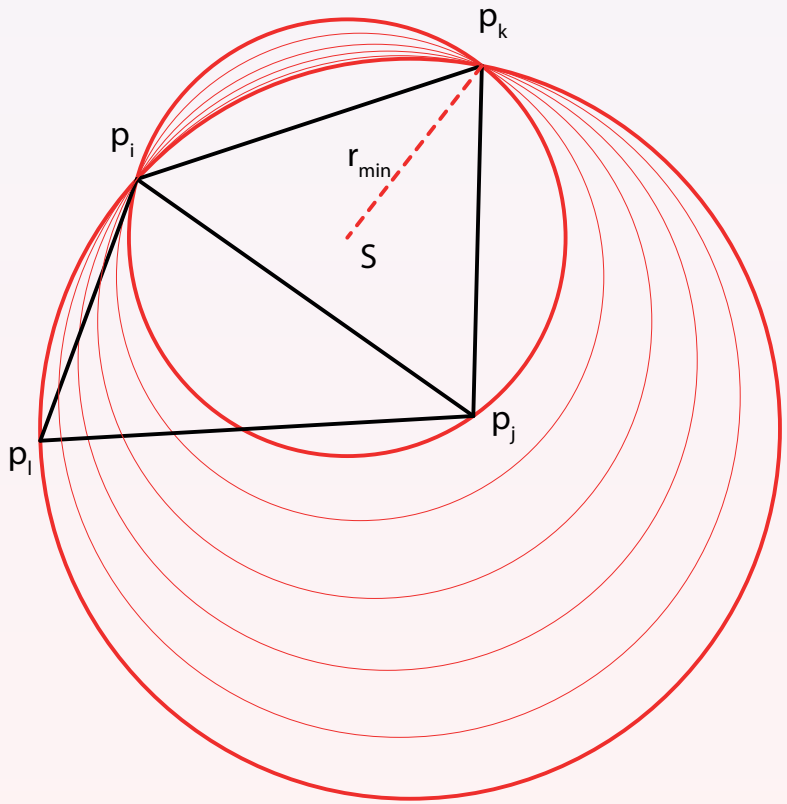
Projekce P na paraboloid.
Konstrukce $\mathcal{H}(P)$.
Projekce $\mathcal{H}(P)$ do roviny.



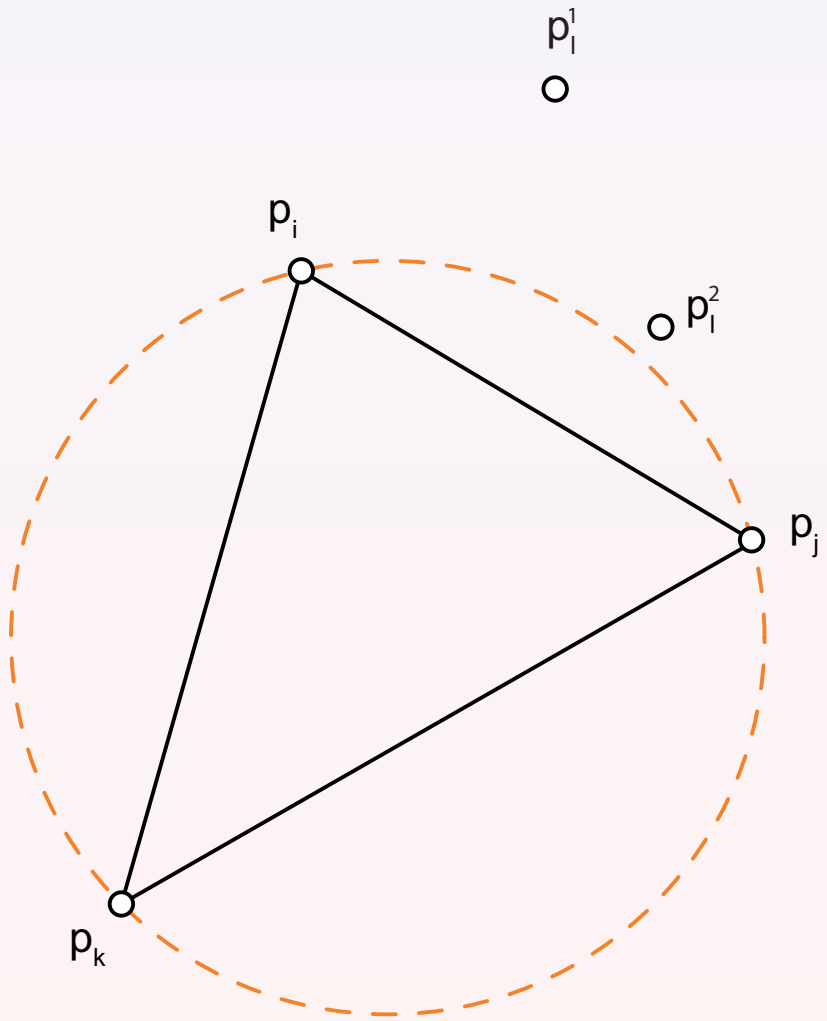
24. Nejmenší opsaná kružnice

Generuje Delaunay trojúhelník nejmenší opsanou kružnici $k(S, r_{min})$?
Lze použít jako alternativní kritérium?

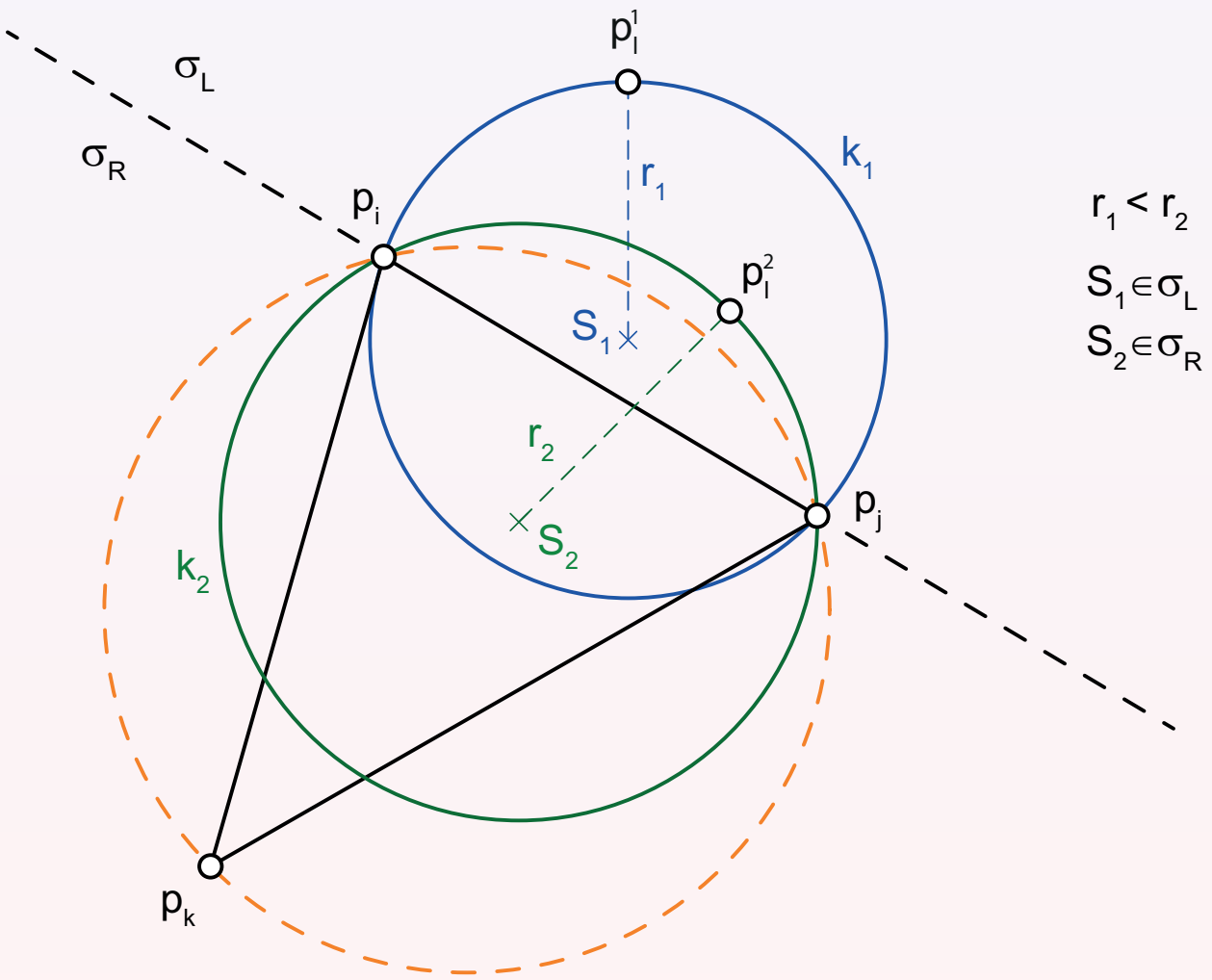




25. Který z bodů splňuje podmínku DT (1/2)?



26. Který z bodů splňuje podmínku DT (2/2)?



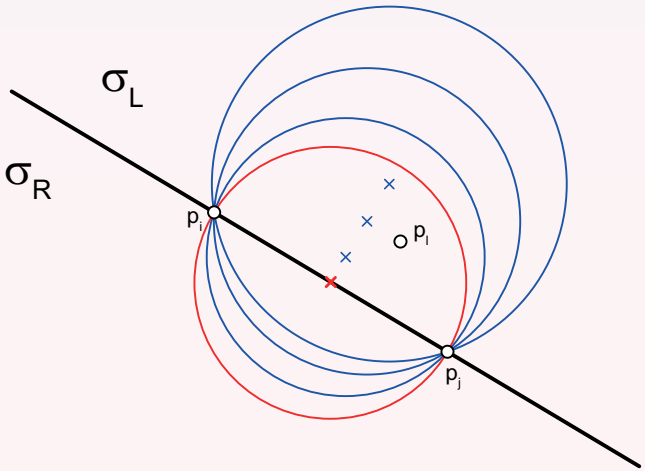
Pozor: $p_l^2 \in k(p_i, p_j, p_l^1)$, ale $p_l^1 \notin k(p_i, p_j, p_l^2)$, ačkoliv $r_1 < r_2$!

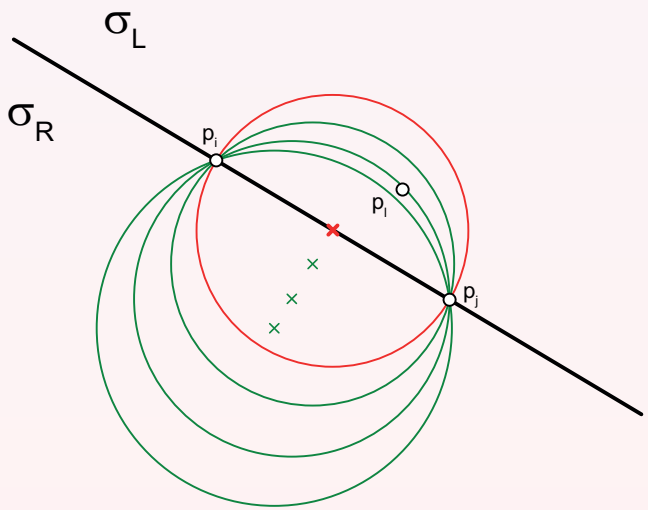
27. Důsledek

Korektní bod p_l nemusí generovat kružnici $k(S, r_{min})$.
Preferováno řešení v pravé polorovině před levou

$$r' = \begin{cases} -r, & S \in \sigma_R, \\ r, & S \in \sigma_L. \end{cases}$$

Nutno testovat, ve které polorovině střed S leží.





28. Edge Flip, legalizace

Vrcholy konvexního 4-úhelníku $P = \{p_i, p_j, p_k, p_l\}$, $P \in \mathcal{H}(P)$.

$$\mathcal{DT}(P) = \{t_1(p_i, p_l, p_k), t_1(l, p_j, p_k)\}, \qquad DT'(P) = \{t'_1(p_i, p_l, p_j), t'_2(p_i, p_j, p_k)\}.$$

Edge Flip (Swap)
Přechod $\mathcal{DT}(P) \Rightarrow DT'(P)$, prohození diagonály

$$(p_k, p_l) \Rightarrow (p_i, p_j).$$

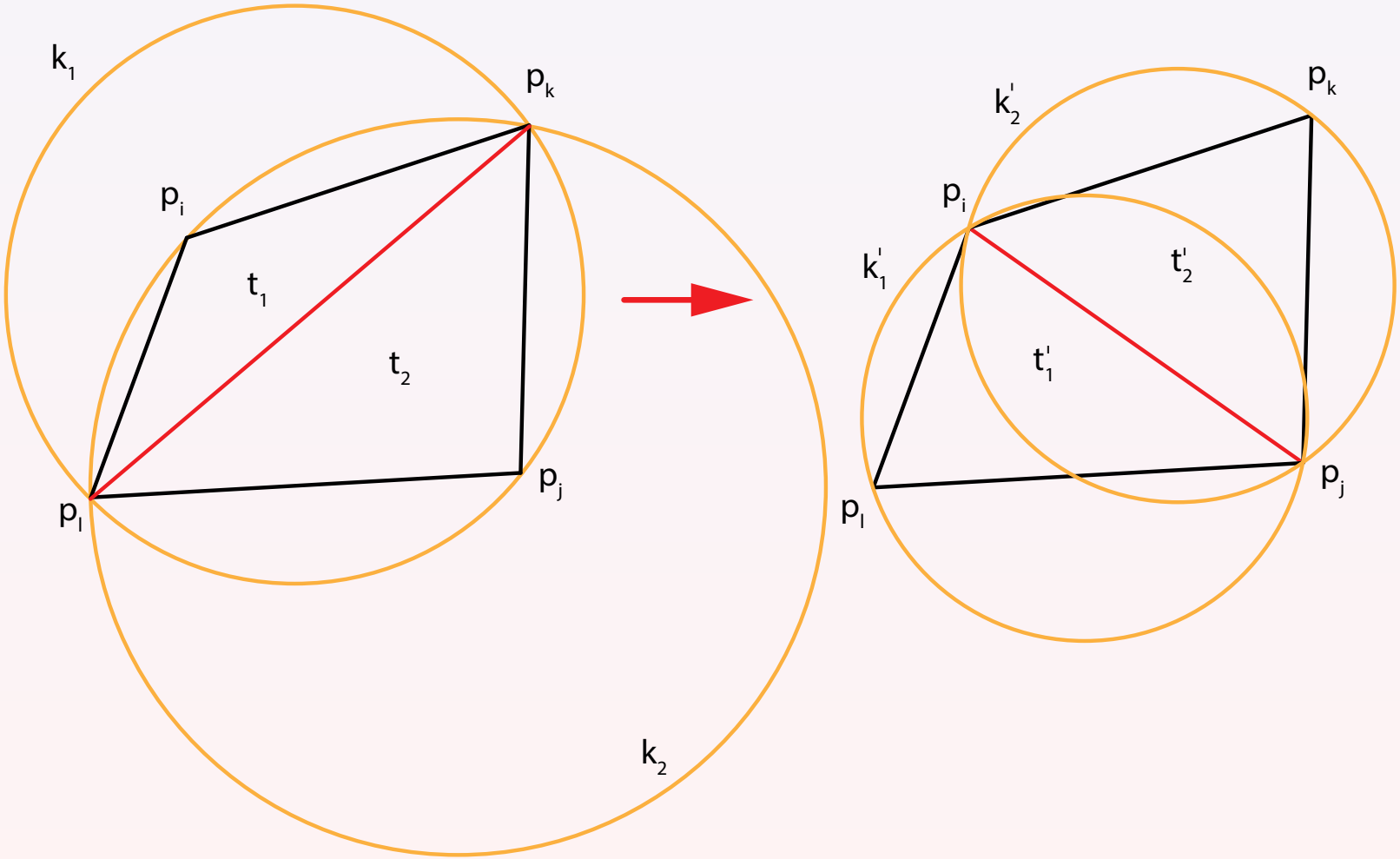
Trojúhelníky $t'_1(p_i, p_l, p_k)$, $t'_2(p_l, p_j, p_k)$ *legální*.
Jsou lokálně optimální vzhledem k max-min kritériu.

Poloměry opsaných kružnic $k_1(S_1, r_1)$, $k(S_2, r_2)$ a $k'_1(S'_1, r'_1)$, $k(S'_2, r'_2)$

$$r'_1 < r_1, \qquad r'_2 < r_2.$$

Opakovaně prováděna nad všemi konvexními čtyřúhelníky DT.
U nekonvexních min-max splněno automaticky.
Operace nazývána **legalizací**.

29. Edge Flip, legalizace



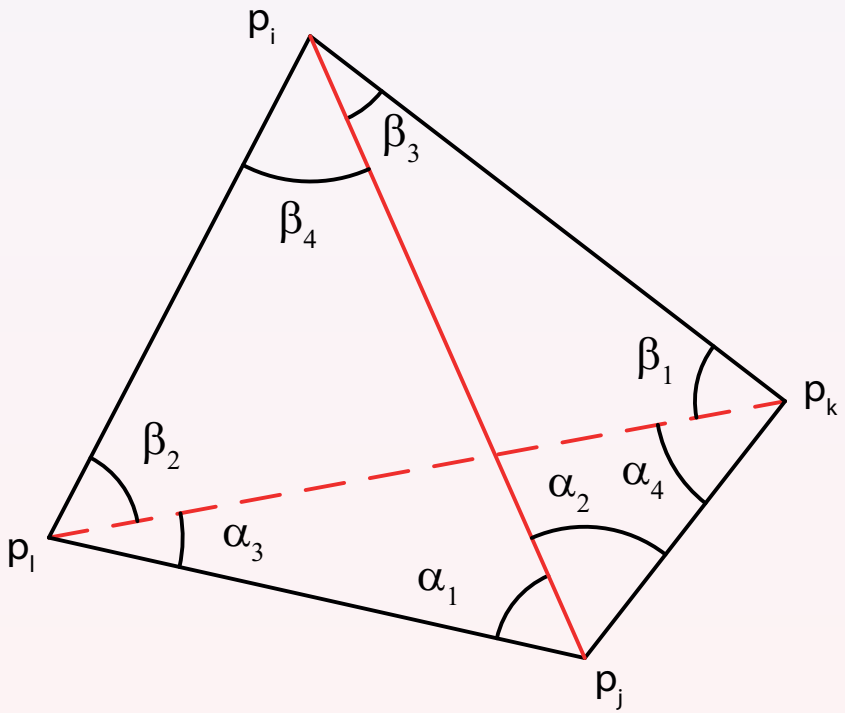
30. Vztah mezi úhly v $\mathcal{DT}(P)$ a $\mathcal{DT}'(P)$

Vnitřní úhly v trojúhelníkách, $\mathcal{DT}(P)$:

$$\alpha_1 + \alpha_2, \quad \alpha_3, \quad \alpha_4, \quad \beta_1, \quad \beta_2, \quad \beta_3 + \beta_4.$$

Vniřní úhly v trojúhelníkách, $\mathcal{DT}'(P)$:

$$\alpha_1, \quad \alpha_2, \quad \beta_3, \quad \beta_4, \quad \beta_1 + \alpha_4, \quad \beta_2 + \alpha_3.$$



Pro každý úhel po swapu existuje nejméně jeden menší před swapem

$$\alpha_1 > \beta_1, \quad \alpha_2 > \beta_2, \quad \beta_3 > \alpha_3,$$

$$\beta_4 > \alpha_4, \quad \beta_1 + \alpha_4 > \alpha_4, \quad \beta_2 + \alpha_3 > \alpha_3.$$

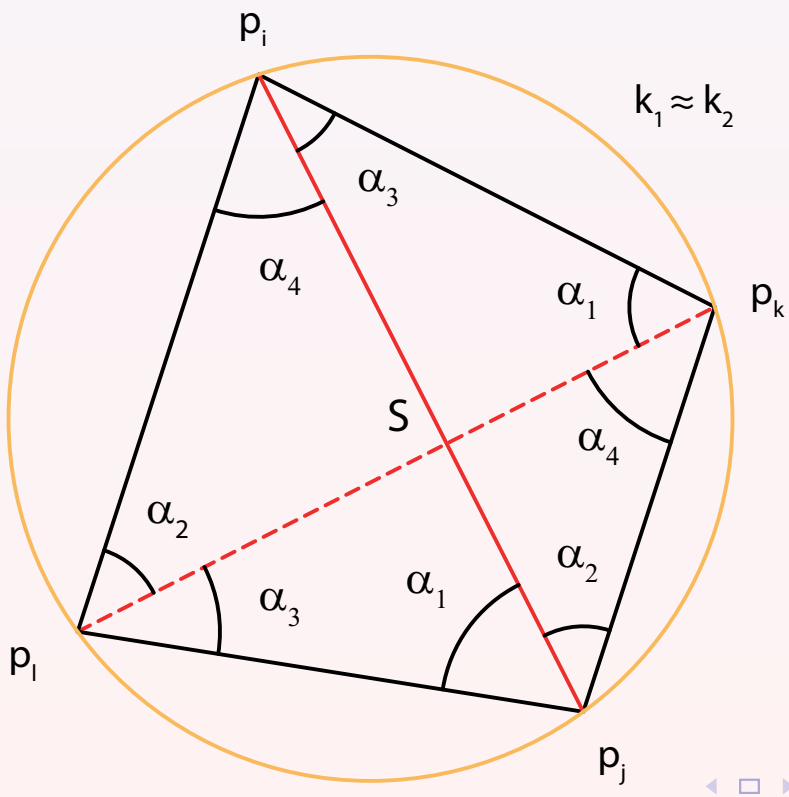
31. Nejednoznačnost DT

Pokud body $\{p_i, p_j, p_k, p_l\}$ na kružnici $k(S, r)$, pak $\mathcal{DT}(P) \equiv \mathcal{DT}'(P)$.
Vnitřní úhly v trojúhelníkách, $\mathcal{DT}(P)$:

$$\alpha_1 + \alpha_2, \quad \alpha_3, \quad \alpha_4, \quad \alpha_1, \quad \alpha_2, \quad \alpha_3 + \alpha_4.$$

Vnitřní úhly v trojúhelníkách, $\mathcal{DT}'(P)$:

$$\alpha_1, \quad \alpha_2, \quad \alpha_3, \quad \alpha_4, \quad \alpha_1 + \alpha_4, \quad \alpha_2 + \alpha_3.$$



32. Test legality, opsaná kružnice

Kružnice $k(p_1, p_2, p_3)$, kde $p_1 = [x_1, y_1]$, $p_2 = [x_2, y_2]$, $p_3 = [x_3, y_3]$.
Analyzovaný bod $p = [x, y]$.

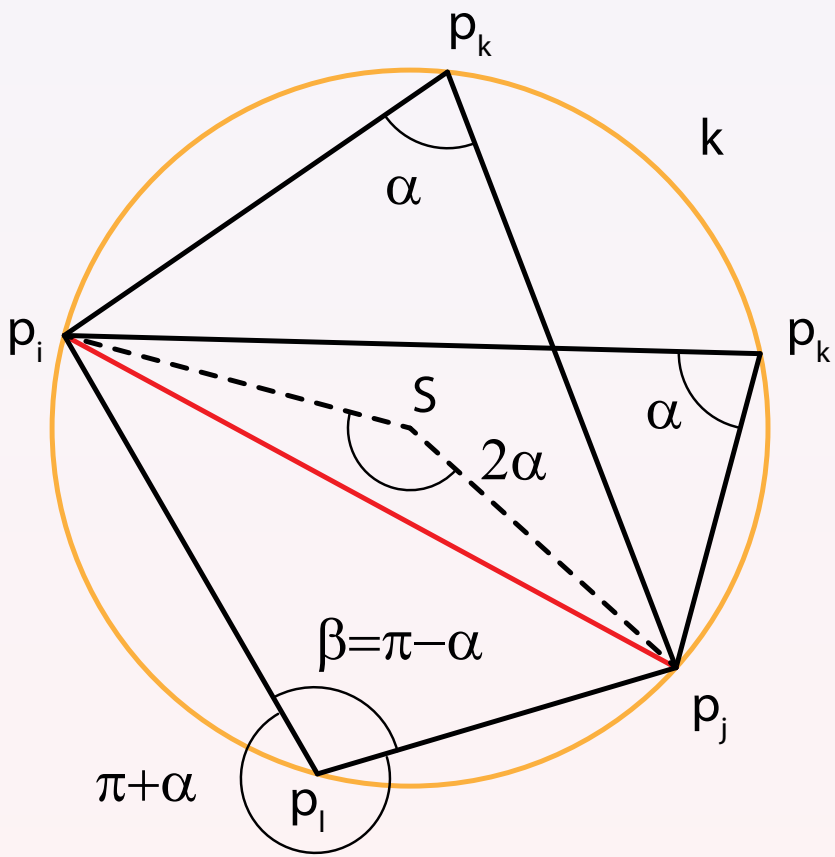
Předpoklad: body p_1, p_2, p_3 mají CW orientaci.

Testovací kritérium t:

$$t = \begin{vmatrix} x & y & x^2 + y^2 \\ x_1 & y_1 & x_1^2 + y_1^2 \\ x_2 & y_2 & x_2^2 + y_2^2 \\ x_3 & y_3 & x_3^2 + y_3^2 \end{vmatrix} \Rightarrow \begin{cases} t > 0, & P \notin k, \\ t = 0, & P \in \partial k, \\ t < 0, & P \in k. \end{cases}$$

Průmyslový test, pouze +, -, *.

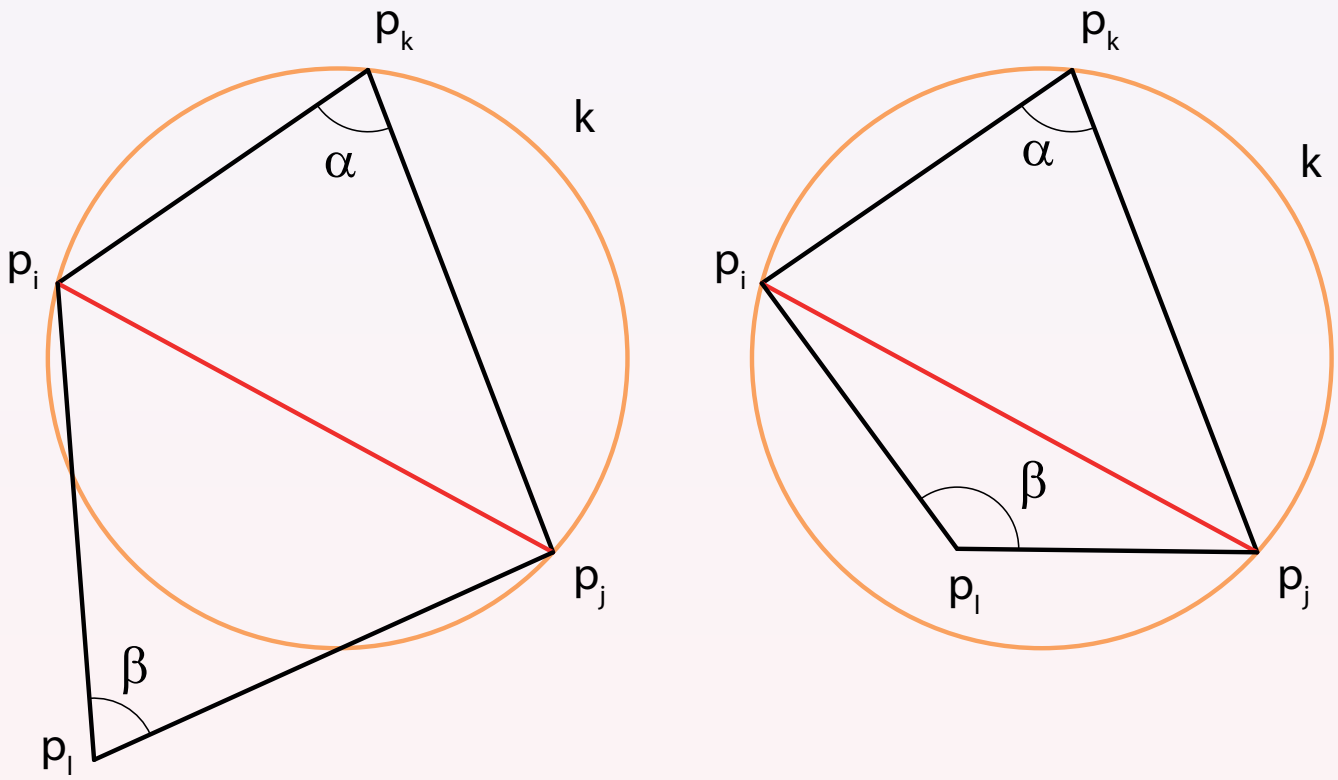
33. Úhly ve čtyřúhelníku (1/2)



$$\alpha + \beta = \alpha + \pi - \alpha = \pi.$$

34. Úhly ve čtyřúhelníku (2/2)

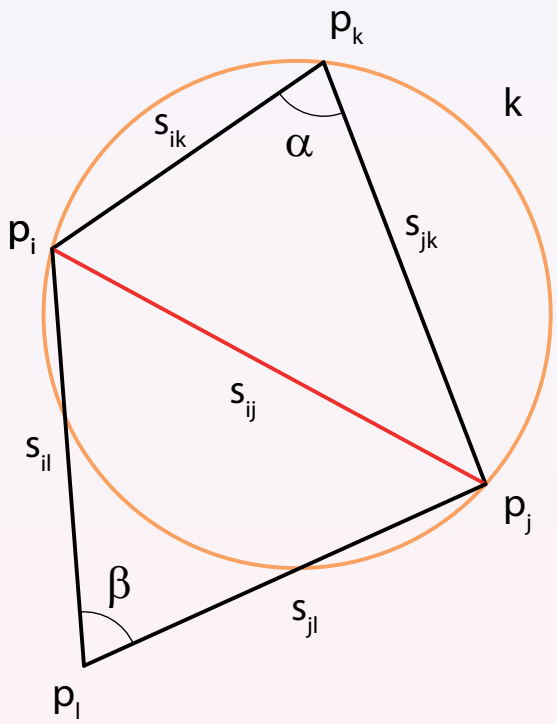
Legální vs nelegální triangulace:



$$\alpha + \beta \begin{cases} < \pi, & DT(P) \text{ je legální,} \\ = \pi, & \text{Body } p_i, p_j, p_k, p_l \text{ leží na kružnici,} \\ > \pi, & DT(P) \text{ je nelegální.} \end{cases}$$

35. Test legality, úhly

Numericky robustní test ověřující legalitu dvojice trojúhelníků $t_1(p_i, p_j, p_k)$ *a* $t_2(p_i p_j, p_l)$.



Nelegální swap

$$\sin(\alpha + \beta) = \cos\alpha \sin\beta + \cos\beta \sin\alpha < \sin(\pi),$$

kde

$$\cos\alpha = \frac{s_{ik}^2 + s_{jk}^2 - s_{ij}^2}{2 s_{ik} s_{jk}}, \qquad \cos\beta = \frac{s_{il}^2 + s_{jl}^2 - s_{ij}^2}{2 s_{il} s_{jl}}.$$

36. Test legality, odvození

Dosazení

$$\cos^2\alpha = \frac{((x_i - x_k)(x_j - x_k) + (y_i - y_k)(y_j - y_k))^2}{((x_i - x_k)^2 + (y_i - y_k)^2)((x_j - x_k)^2 + (y_j - y_k)^2)},$$
$$\cos^2\beta = \frac{((x_i - x_l)(x_j - x_l) + (y_i - y_l)(y_j - y_l))^2}{((x_i - x_l)^2 + (y_i - y_l)^2)((x_j - x_l)^2 + (y_j - y_l)^2)}.$$

Pak

$$\sin^2\alpha = 1 - \cos^2\alpha = \frac{(x_k(y_j - y_i) + x_j(y_i - y_k) + x_i(y_k - y_j))^2}{((x_i - x_k)^2 + (y_i - y_k)^2)((x_j - x_k)^2 + (y_j - y_k)^2)},$$
$$\sin^2\beta = 1 - \cos^2\beta = \frac{(x_l(y_j - y_i) + x_j(y_i - y_l) + x_i(y_l - y_j))^2}{((x_i - x_l)^2 + (y_i - y_l)^2)((x_j - x_l)^2 + (y_j - y_l)^2)}.$$

Vyjádříme $\sin\alpha$, $\sin\beta$, $\cos\alpha$, $\cos\beta$ a dosadíme do swapovací podmínky.

37. Test legality

Podmínka přejde do tvaru

$$\frac{(x_{ik}x_{jk} + y_{ik}y_{jk})(x_{jl}y_{il} - x_{il}y_{jl}) + (x_{jk}y_{ik} - x_{ik}y_{jk})(x_{jl}x_{il} + y_{jl}y_{il})}{[(x_{ik}^2 + y_{ik}^2)(x_{jk}^2 + y_{jk}^2)(x_{jl}^2 + x_{jl}^2)(x_{il}^2 + x_{il}^2)]^{1/2}} < 0.$$

Jmenovatel vždy kladný.
Swapujeme, pokud

$$(x_{ik}x_{jk} + y_{ik}y_{jk})(x_{jl}y_{il} - x_{il}y_{jl}) < (x_{jk}y_{ik} - x_{ik}y_{jk})(x_{jl}x_{il} + y_{jl}y_{il}),$$

kde

$$\begin{aligned} x_{ik} &= x_i - x_k & y_{ik} &= y_i - y_k, \\ x_{jk} &= x_j - x_k & y_{jk} &= y_j - y_k, \\ x_{jl} &= x_j - x_l & y_{jl} &= y_j - y_l, \\ x_{il} &= x_i - x_l & y_{il} &= y_i - y_l. \end{aligned}$$

Test numericky stabilní, pouze +, -, *.
Průmyslový standard.

38. Datový model triangulace

Datová struktura používaná při konstrukci $\mathcal{DT}$, uchovává topologii trojúhelníků v $\mathcal{DT}$.
Nechť dva incidující trojúhelníky $t_i, t_j \in \mathcal{DT}$ se společnou $e_{ij} \in t_i$ a $e_{ji} \in t_j$.

Strany trojúhelníků:
Každá strana e_{ij} v trojúhelníku t_i v CCW orientaci uchovává:

- pointer na následující hranu $e_{i+1,j}$ v t_i.
- pointer na stranu e_{ji} v incidujícím trojúhelníku t_j

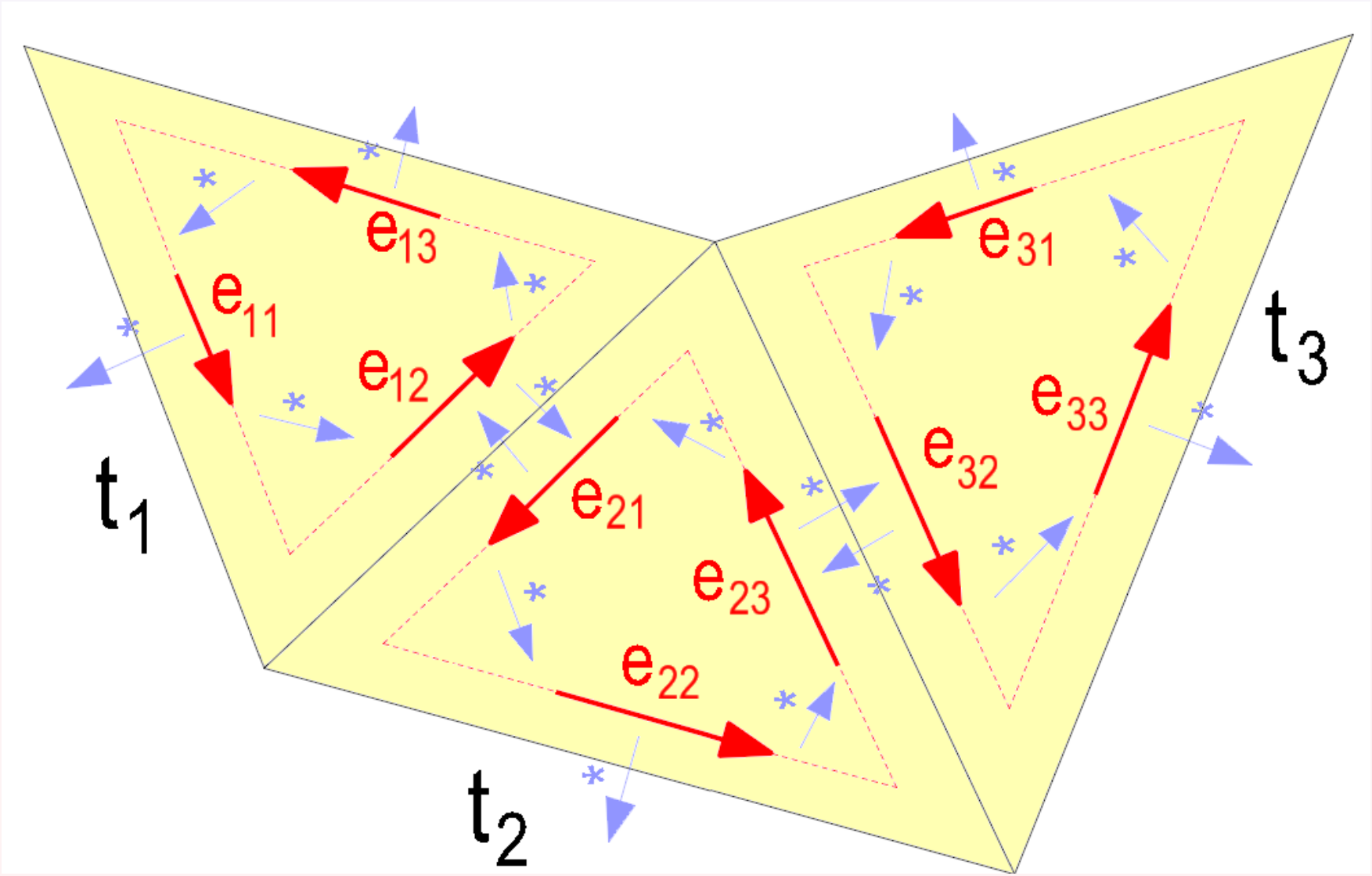
Strany ležící na $\mathcal{H}$ mají pointer inicializovaný na NULL).
Kromě stran na $\mathcal{H}$ každá e popsána dvakrát (jako e_{ij} a e_{ji}).
Strany e_{ij} a e_{ji} mají opačnou orientací.

Tyto zdvojené strany nazývány *Twin Edges*.

Trojúhelníky:
Každý trojúhelník t_i popsán trojicí hran $(e_{ij}, e_{i+1,j}, e_{i+2,j})$, CCW orientace.
Tvoříkruhový seznam (Circular List) .

Pro každou hranu lze snadno nalézt předcházející/následující hranu a incidující trojúhelníky.

39. Ukázka datového modelu



40. Metody konstrukce $\mathcal{DT}$

Metody přímé konstrukce $\mathcal{DT}$:

- Lokální prohazování.
- Inkrementální konstrukce.
- Inkrementální vkládání.
- Rozděl a panuj.
- Sweep Line.

Nepřímá konstrukce: přes Voronoi diagram, v praxi není používána.

Metoda lokálního prohazování:
Metoda je použitelná pouze ve 2D, obtížně lze převést do vyšší dimenze.
Převod libovolné triangulace $\mathcal{T}$ na $\mathcal{DT}$.
Prohazování nelegálních hran v dvojicích trojúhelníků tvořících konvexní čtyřúhelník.
Složitost algoritmu je $O(N)$, nutno připočítat složitost na triangulačního algoritmu.
Lze použít vzhledem k libovolnému kritériu, např. DDT.

41. Algoritmus lokálního prohazování

Algoritmus 2: Delaunay Triangulation Local(P)

1: Vytvoř pomocnou triangulaci $T(P)$.
2: legal=false
3: while $T(P)$!legal
4: legal=true;
5: Opakuj pro $\forall e_i \in T(P)$
6: Vezmi hranu $e_i \in T(P)$
7: Nalezni trojúhelníky t_1, t_2 incidující s e_i.
8: if $(t_1 \cup t_2)$ konvexní a nelegální
9: Legalize (t_1, t_2).
10: legal=false;

42. Metoda inkrementální konstrukce

Algoritmus lze použít ve 2D i 3D.
2D varianta pracuje s prázdnou kružnicí, 3D varianta s prázdnou koulí o poloměru r.

Založena na postupném přidávání bodů do již vytvořené $\mathcal{DT}$.

Nad existující Delaunayovskou hranou $e = (p_1, p_2)$ hledán $\underline{p}$, minimalizující poloměr $k_i = (e, p_i)$, $p_i \in \sigma_r(e)$

$$\underline{p} = \arg \min_{\forall p_i \in \sigma_L(e)} r(k_i), \qquad k_i = (a, b, p_i), \qquad e = (a, b).$$

Delaunayovská hrana je orientována, bod p hledáme pouze vlevo od ní.
Alternativně test prázdné opsané kružnice.

Do $\mathcal{DT}$ přidány hrany trojúhelníku $\triangle(p_1, p_2, \underline{p})$:

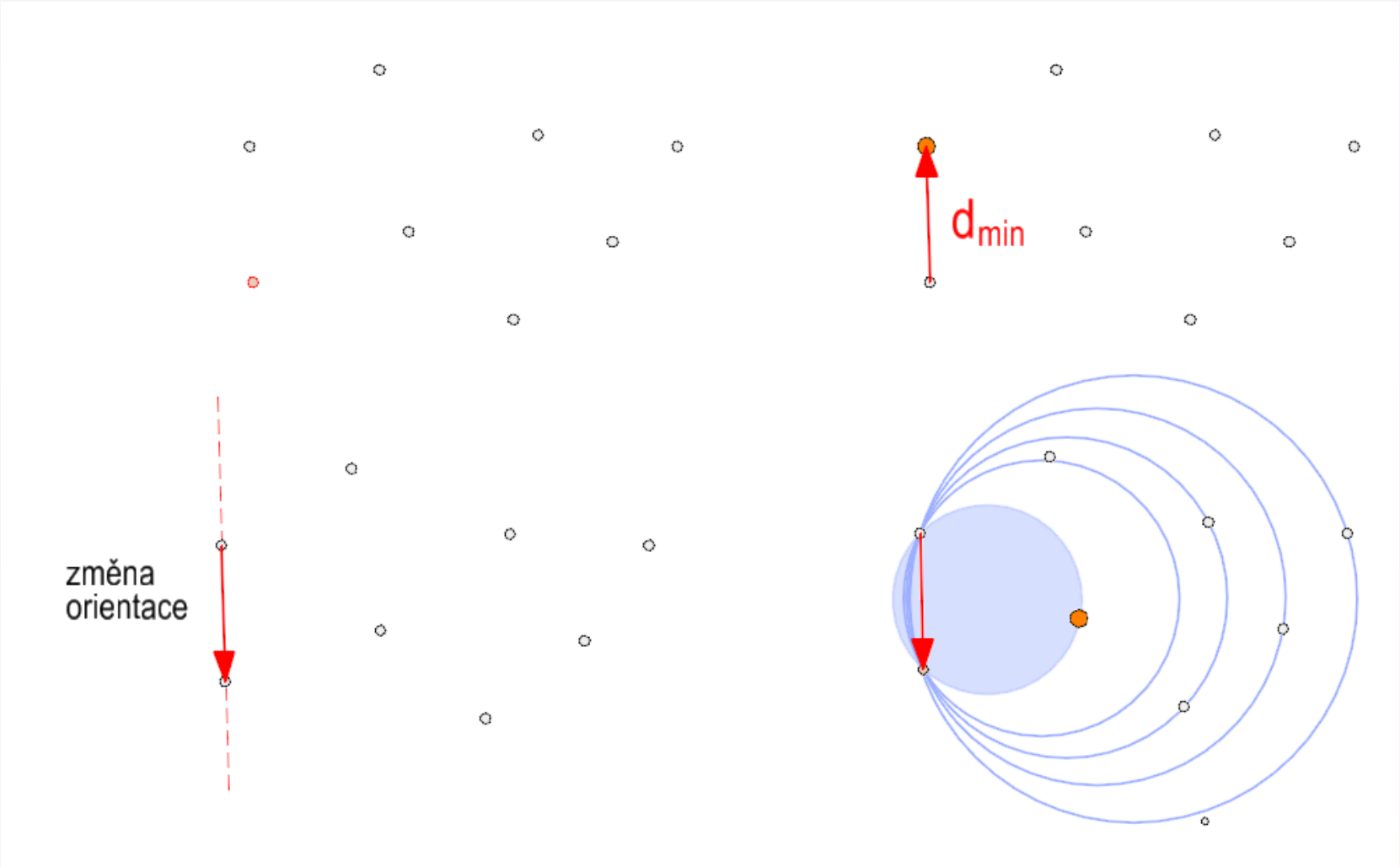
$$e_1 = (p_2, \underline{p}), e_2 = (\underline{p}, p_1),$$

pakliže hrany $e_1' = (\underline{p}, p_2,), e_2' = (p_1, \underline{p},)$ nejsou v AEL.
Pokud p nalezen v $\sigma_r(e)$, změníme orientaci hrany e a hledání opakujeme.

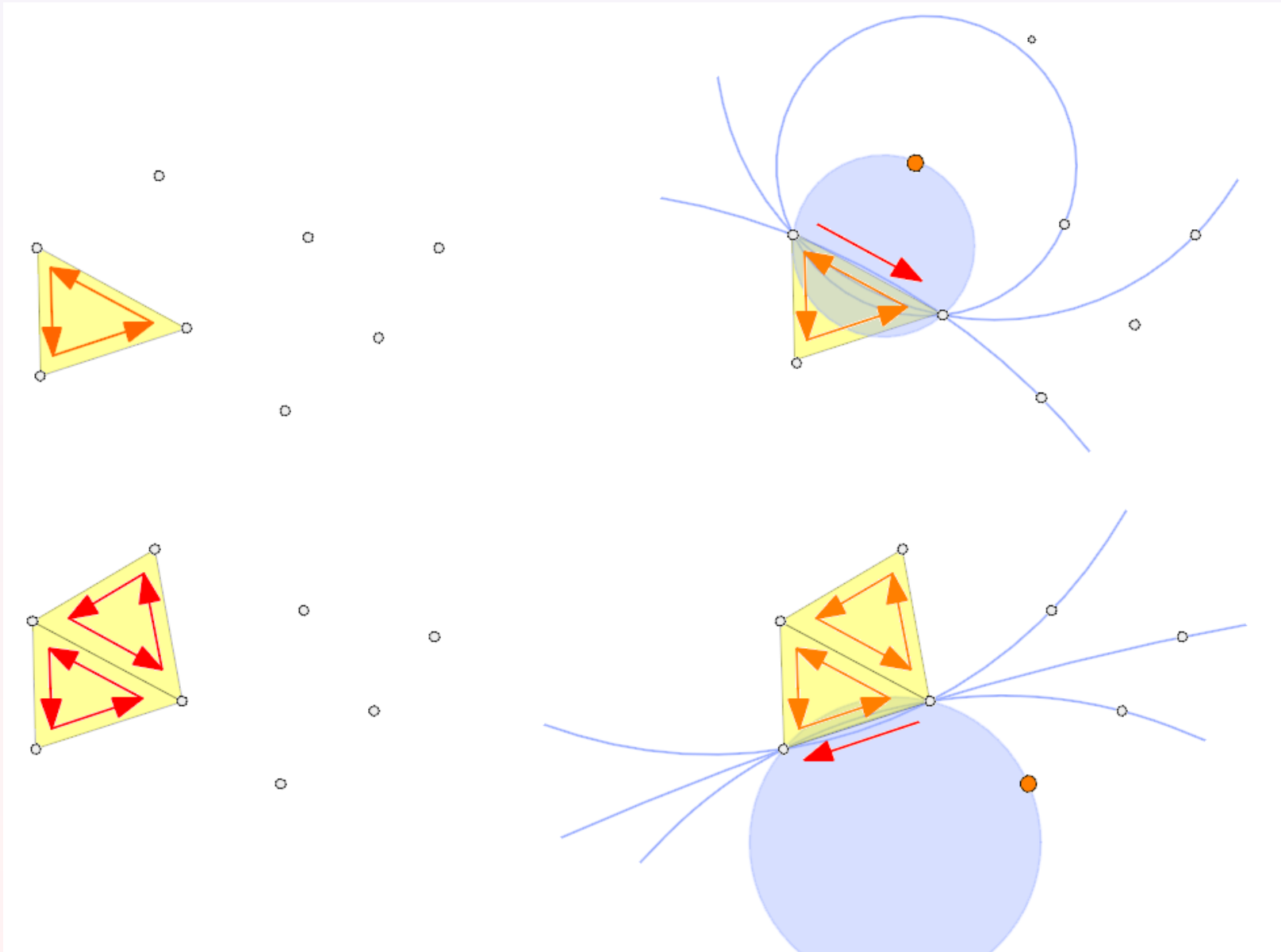
Při konstrukci používána modifikovaná datová struktura AEL (Active Edge List).
Obsahuje hrany e, ke kterým hledáme body $\underline{p}$, neukládá se topologický model.

Složitost je $O(n^2)$, lze vylepšit, algoritmus nestabilní.

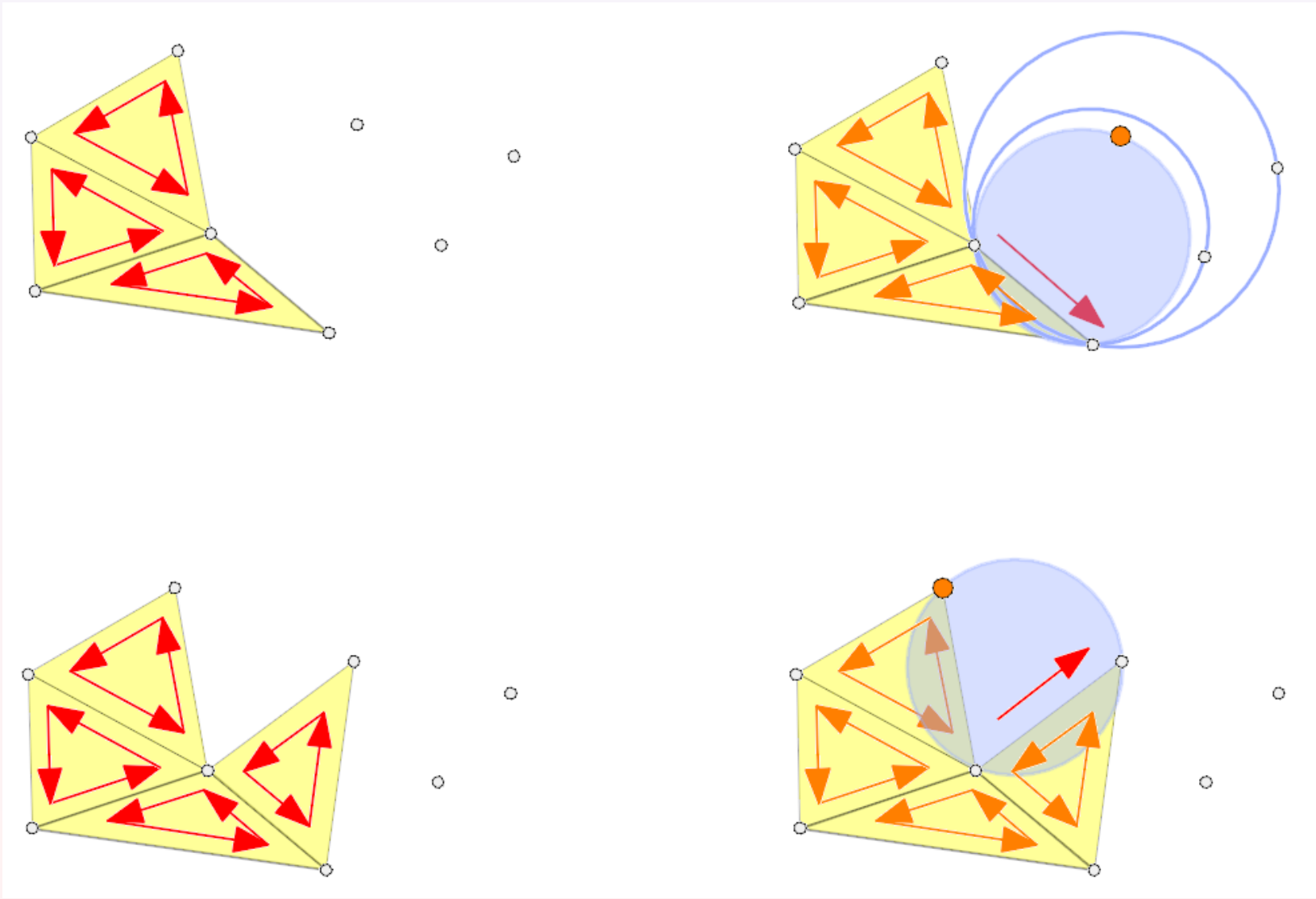
43. Ilustrace inkrementální konstrukce (1/3)



44. Ilustrace inkrementální konstrukce (2/3)



45. Ilustrace inkrementální konstrukce (3/3)



46. Algoritmus inkrementální konstrukce $\mathcal{DT}$ (1/2)

Algoritmus 2: Delaunay Triangulation Incremental (P, AEL, $\mathcal{DT}$)

1: $p_1 = rand(P)$, $\|p_2 - p_1\|_2 = min$. //Náhodný a nejbližší bod

2: Vytvoř hranu $e = (p_1 p_2)$

3: $\underline{p} = \arg\min_{\forall p_i \in \sigma_L(e)} r'(k_i)$, $k_i = (a, b, p_i)$, $e = (a, b)$

4: Pokud $\nexists \underline{p}$, prohoď orientaci $e \leftarrow (p_2 p_1)$. Jdi na 3).

5: $e_2 = (p_2, \underline{p})$, $e_3 = (\underline{p}, p_1)$ //Zbyvajici hrany trojuhelniku

6: AEL$\leftarrow e$, AEL$\leftarrow e_2$, AEL$\leftarrow e_3$ //Pridani 3 hran do AEL

7: $\mathcal{DT} \leftarrow e$, $\mathcal{DT} \leftarrow e_2$, $\mathcal{DT} \leftarrow e_3$ //Pridani 3 hran do DT

8: while AEL not empty:

9: $AEL \rightarrow e$, $e = (p_1 p_2)$ //Vezmi první hranu z AEL

10: $e = (p_2 p_1)$ //Prohod jeji orientaci

11: $\underline{p} = \arg\min_{\forall p_i \in L(e)} r'(k_i)$, $k_i = (a, b, p_i)$, $e = (a, b)$

12: if $\exists \underline{p}$: //Takovy bod existuje

13: $e_2 = (p_2, \underline{p})$, $e_3 = (\underline{p}, p_1)$ //Zbyvajici hrany trojuhelniku

14: $\mathcal{DT} \leftarrow e$ //Pridej hranu do $\mathcal{DT}$ ale ne do AEL

15: add(e_2,AEL,$\mathcal{DT}$), add(e_3,AEL,$\mathcal{DT}$) //Pridej do $\mathcal{DT}$ i do AEL(?)

47. Algoritmus inkrementální konstrukce $\mathcal{DT}$ (2/2)

Při přidání $e = (a, b)$ do AEL kontrola, zda neobsahuje hranu s opačnou orientací $e' = (b, a)$.
Pokud ano, je e' odstraněna z AEL.
Pokud ne, je e přidána do AEL.
Hrana e je v obou případech přidána do DT.

Triangulace ukládána po trojúhelnících.

Algoritmus 3: Add ($e = (a, b)$, AEL,$\mathcal{DT}$)

1: Vytvoř hranu $e' = (b, a)$
2: if $(e' \in AEL)$
3: $\quad AEL \rightarrow e'$ //Odstran z AEL
4: else:
5: $\quad AEL \leftarrow e$ //Pridej do AEL
6: $\mathcal{DT} \leftarrow (a, b)$. //Pridej do DT

V praxi není používán z důvodu kvadratické složitosti (hledání, mazání).
Výhodou je však poměrně jednoduchá implementace.
Pro rychlé hledání použita množina.

48. Metoda inkrementálního vkládání

Často používaná metoda konstrukce $\mathcal{DT}$.
Lze použít v $\mathbb{R}^2$ i $\mathbb{R}^3$.
Složitost je $O(n^2)$, po úpravách lze dosáhnout $O(n \cdot \log(n))$.
Klasický případ rekurzivní úlohy (fáze legalizace).

Princip algoritmu:
V každém kroku do $\mathcal{DT}$ přidán jeden bod a provedena legalizace $\mathcal{DT}$. Nechť S představuje podmnožinu datasetu P obsahující m bodů a p přidávaný bod

$$\mathcal{DT}_{m+1} = \mathcal{DT}_m \cap p.$$

Algoritmus tvořen 4 fázemi:

- Konstrukce simplexu Ω oblasti P (obalový trojúhelník).
- Přidání p do $\mathcal{DT}_m$.
- Legalizace triangulace $\mathcal{DT}_{m+1}$.
- Odstranění simplexových hran.

49. Fáze1: Konstrukce simplexu Ω

Žádný z bodů P neleží vně simplexu Ω s vrcholy p_{-1}, p_{-2}, p_{-3}.
$\mathcal{DT}$ bude probíhat nad sjednocením obou množin, tj. $\mathcal{DT}(P \cup \Omega)$.

Zaručíme tak, že přidání každého bodu p_i do $\mathcal{DT}_m$ proběhne v souladu s souladu s níže uvedenými pravidly.

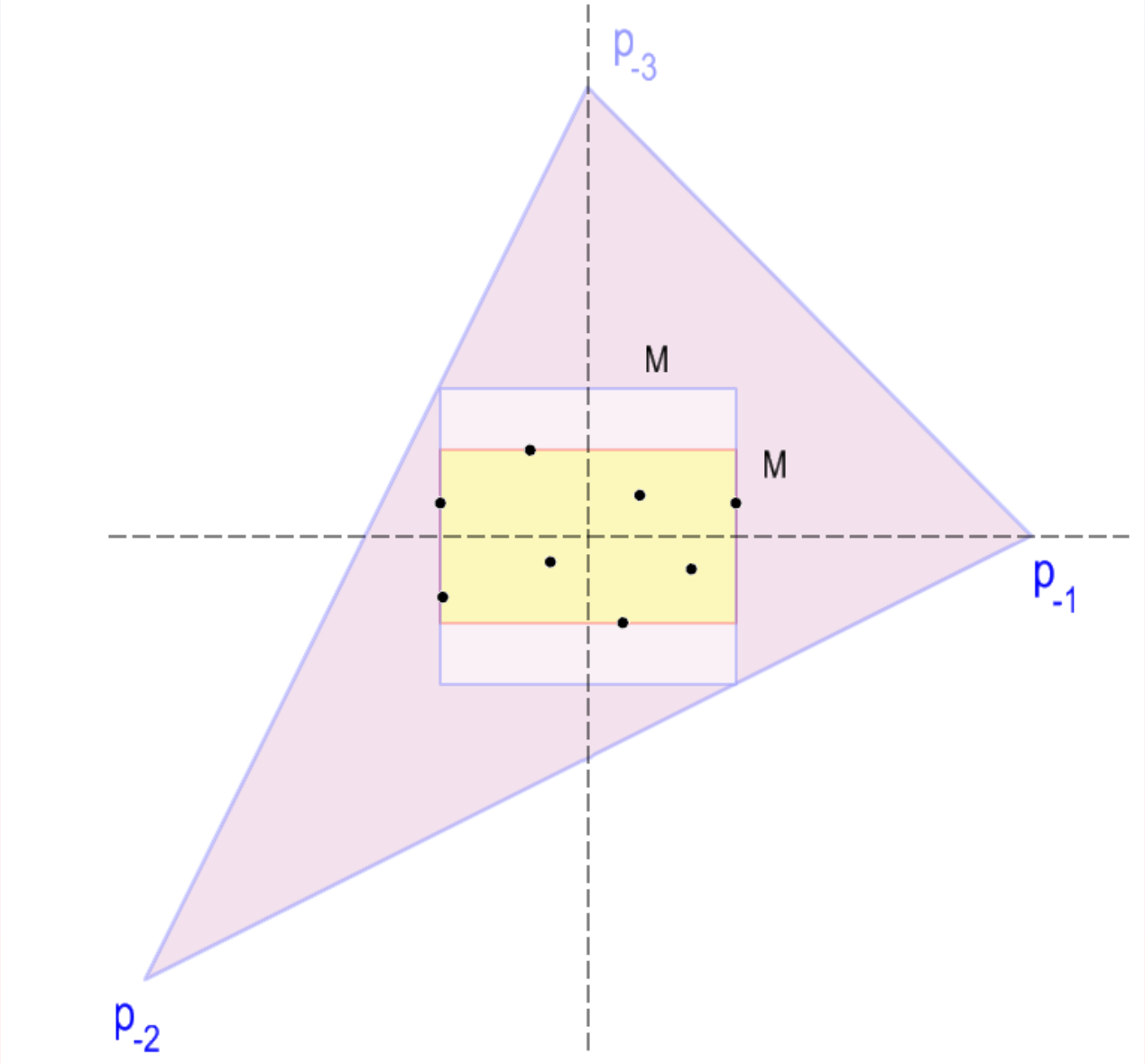
Vrcholy simplexu p_{-1}, p_{-2}, p_{-3} dostatečně daleko od P, aby neovlivňovaly trojúhelníky nad body P.
Souřadnice vrcholů simplexu Ω vcházejí z MBR zkonstruovaného nad P.

$$M = \max(x_{max} - x_{min}, y_{max} - y_{min})$$

Pak $p_{-1} = [-3M, 0]$, $p_{-2} = [0, 3M]$, $p_{-31} = [-3M, -3M]$.

Při legalizaci nutné upravit simplexové trojúhelníky (tj. takové, jejichž alespoň jeden vrchol $p_i \in \Omega$.

50. Ilustrace simplexu Ω



51. Fáze 2: Přidání p do $\mathcal{DT}$

Nalezení trojúhelníku/trojúhelníků t_i, se kterými p inciduje.
Kritická pasáž algoritmu, výpočetně nejnáročnější krok.
Nelze prohledávat všechny trojúhelníky.
Množství procházených trojúhelníků nutno minimalizovat.

Vyhledání incidujícího trohúhelníku:
Dvě strategie:

- Metoda procházky (heuristika, $O(\sqrt{n})$), Point Location Problem.
- DAG Tree (konstrukce ternárního stromu, efektivnější, $O(\log_2 n)$).

Metody procházek:
Startovní bod z, analyzovaný bod q.
Hledáme $t_i, q \in t_i$.

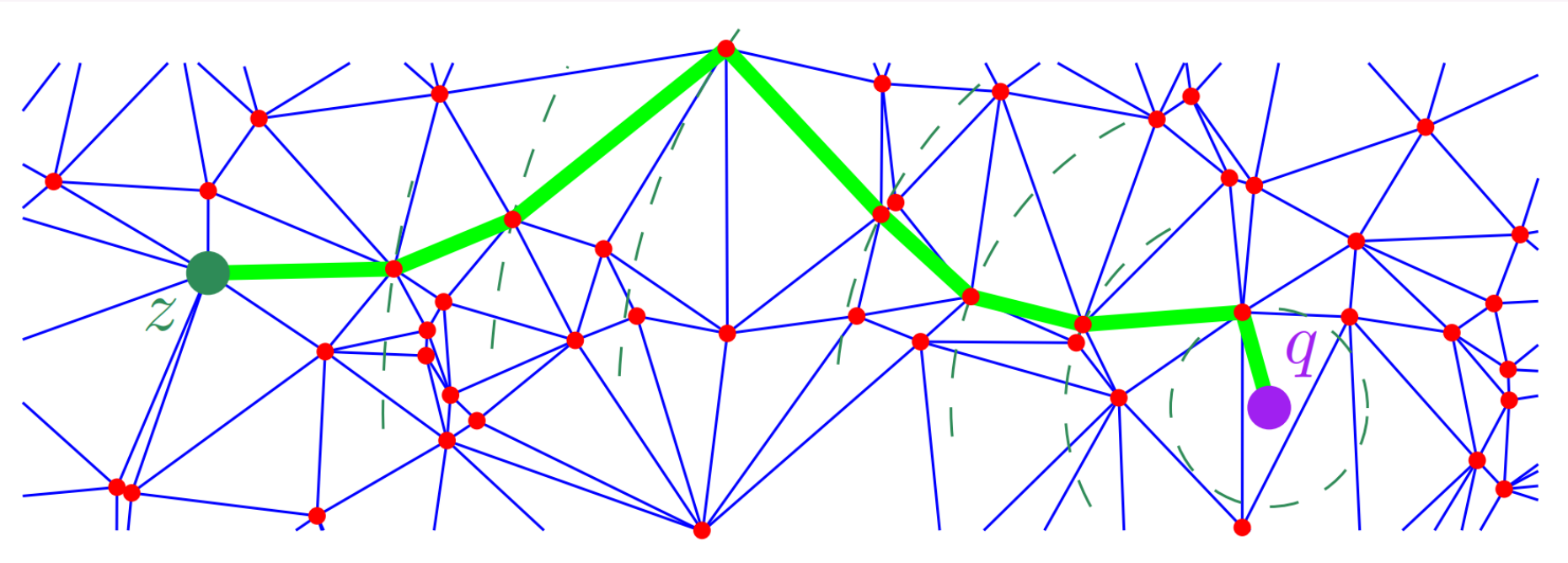
- Greedy Walk: přes nejbližší vrcholy.
- Straight Walk: přes t protnuté (q, z),
- Visibility Walk: díváme se vlevo/vpravo.
- Orthogonal Walk: 2 ortogonální směry.

52. Greedy Walk

Nejjednodušší procházkový algoritmus.

Procházka přes sousedící trojúhelníky.

V každém okamžiku zvolen bod nejbližší od q.



(Devillers, 2016).

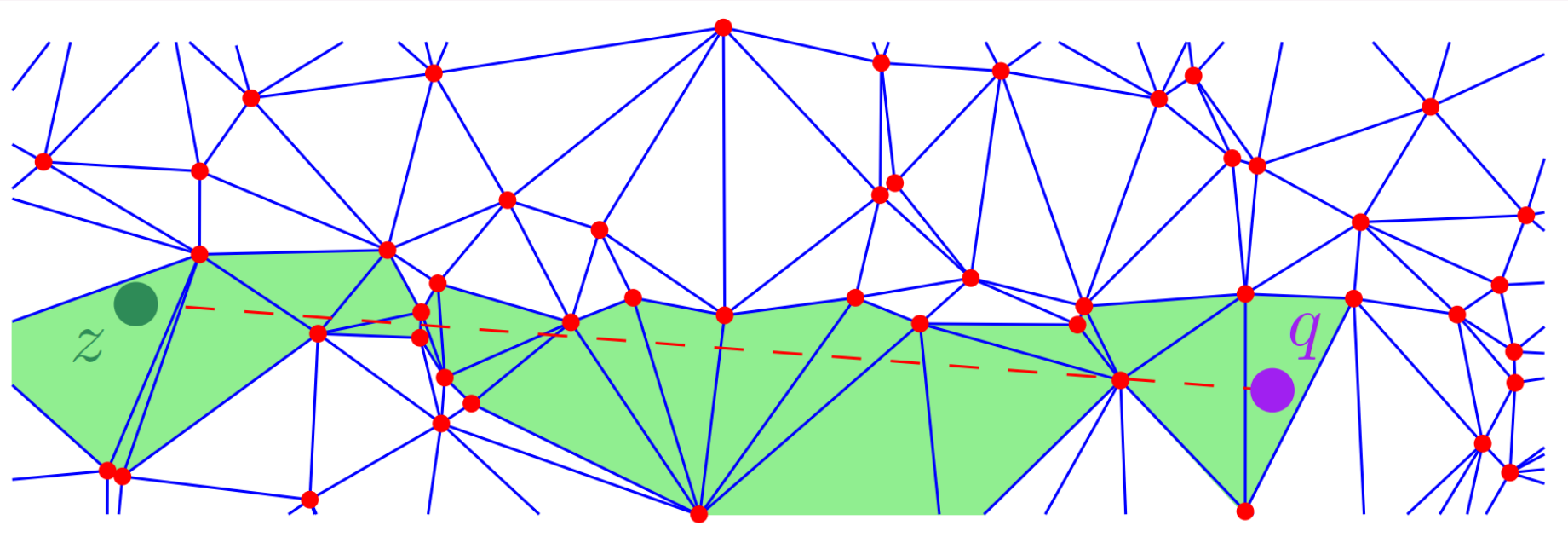
53. Straight Walk

Postupná cesta přes sousedící trojúhelníky protnuté (q, z).

V každém trojúhelníku hledána protnutá strana.

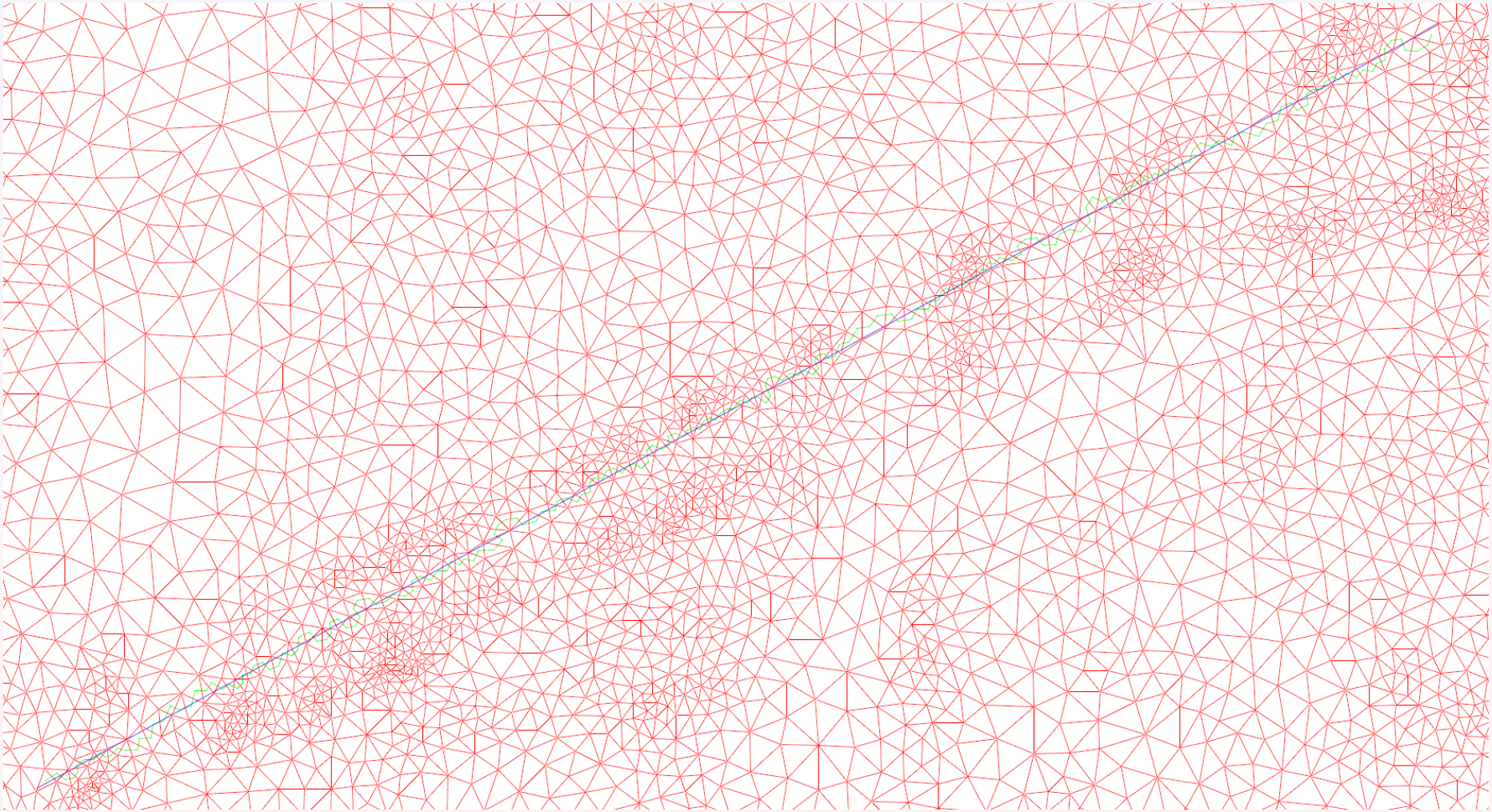
Alternativně orientační test.

Pozor na singularity: $p_i \in (q, z)$.



(Devillers, 2016).

Straight Walk



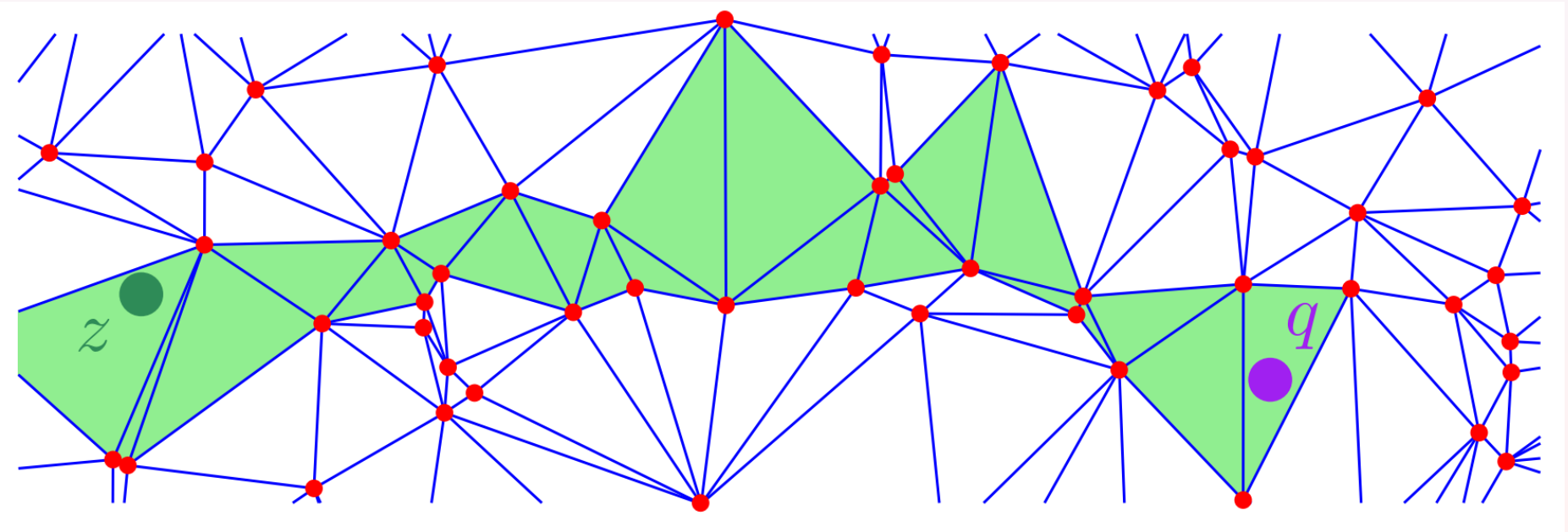
54. Visibility Walk

Postupná cesta přes sousedící trojúhelníky.

Nejefektivnější procházka, Lawson (1977).

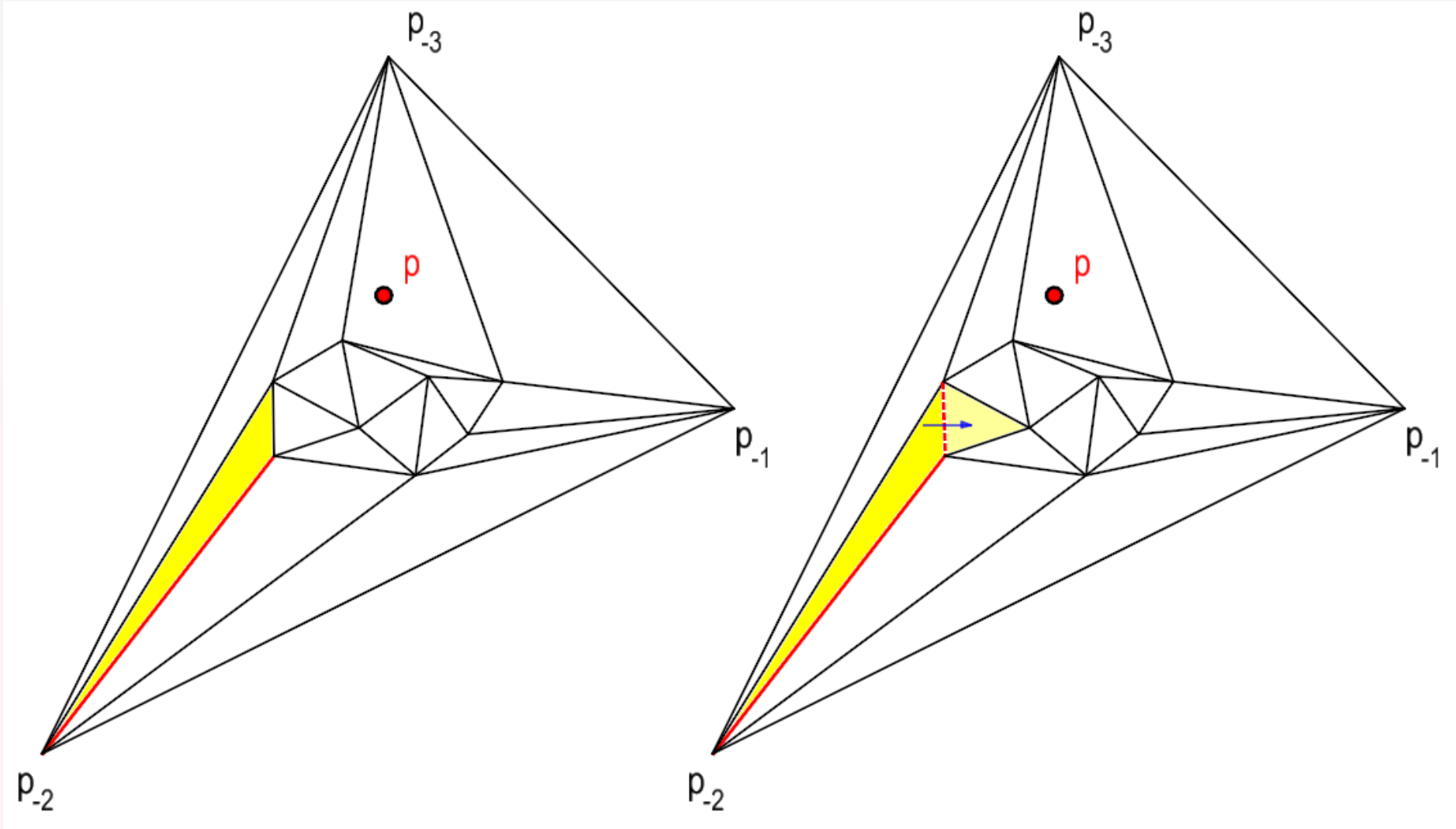
Z t_i přecházíme na t_j, pokud t_i a q v opačné polorovině vzhledem k hraně e_{ij}

$$e_{i,j} = \begin{cases} e_{i+1,j}, & q \epsilon \sigma_l(e_{ij}), \\ e_{j,i}, & q \epsilon \sigma_r(e_{ij}). \end{cases}$$

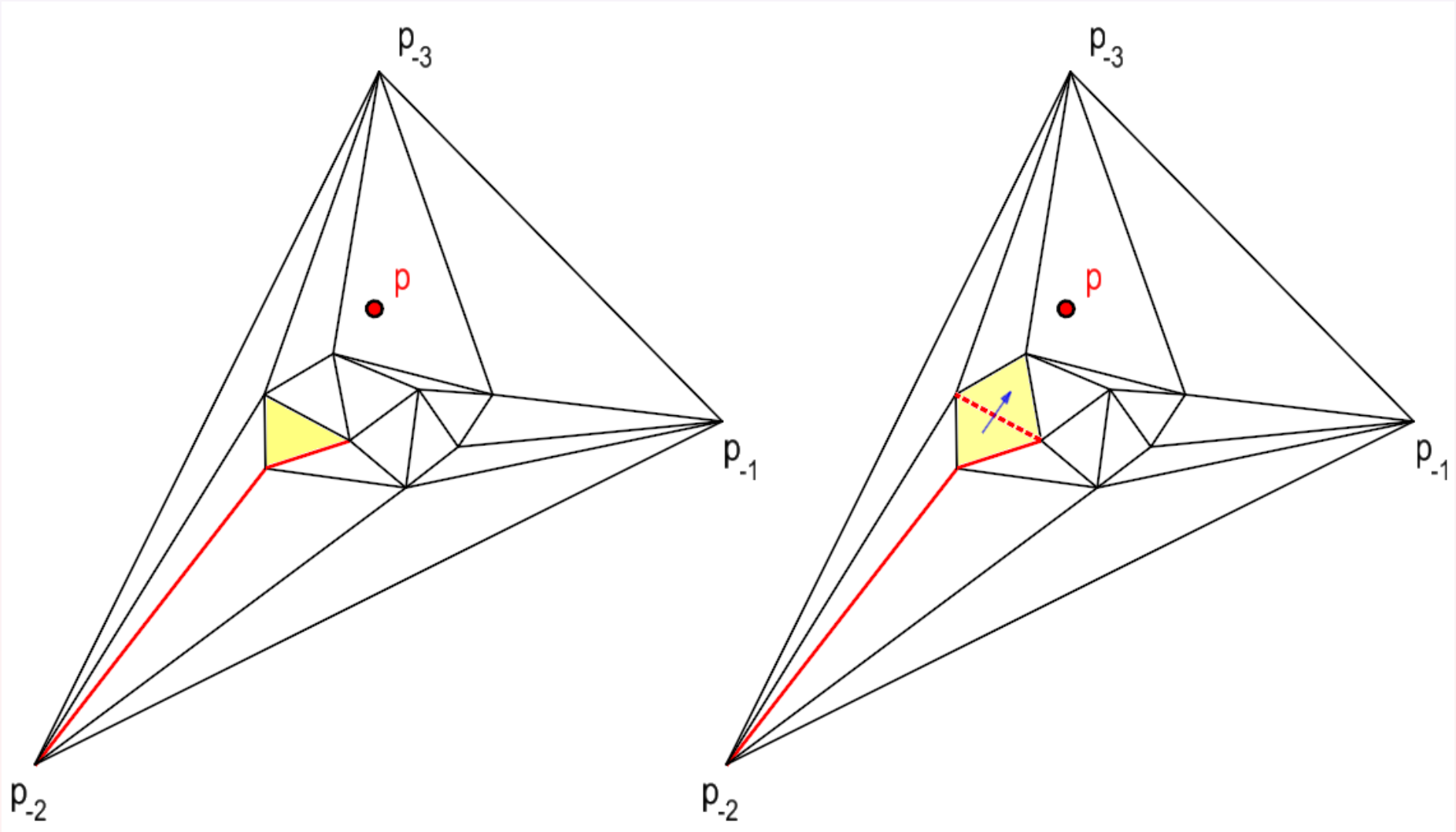


(Devillers, 2016).

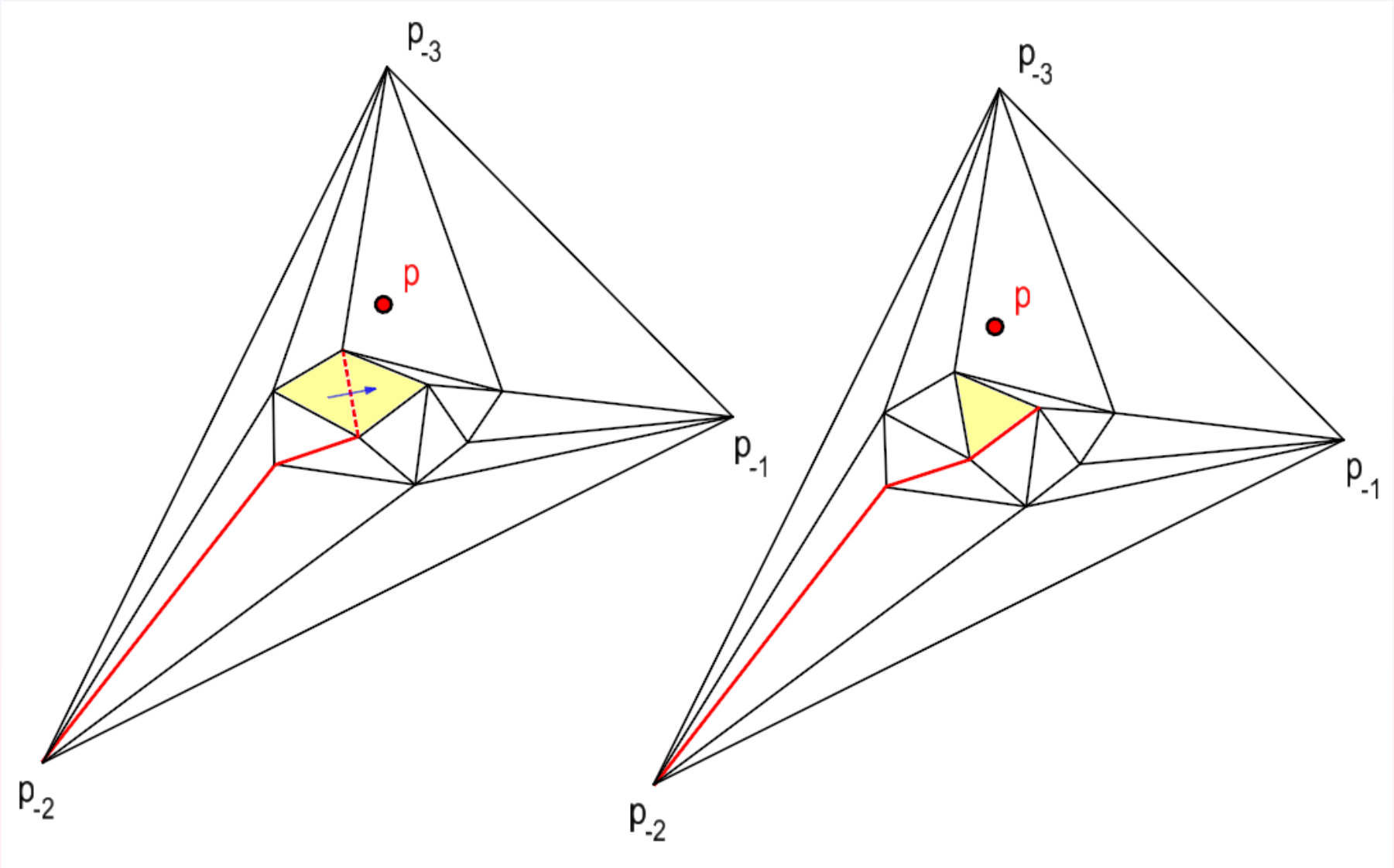
55. Ukázka Visibility Walk (1/6)



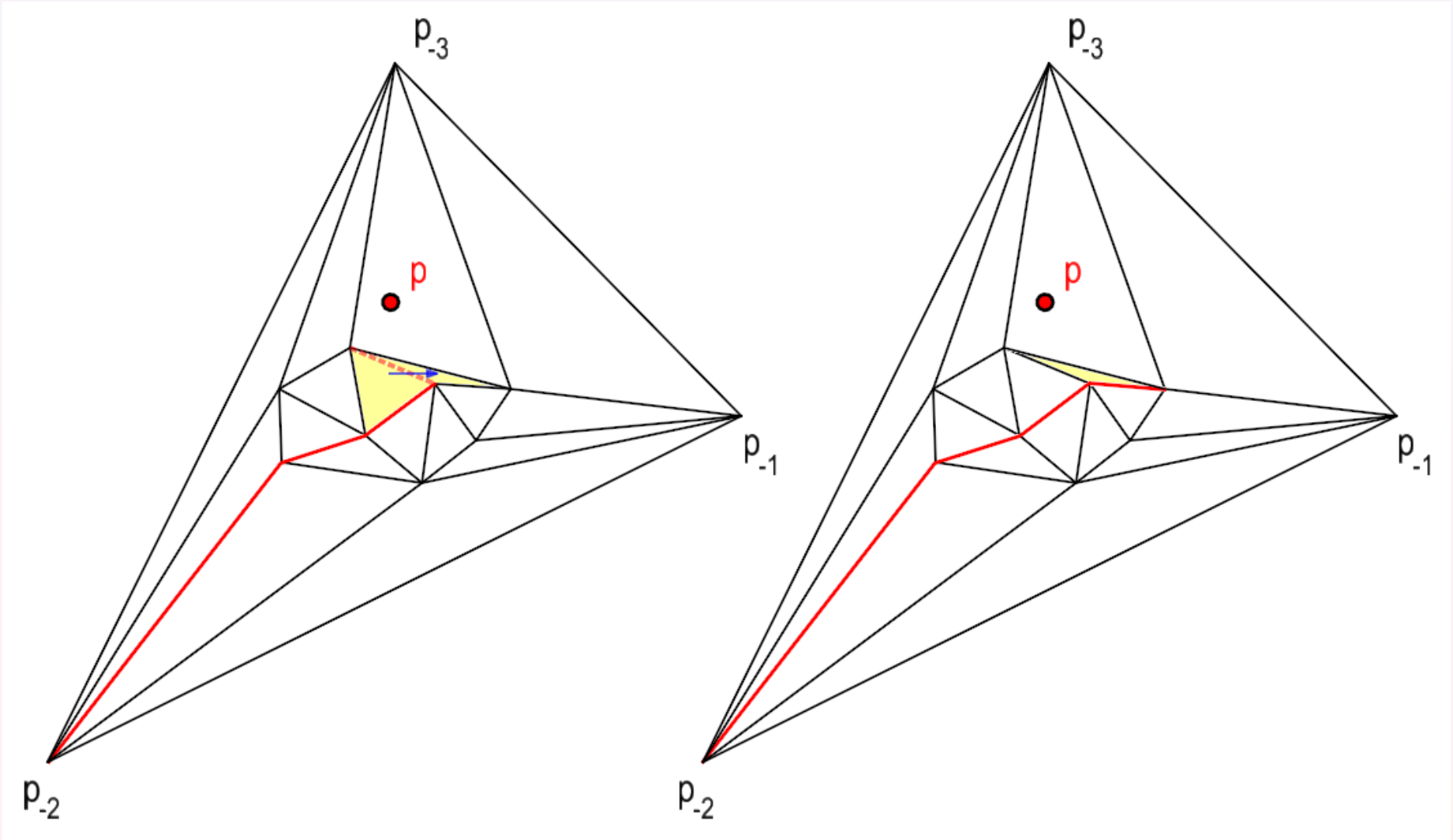
56. Ukázka Visibility Walk (2/6)



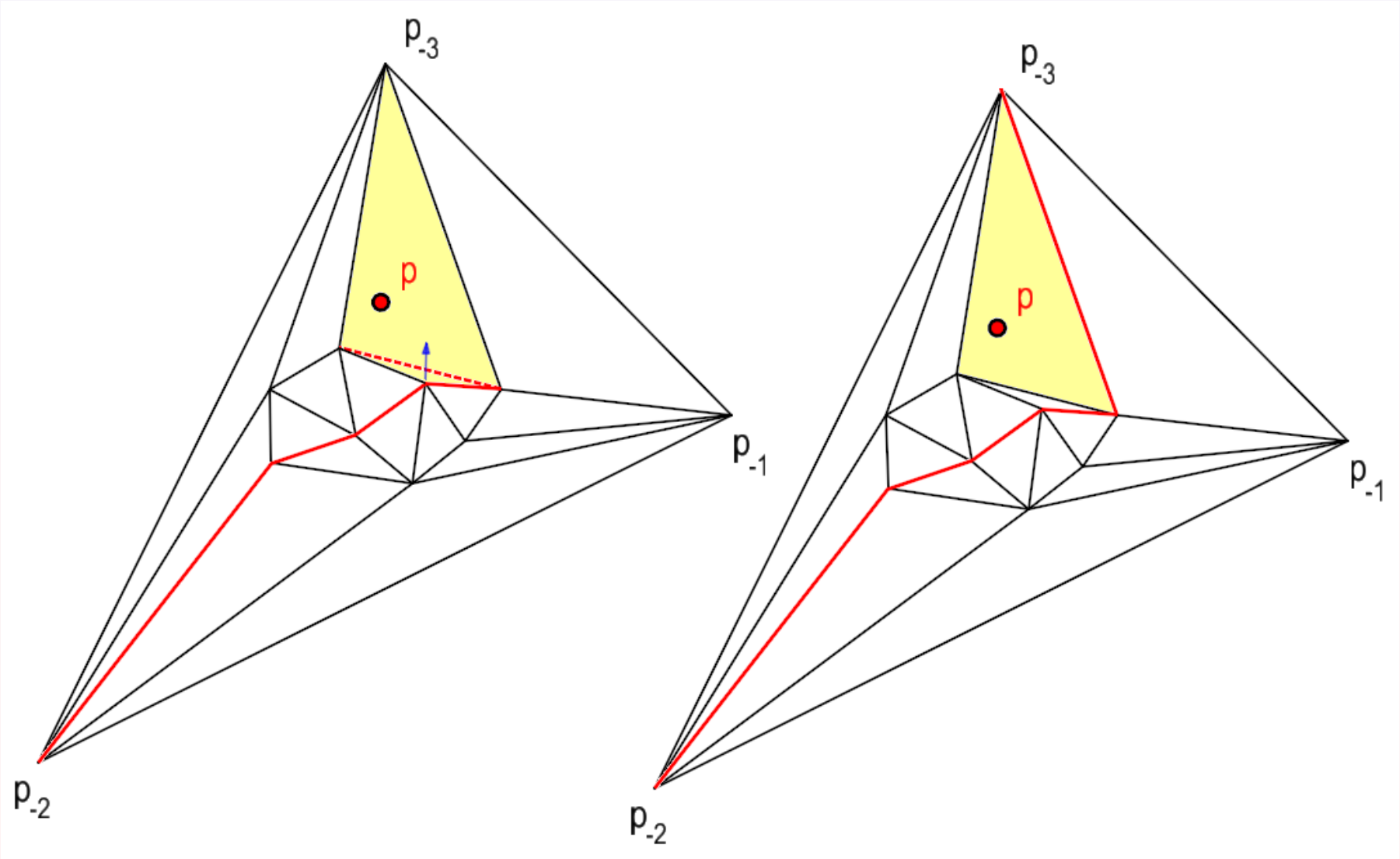
57. Ukázka Visibility Walk (3/6)



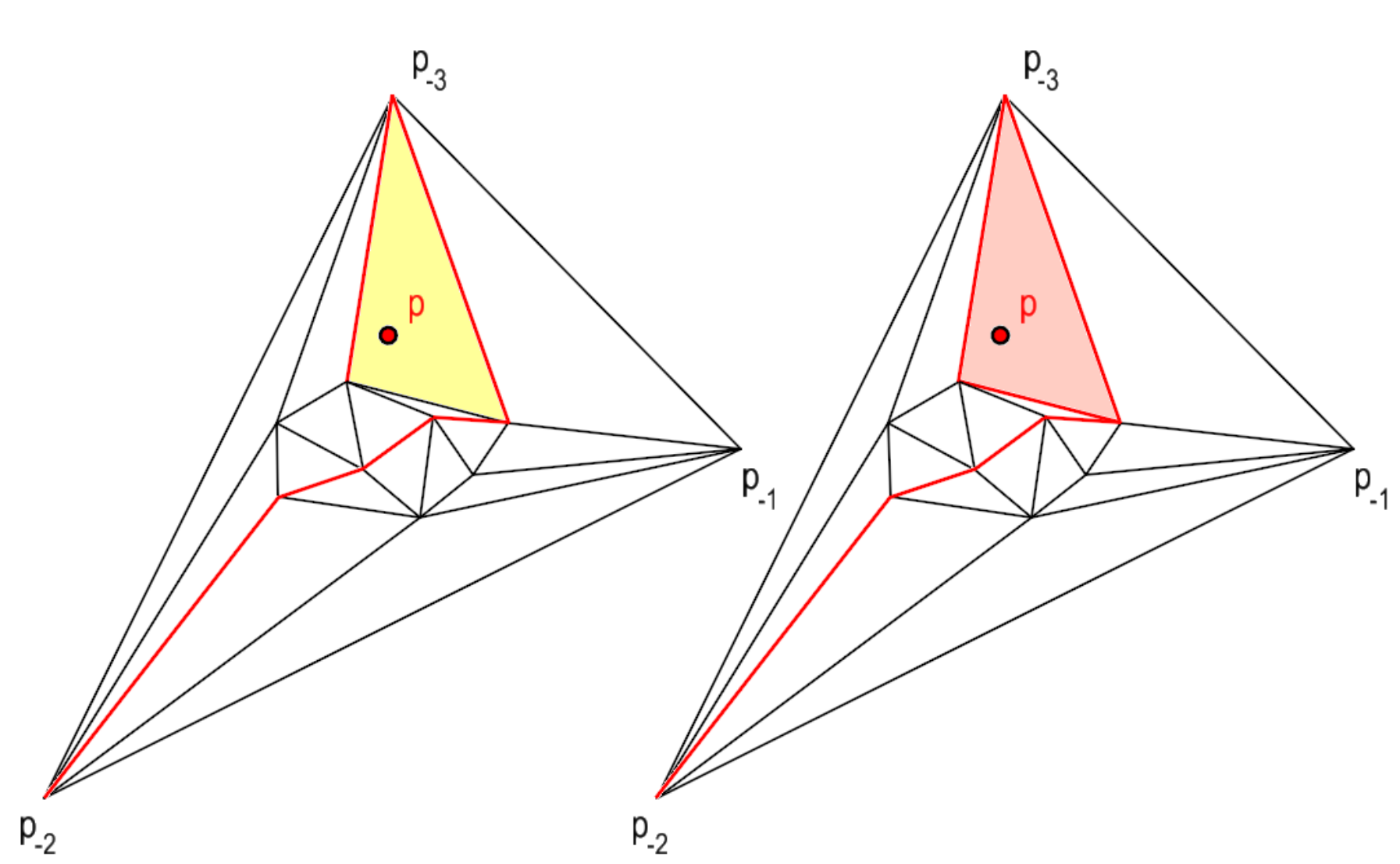
58. Ukázka Visibility Walk (4/6)



59. Ukázka Visibility Walk (5/6)



60. Ukázka Visibility Walk (6/6)



61. Visibility Walk, algoritmus

Složitost $O(\sqrt{n})$.
Pozor na float aritmetiku: pravidelné rastry.

Algoritmus 3: VW (p,τ,)

1: $t = t_0$, found = false
2: while (!found)
3: found = true;
4: for $\forall e(a,b) \in \tau$ //Hrana $e_{ij} = t \cap t_{inc}$
5: if $p \in \sigma_r(a,b)$
6: $t \leftarrow t_{inc}$ //Incidující trojúhelník
7: found = false
8: break
9: return t

---

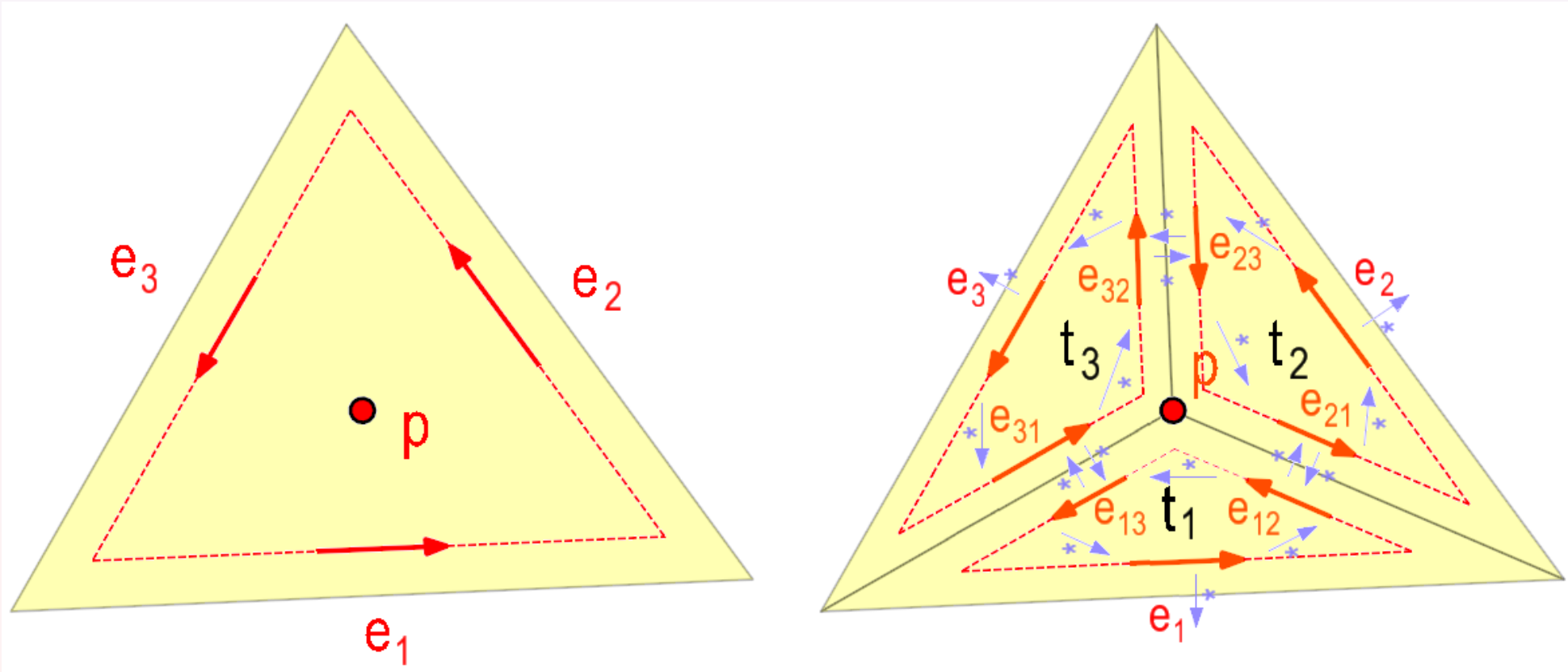
62. Vztah bodu p a nalezeného t_i

Existují tři různé varianty vzájemné polohy přidávaného bodu p a nalezeného trojúhelníku t_i:

- *Bod p leží ve vrcholu t_i*
  Bod neovlivní již vytvořenou triangulaci $\mathcal{DT}_m$, bude zanedbán.
  Triangulace ponechána beze změny, $\mathcal{DT}_{m+1} = \mathcal{DT}_m$.
- *Bod $p \in t_i$*
  Zkonstruovány tři nové hrany spojující p s vrcholy t_i.
  Trojúhelník t se rozpadne na tři nové trojúhelníky se společným vrcholem.
- *Bod p leží ve straně t_i, t_j*
  Oba incidující trojúhelníky t_i, t_j, v jejichž společné hraně přidávaný bod leží, rozděleny dvojicí úseček jdoucích z p do protilehlých vrcholů t_i, t_j.
  Vzniknou čtyři nové trojúhelníky se společným vrcholem.

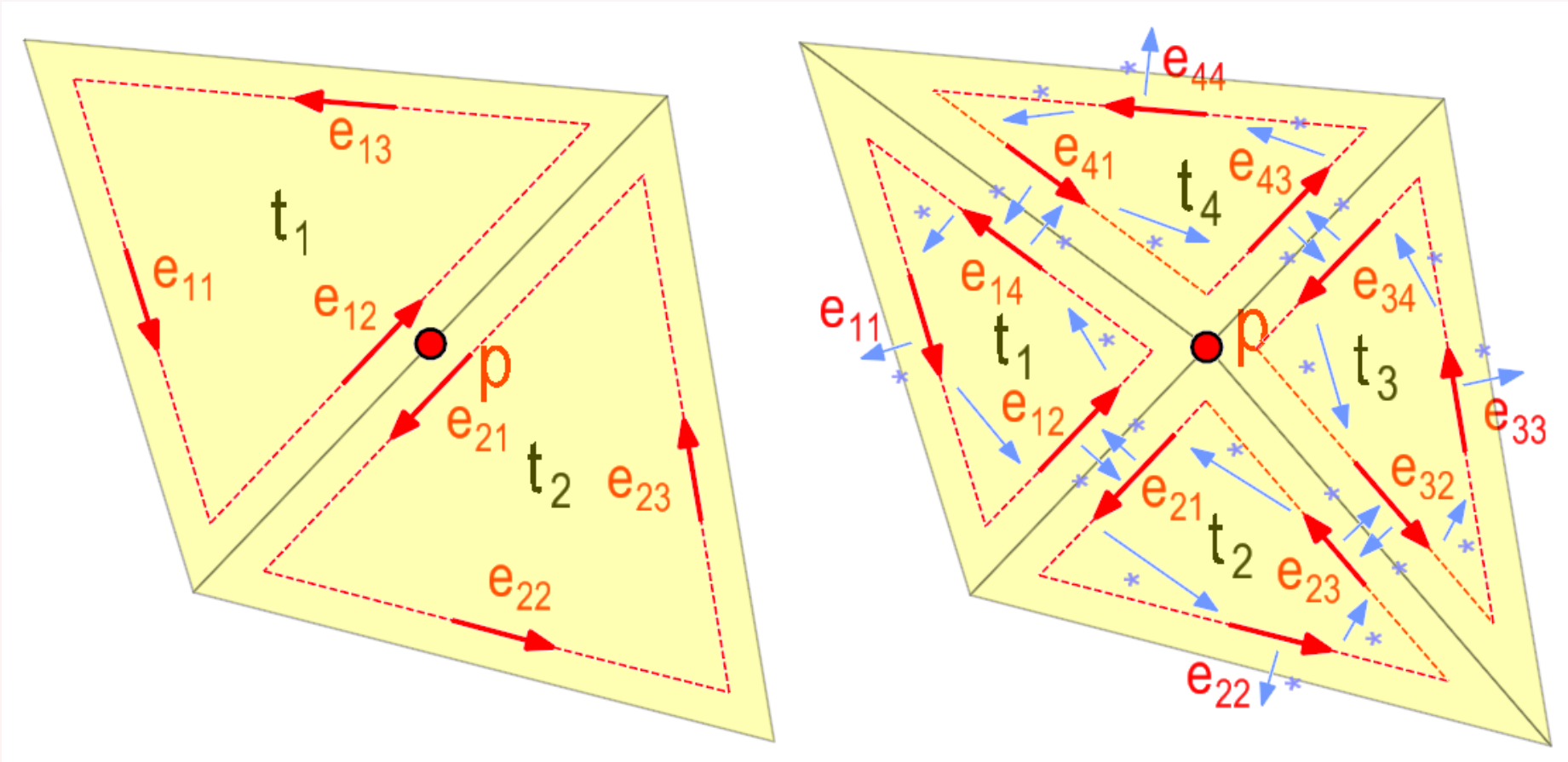
63. Bod $p \in t_i$

Vytvoření nových hran: $e_{12}, e_{13}, e_{21}, e_{23}, e_{31}, e_{32}$.
Vytvoření topologie: provázení všech hran s použitím pointerů.



64. Bod p leží ve straně t_i, t_j

Změna koncových bodů hran: e_{12}, e_{21}, e_{23}.
Zrušení hrany e_{13}.
Vytvoření nových hran: e_{14}, e_{32}, e_{33}, e_{34}, e_{41}, e_{43}, e_{44}.
Vytvoření topologie: provázení všech hran s použitím pointerů.



65. Fáze 3: Legalizace nově vytvořené triangulace

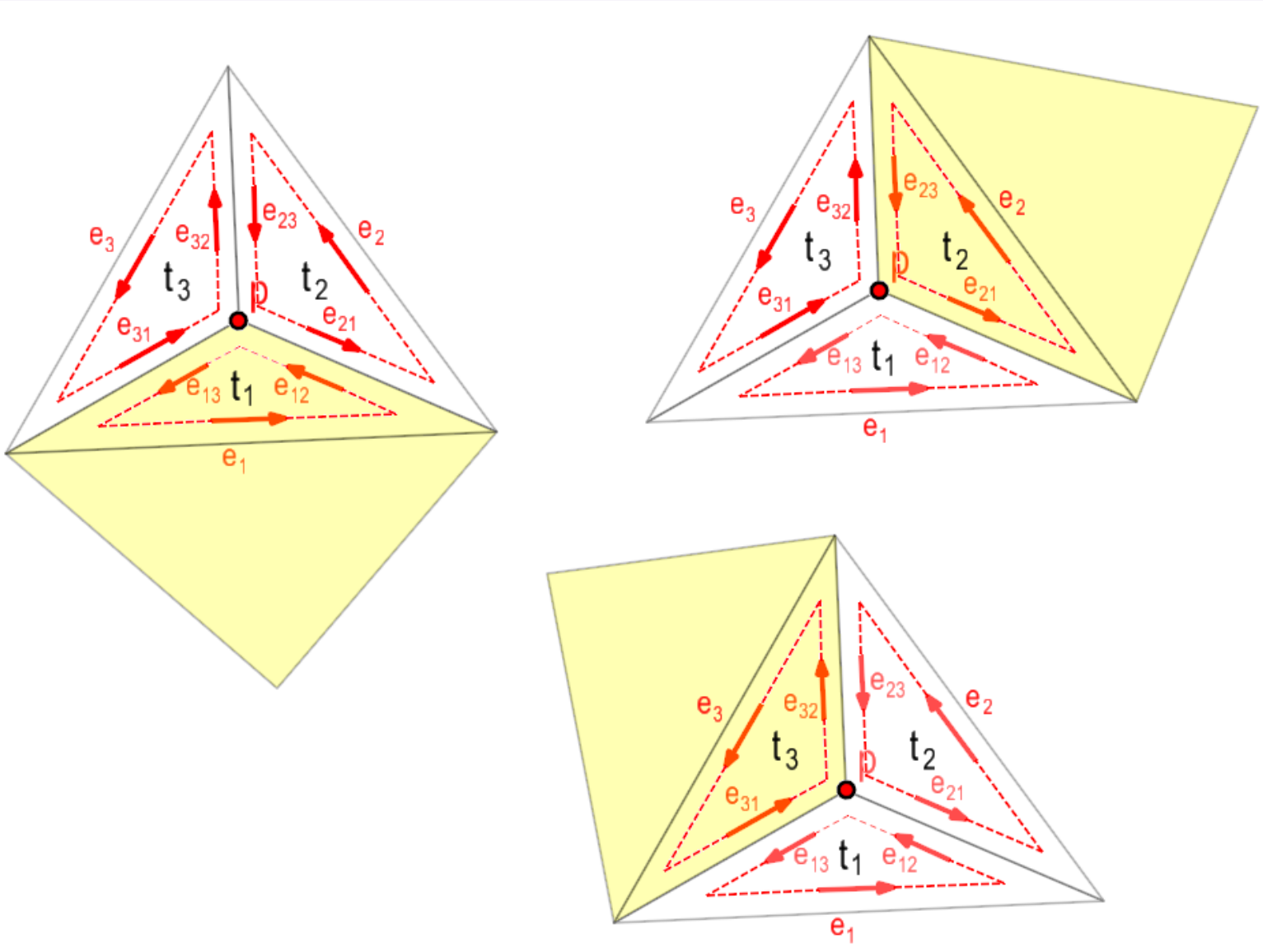
Takto vzniklá triangulace nemusí být delaunayovská, triangulaci je proto nutno legalizovat.

- $p \in t_i$
  Pokud přidávaný bod $p \in t_i$, legalizujeme nově vzniklé trojúhelníky t_1, t_2, t_3 s incidujícími trojúhelníky $\mathcal{DT}$ (celkem 3 legalizace).
- *Bod p leží ve straně* t_i, t_j
  Pokud přidávaný bod *p leží ve straně* t_i, legalizujeme nově vzniklé trojúhelníky t_1, t_2, t_3, t_4 s incidujícími trojúhelníky $\mathcal{DT}$ (celkem 4 legalizace).

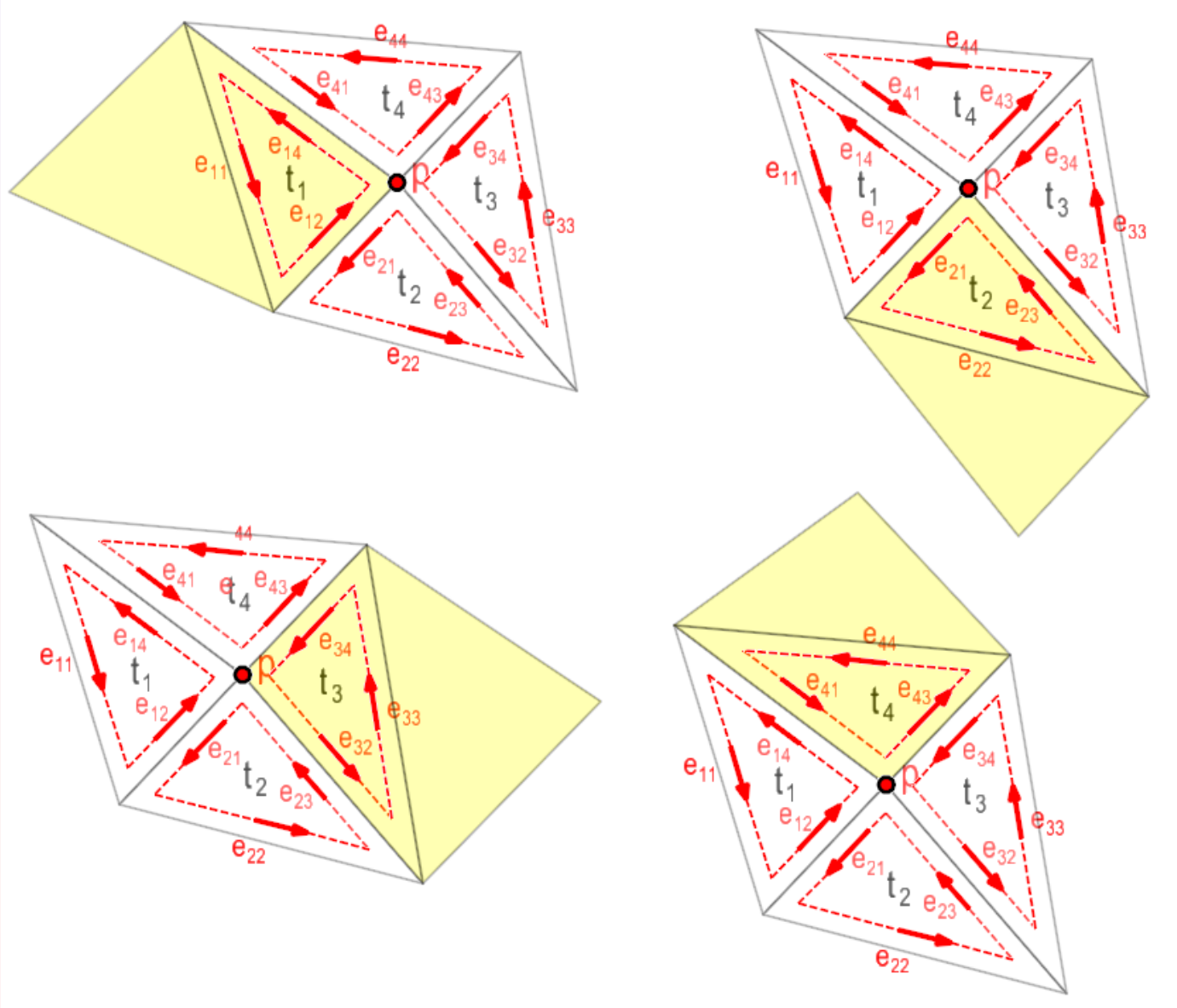
Tím proces legalizace nekončí, přehození úhlopříčky v některém z výše uvedených případů vyvolá potřebu legalizace k jejich incidujícím trojúhelníkům. Pokud alespoň jeden z vrcholů trojúhelníka představuje bod Ω, nutno *upravit legalizační pravidla*.

Tento krok je vyvolává potřebu rekurzivního řešení problému.
V nepříznivém případě může vložený bod p způsobit přegenerování značného množství $t_i \in \mathcal{DT}$

66. Ilustrace procesu legalizace, $p \in t_i$



67. Ilustrace procesu legalizace, *p leží ve straně* t_i, t_j



68. Pravidla legalizace pro vrcholy Ω (1/2)

Jsou reakcí na situaci, že do $\mathcal{DT}$ je zahrnut i simplexový trojúhelník Ω.

Aby nedošlo k nevhodnému ovlivnění oblasti P oblastí Ω, mohou být do $\mathcal{DT}(P \cup \Omega)$ přidány i nelegální hrany.
Hrany nelegální v $\mathcal{DT}(P \cup \Omega)$ však budou legální $\mathcal{DT}(P)$.
Jedná se o hrany ležící na $\mathcal{H}(P)$.

Nechť p_i, p_j, p_k a p_i, p_j, p_l představují dva incidující troúhelníky a p_i, p_j představuje testovanou hranu.

Nutno uvažovat následující případy:

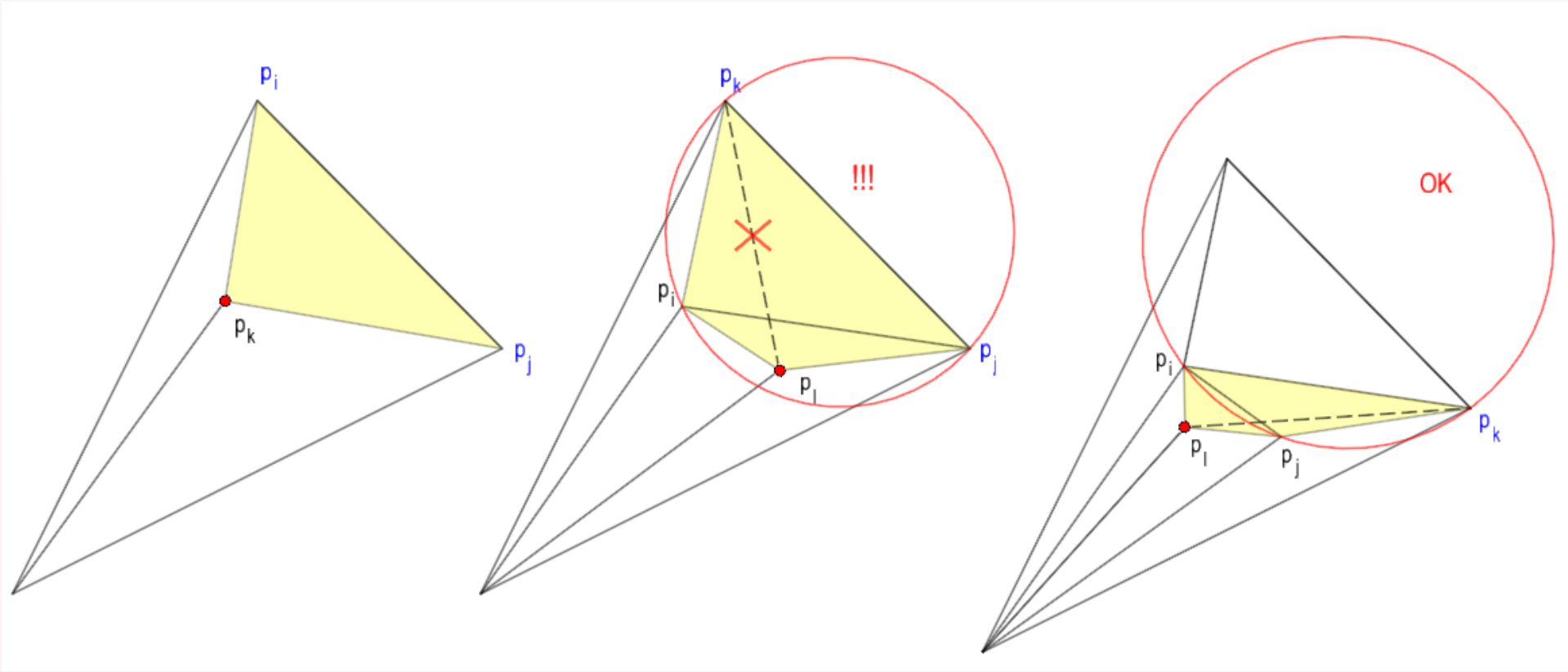
- *indexy i, j jsou záporné:*
  Hrana $\overline{p_i, p_j}$ je legální.
- *všechny indexy $i, j, k, l > 0$:*
  Normální případ, prováděno běžné testování.

69. Pravidla legalizace pro vrcholy Ω (1/2)

- *jeden z indexů i, j, k, l záporný:*
  Pokud jeden z bodů $p_i, p_j \in \Omega$, pak je strana $\overline{p_i, p_j}$ je nahrazena $\overline{p_k, p_l}$, v opačném případě je ponechána
  Výsledná hrana nemusí být legální vzhledem k $\mathcal{DT}(P \cup \Omega)$, leží na $\mathcal{H}(P)$, leží na $\mathcal{H}(P)$.
- *dva z indexů i, j, k, l jsou záporné*
  Jeden z indexů i, j a jeden z indexů k, l záporný.
  Pokud je negativní index i, j menší (v abs. hodnotě) než negativní index k, l , je $\overline{p_i, p_j}$ v pořádku;
  V opačném případě je $\overline{p_i, p_j}$ nahrazena $\overline{p_k, p_l}$.
  Výsledná hrana nemusí být legální vzhledem k $\mathcal{DT}(P \cup \Omega)$, leží na $\mathcal{H}(P)$.
- *tři z indexů i, j, k, l jsou záporné*
  Situace nemůže nastat.

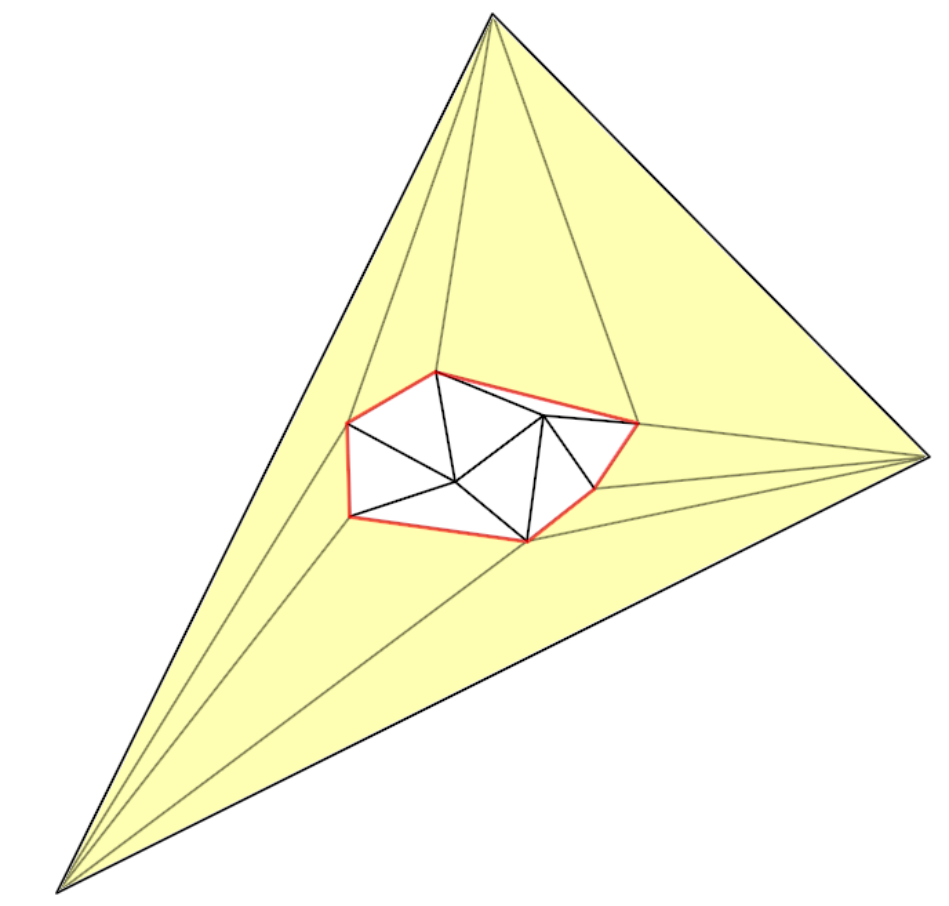
70. Ilustrace legalizačních pravidel pro vrcholy Ω

Vlevo indexy i, j záporné. Uprostřed 2 *z indexů* i, j, k, l záporné, swap nelegální vzhledem $\mathcal{DT}(P \cup \Omega)$. *Vpravo* dva z indexů i, j, k, l záporné, swap není třeba provádět (bod $\Omega \in k$ nebrán v potaz).



71. Fáze 4: Odstranění simplexových hran

Odstranění všech stran $\mathcal{DT}(P \cup \Omega)$ které incidují z Ω, výsledkem $\mathcal{DT}(P)$.
Výsledkem oříznutí na konvexní obálku.



72. Implementace algoritmu inkrementálního vkládání (1/2)

Algoritmus 3: DT Incremental Inserton (P ,$\mathcal{DT}$)

1: Vytvoření simplexu: do $\mathcal{DT} \leftarrow t(p_{-1}, p_{-2}, p_{-3})$
2: Opakuj pro $i \in 1, ..., n$:
3: Přidej p do $\mathcal{DT}$.
4: Najdi $t(p_i, p_j, p_k)$ takový, že $p \in t$.
5: Jestliže $\exists p \in t(p_i, p_j, p_k)$:
6: $\mathcal{DT} \leftarrow t(p, p_i, p_j)$ //Pridani trojuhelniku
7: $\mathcal{DT} \leftarrow t(p, p_j, p_k)$ //Pridani trojuhelniku
8: $\mathcal{DT} \leftarrow t(p, p_k, p_l)$ //Pridani trojuhelniku
9: Legalizace hrany $(p_i, p_j) \in t(p, p_i, p_j)$.
10: Legalizace hrany $(p_j, p_k) \in t(p, p_j, p_k)$.
11: Legalizace hrany $(p_k, p_l) \in t(p, p_k, p_l)$.

73. Implementace algoritmu inkrementálního vkládání (2/2)

Algoritmus 3: DT Incremental Inserton ($\overline{a,b}$, AEL,$\mathcal{DT}$)

12: Jinak jestliže $\exists p \in \partial t_1(p_i, p_j, p_k) \wedge \in \partial t_2(p_i, p_l, p_j)$:
13: $\mathcal{DT} \leftarrow t(p, p_k, p_i)$ //Pridani trojuhelniku
14: $\mathcal{DT} \leftarrow t(p, p_j, p_k)$ //Pridani trojuhelniku
15: $\mathcal{DT} \leftarrow t(p_i, p_l, p)$ //Pridani trojuhelniku
16: $\mathcal{DT} \leftarrow t(p, p_l, p_j)$ //Pridani trojuhelniku
17: Legalizace hrany $(p_i, p_l) \in t(p, p_i, p_l)$.
18: Legalizace hrany $(p_l, p_j) \in t(p, p_l, p_j)$.
19: Legalizace hrany $(p_j, p_k) \in t(p, p_j, p_k)$.
20: Legalizace hrany $(p_k, p_i) \in t(p, p_k, p_i)$.
21: Odstranění simplexových hran z $\mathcal{DT}$.

74. Algoritmus legalizace hrany

Přidávaný bod označen jako p.

Strana (p_i, p_j) představuje stranu v trojúhelníku t, která má být prohozena.

Incidující trojúhelník t' tvořen hranami p_i, p_j, p_l.

Rekurzivní procedura, legalizace volána současně na dvě nové strany.

Algoritmus 4: DT Legalizace hrany $((p_i, p_j), t(p_i, p_j, p_k))$

1: Najdi trojúhelník $t'(p_i, p_l, p_j)$ incidující s hranou (p_i, p_j) trojúhelníku t.
2: if (p_i, p_j) nelegální
3: Prohoď (p_i, p_j) za (p_k, p_l)
4: Legalizace hrany $(p_i, p_k), t(p_i, p_l, p_k)$.
5: Legalizace hrany $(p_j, p_k), t(p_j, p_k, p_l)$.

---

75. Triangulace se vstupní podmínkou

Označovány jako Constrained Triangulations (tj. triangulace s omezením).
Do triangulace zavedeny povinné hrany spojující definované body p.
Poloha povinných hran se při triangulaci již nesmí měnit.

Povinné hrany při konstrukci triangulace kříží jiné možné hrany, které jsou vzhledem k nějakému kritériu lokálně optimální (a pro triangulaci vhodnější), avšak tyto hrany nejsou použity.

Triangulace se vstupní podmínkou proto nejsou lokálně optimální.

Geometrický předpoklad: povinné hrany se nesmějí protínat.

Široké použití v kartografii tvorbě digitálních modelů terénu, povinné hrany umožňují lepší modelování morfologie terénu.

Zástupci:

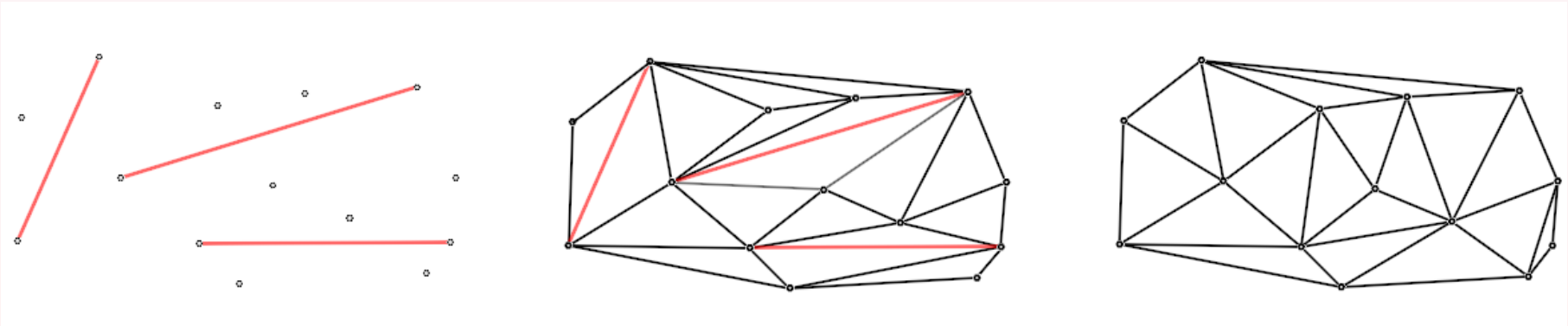
- Greedy triangulace se vstupní podmínkou (Constrained Greedy Triangulation).
- Delaunay triangulace se vstupní podmínkou (Constrained Delaunay Triangulation).

76. Greedy triangulace se vstupní podmínkou

Greedy triangulace se vstupní podmínkou $\mathcal{CGT}(P)$ není příliš často používána.

Tvorba $\mathcal{CGT}(P)$ probíhá ve dvou krocích:

- *Přidání povinných hran*
  Do prázdné triangulace přidány povinné hrany.
  Žádná z povinných hran nemůže být při triangulaci nahrazena možnou kratší stranou.
- *Tvorba* $\mathcal{GT}(P)$
  Konstrukce Greedy triangulace nad P.



77. Delaunay triangulace se vstupní podmínkou

Nejpoužívanější triangulace v geoinformatice.
Na rozdíl od $\mathcal{CGT}(P)$ nutná redefinice triangulace: přes povinné hrany neprobíhá swapování.

Triangulace jako celek nemaximalizuje minimální úhel v trojúhelnících.
Zavedení povinných hran sníží rychlost triangulačního algoritmu.

Možné geometrické problémy: povinná hrana kolineární s nějakou hranou
$\mathcal{DT}(P) \Rightarrow$rozdělení povinné hrany na 3 části, povinná hrana protíná bod
$\mathcal{DT}(P) \Rightarrow$rozdělení povinné hrany na 2 části.

Konstrukce probíhá ve třech krocích:

- Vytvoření $\mathcal{DT}(P)$.
- Zadání povinných hran do $\mathcal{DT}(P)$.
- Převod $\mathcal{DT}(P)$ na $\mathcal{CDT}(P)$.

Pro bod 3 existuje řada algoritmů, např. Sloan (1992).

78. Převod $\mathcal{DT}(P)$ na $\mathcal{CDT}(P)$

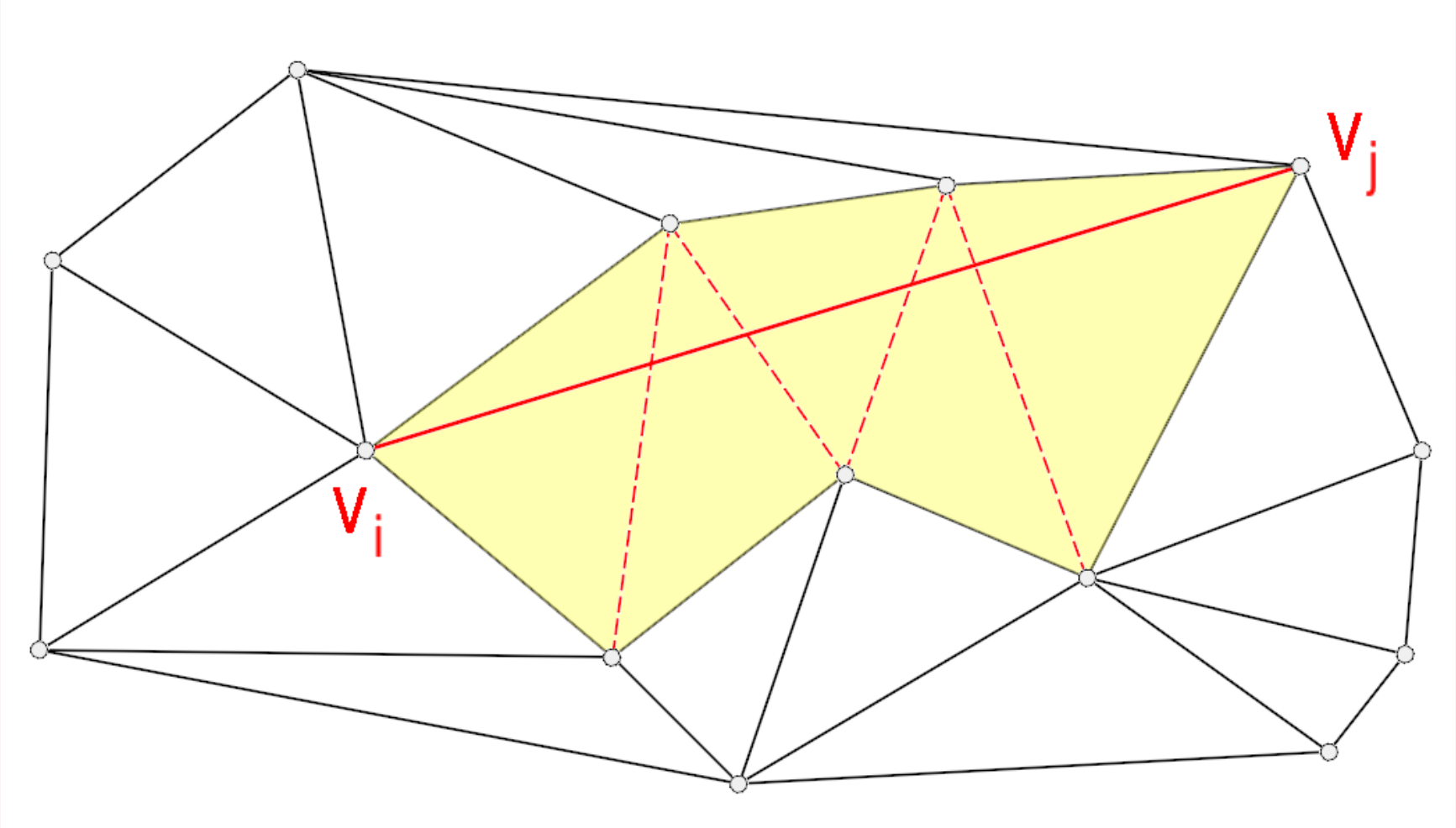
Algoritmus převodu $\mathcal{DT}(P)$ na $\mathcal{CDT}(P)$ je rekurzivní.
Každá *povinná hrana* definována dvojicí vrcholů (v_i, v_j).
Seznam protnutých hran: S.
Seznam nově vytvořených hran: H.

Postup je tvořen následujícími kroky:

- Nalezení stran $\mathcal{DT}(P)$ protínajících povinnou hranu (v_i, v_j).
- Zrušení všech stran v $\mathcal{DT}(P)$ protínajících povinnou hranu (v_i, v_j).
- Vytvoření pomocné triangulace.
- Obnovení $\mathcal{DT}(P)$.
- Odstranění nadbytečných trojúhelníků.

Nalezení stran $\mathcal{DT}(P)$ protínajících povinnou hranu $\overline{v_i v_j}$:
Testujeme, zda hrana (v_i, v_j) není již v $\mathcal{DT}(P)$.
Pokud nikoliv, v $\mathcal{DT}(P)$ nalezeny všechny hrany protínající (v_i, v_j).
Tyto odtraněny z $\mathcal{DT}(P)$ a přidány do S.

79. Nalezení stran $\mathcal{DT}(P)$ protínajících povinnou hranu (v_i, v_j)



80. Odstranění stran protínajících povinnou hranu (v_i, v_j) z $\mathcal{DT}(P)$

Výsledkem převod $\mathcal{DT}(P)$ na pomocnou triangulaci, jejíž hrany neprotínají (v_i, v_j).

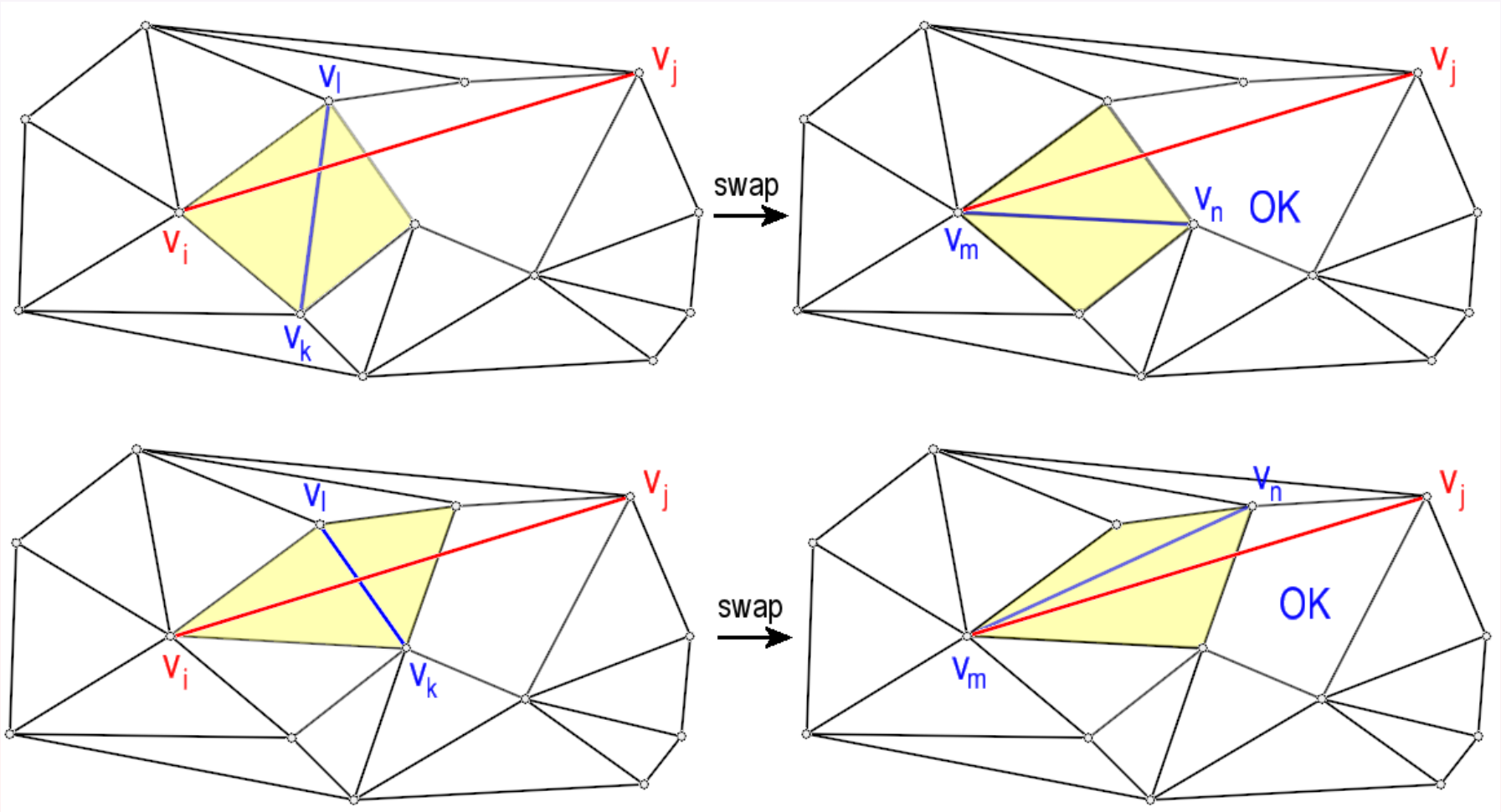
Nechť hrana protínající (v_i, v_j) je označena (v_k, v_l):
Nechť (v_m, v_n) je úhlopříčka v konvexním čtyřúhelníku se společnou stranou (v_k, v_l).

Algoritmus 5: CDT, remove intersecting edges (S,H)

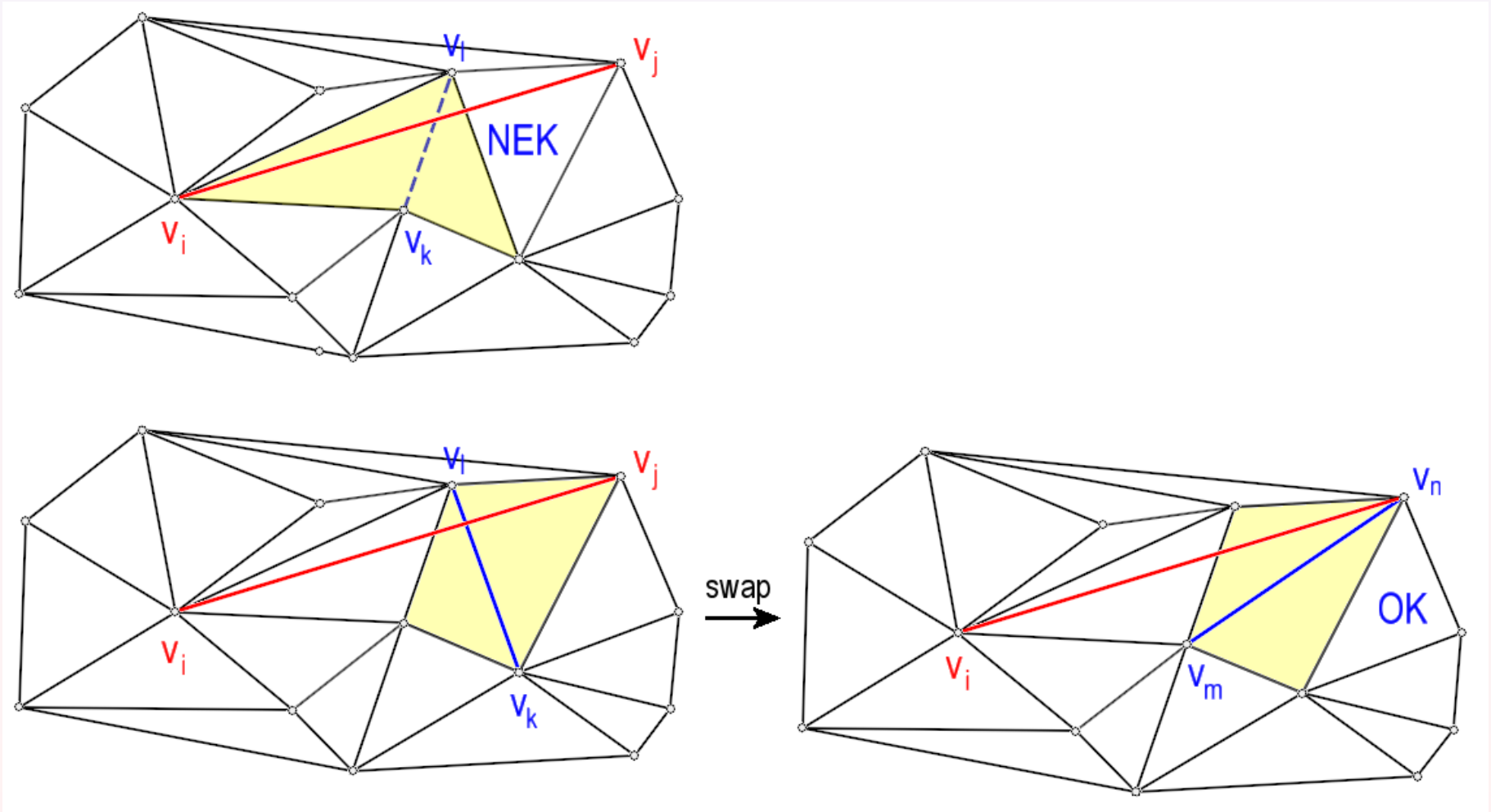
1: Opakuj, dokud S není prázdný
2: Odeber z S protínající hranu (v_k, v_l).
3: Pokud incidující $\triangle$ se společnou hranou (v_k, v_l) netvoří konvexní 4-úhelník
4: Přidej (v_k, v_l) do S a jdi na 2).
5: Pokud tvoří konvexní 4-úhelník:
6: Prohodíme diagonálu (v_k, v_l) v tomto 4 úhelníku za (v_m, v_n) .
7: Pokud (v_m, v_n) neprotíná (v_i, v_j) :
8: Přidej (v_m, v_n) do H.
9: Pokud (v_m, v_n) protíná (v_i, v_j) :
10: Přidej (v_m, v_n) do S.

---

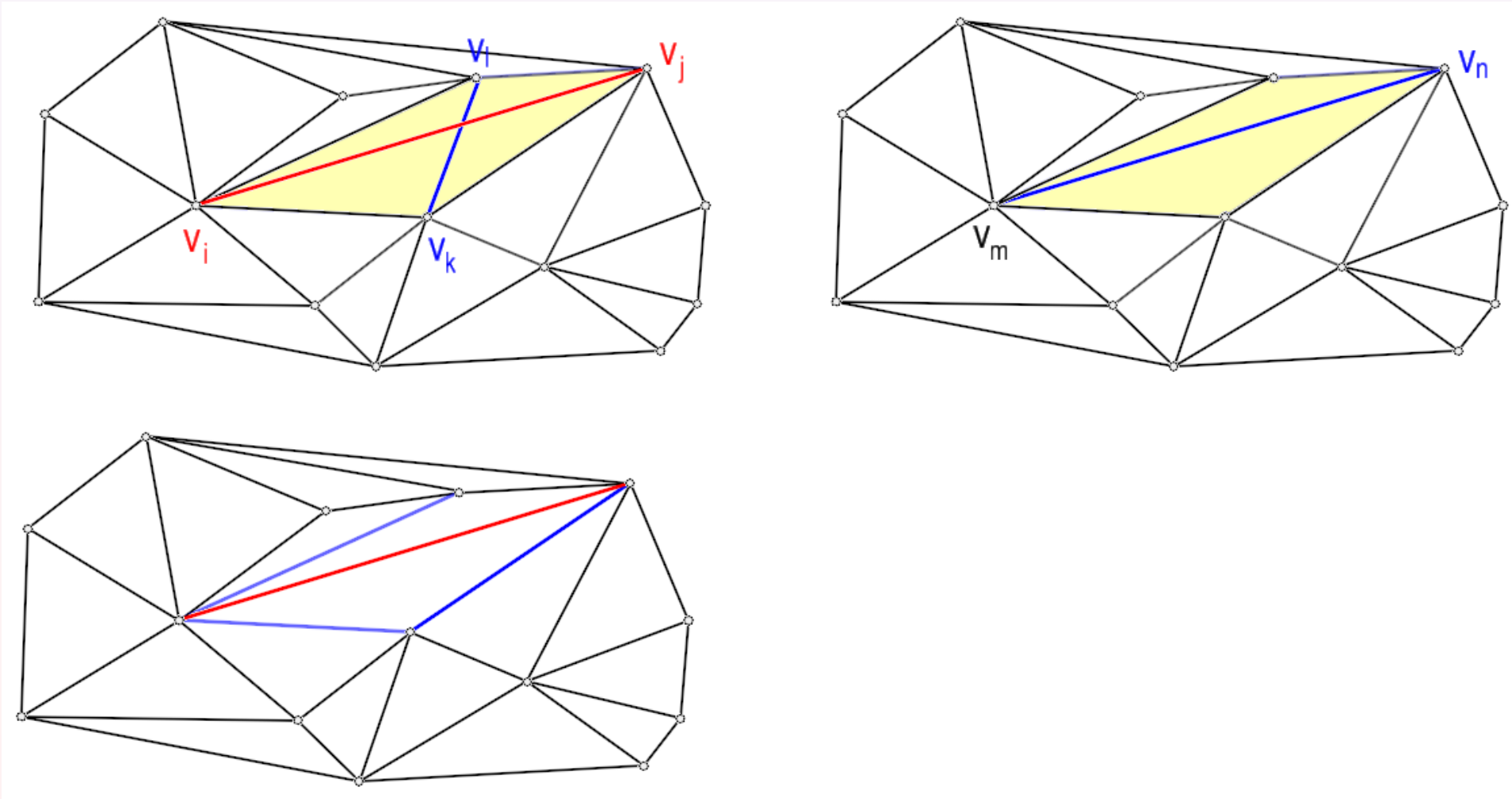
81. Ilustrace odstranění stran protínajících povinnou hranu $\overline{v_i v_j}$ (1/3)



82. Ilustrace odstranění stran protínajících povinnou hranu $\overline{v_i v_j}$ (2/3)



83. Ilustrace odstranění stran protínajících povinnou hranu $\overline{v_i v_j}$ (3/3)



84. Obnovení $\mathcal{DT}(P)$

Triangulace vytvořená v předchozím kroku není Delaunayovská.

Tuto triangulaci je proto nutné převést na Delaunayovskou.

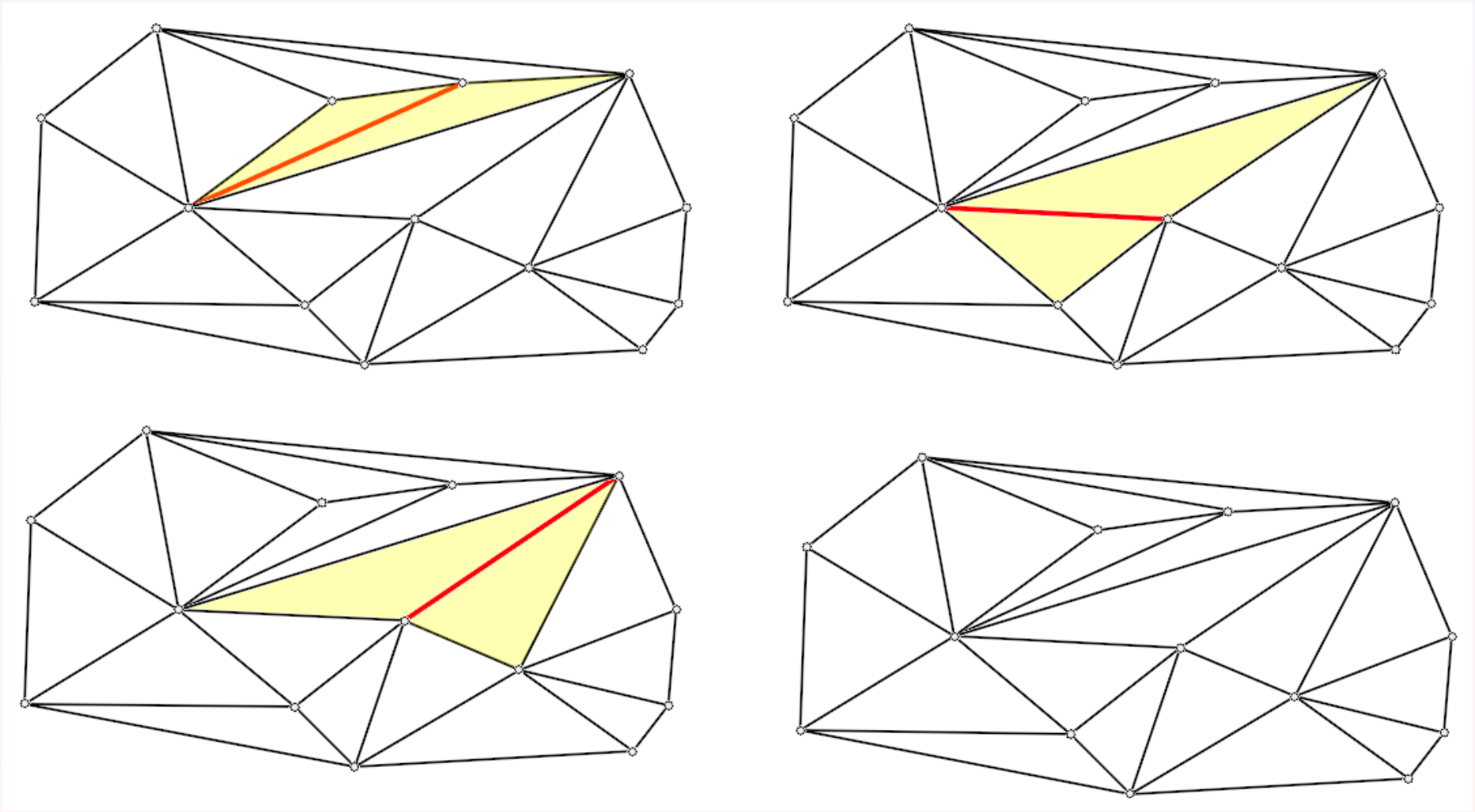
Nad nově vytvořenými hranami je provedena legalizace vzhledem k incidujícím trojúhelníkům.

Algoritmus 6: CDT, Restore DT (S,H)

1: Opakuj, dokud existuje alespoň jeden swap
2: Načti ze seznamu H hranu (v_k, v_l).
3: Pokud $(v_k, v_l) \neq (v_i, v_j)$
4: Pokud incidující $\triangle$ se společnou hranou (v_k, v_l) není legální
5: Prohození diagonály (v_k, v_l) ve 4 úhelníku za (v_m, v_n) .
6: Nahrazení (v_k, v_l) hranou (v_m, v_n) v H.

---

85. Ilustrace obnovení $\mathcal{DT}(P)$

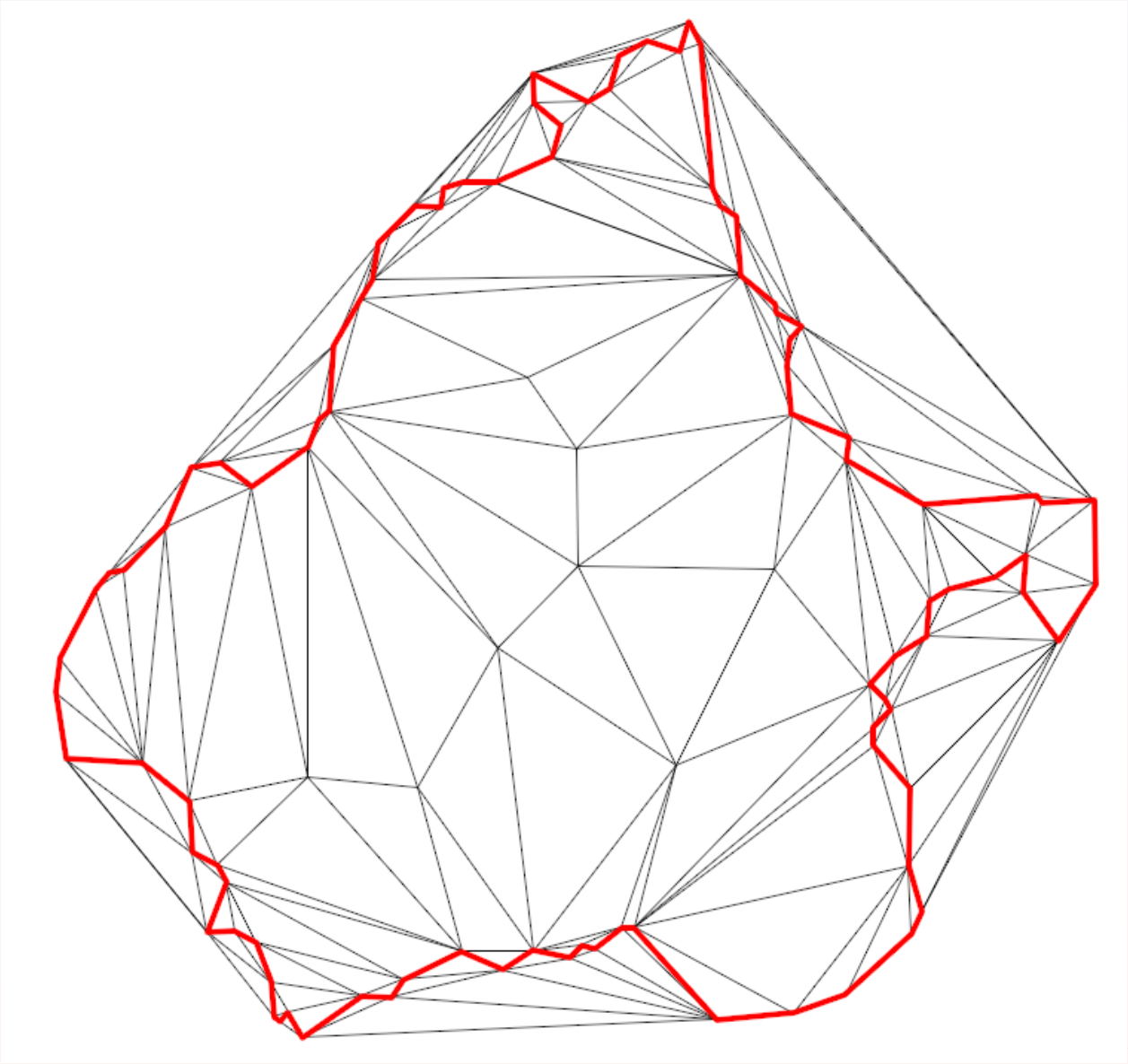


86. Triangulace nekonvexní oblasti a oblasti s otvory

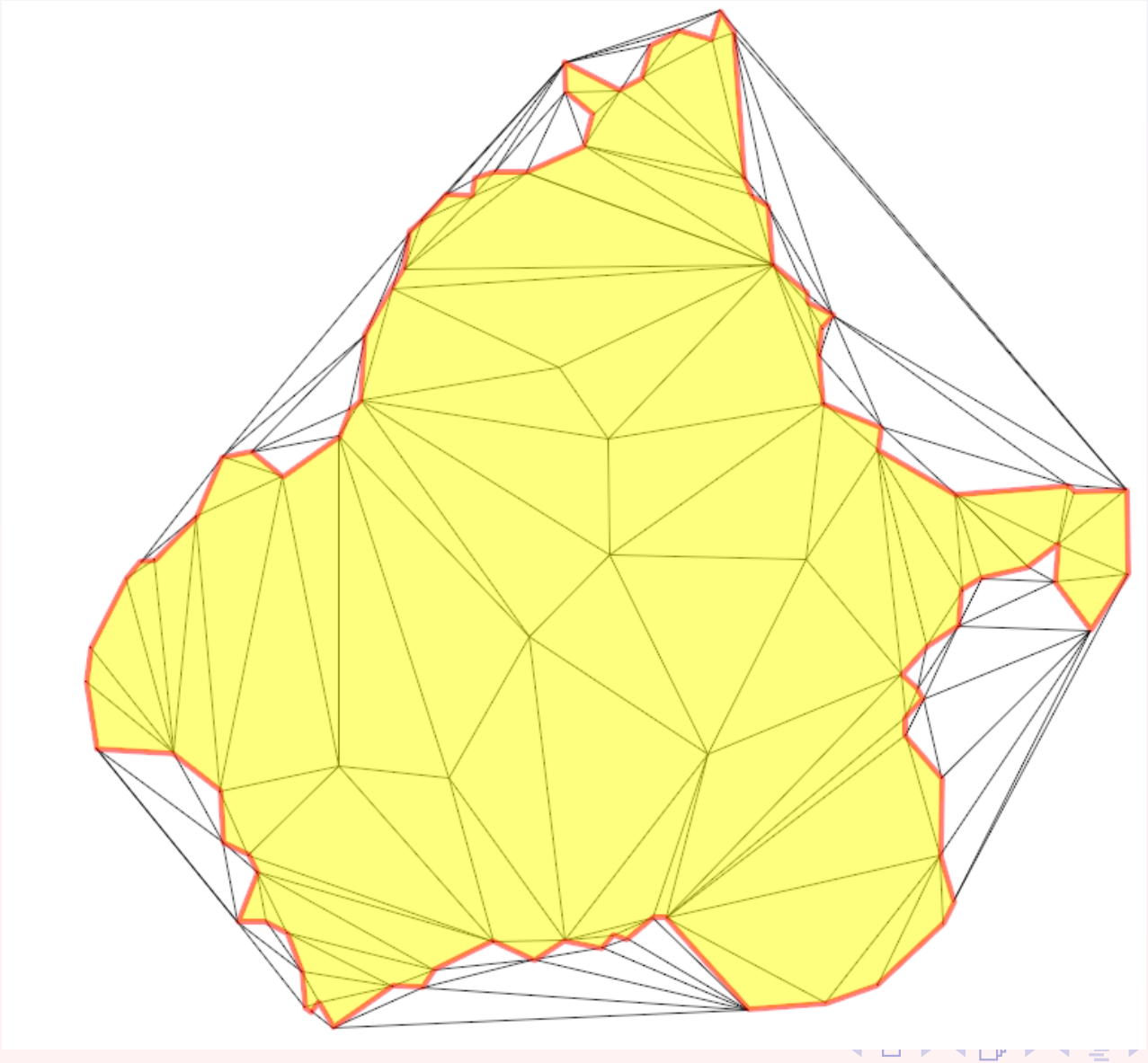
Triangulace nekonvexní oblasti:
Triangulace množiny bodů ohraničené nekonvexním polygonem.
Z triangulace odstraněny všechny trojúhelníky vně polygonu (tj. takové, jejichž těžiště je vně polygonu).
U DMT, triangulace probíhá pouze uvnitř nekonvexní oblasti se vstupními daty, vně oblasti by si algoritmus DMT “vymýšlel”.
Využití Ray Algoritmu.

Triangulace oblastí s otvory:
Oblast obsahuje podoblasti (otvory), uvnitř kterých nebude prováděna triangulace.
Otvory popsány v opačném pořadí, než nadřazená oblast.
Použití při tvorbě DMT, místa bez vrstevnic: vodní plochy, stavební objekty, místa s příliš velkým spádem...

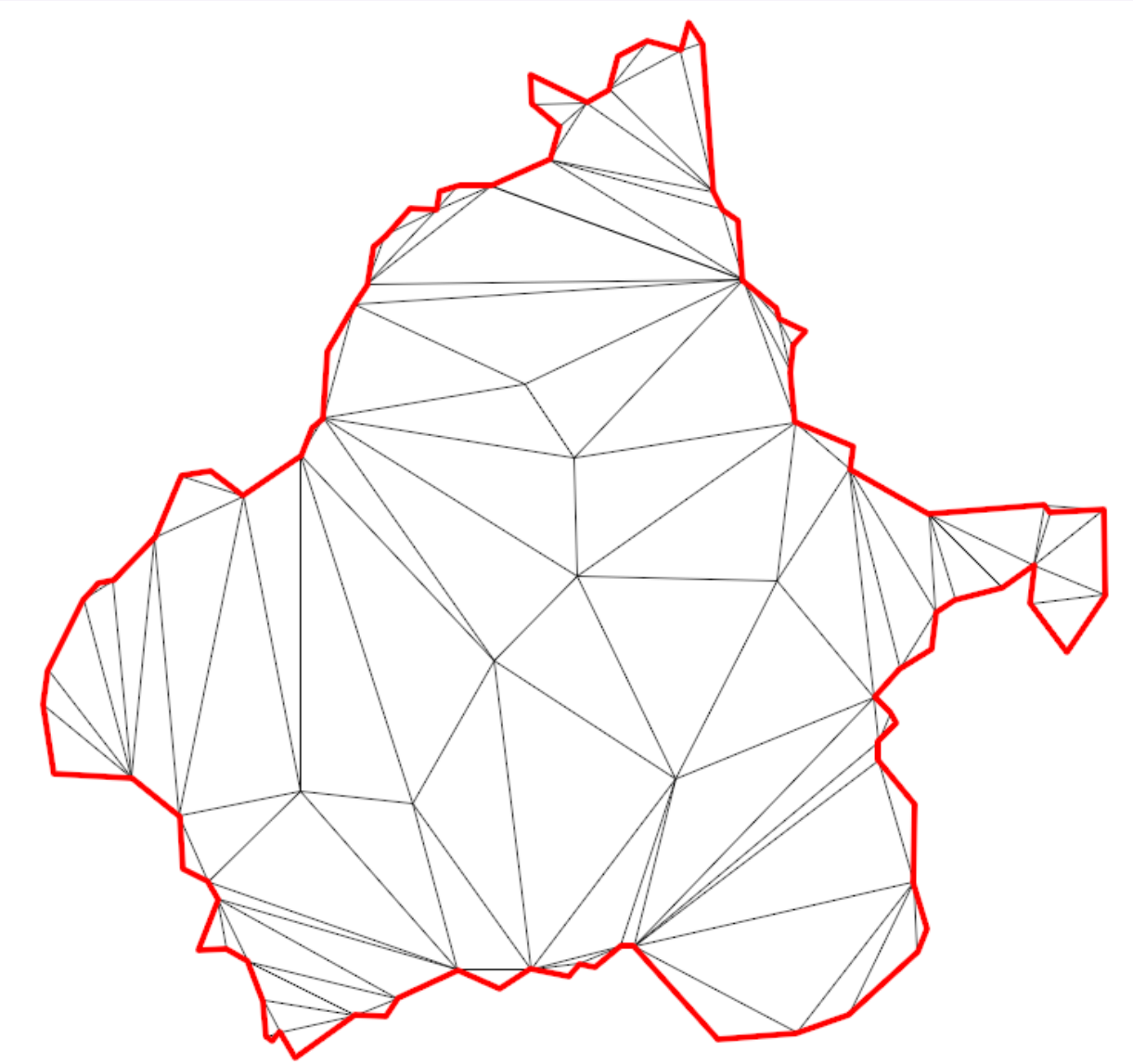
87. Triangulace $\mathcal{DT}(P)$



88. Selekce trojúhelníků uvnitř oblasti



89. Odstranění trojúhelníků vně oblasti



90. Datově závislé triangulace

U všech výše uvedených 2D triangulací tvar trojúhelníkové sítě ovlivňuje pouze poloha bodu, souřadnice z nehraje roli.
Takové triangulace nejsou bez dodatečných informací o terénu (kosterní čáry) vhodné k jeho modelování.

Data Dependent Triangulation (DDT) tyto nedostatky odstraňuje.
Vznikají optimalizací vstupní triangulace (nejčastěji DT) s využitím heuristik či genetických algoritmů.

Výhody:

- DDT berou v potaz výšku bodu, snaha o optimalizaci tvaru trojúhelníkové sítě.
- Trojúhelníkový model lépe zohledňuje skutečný tvar terénu.
- Automatická detekce terénních zlomů, netřeba zadávat povinné hrany.

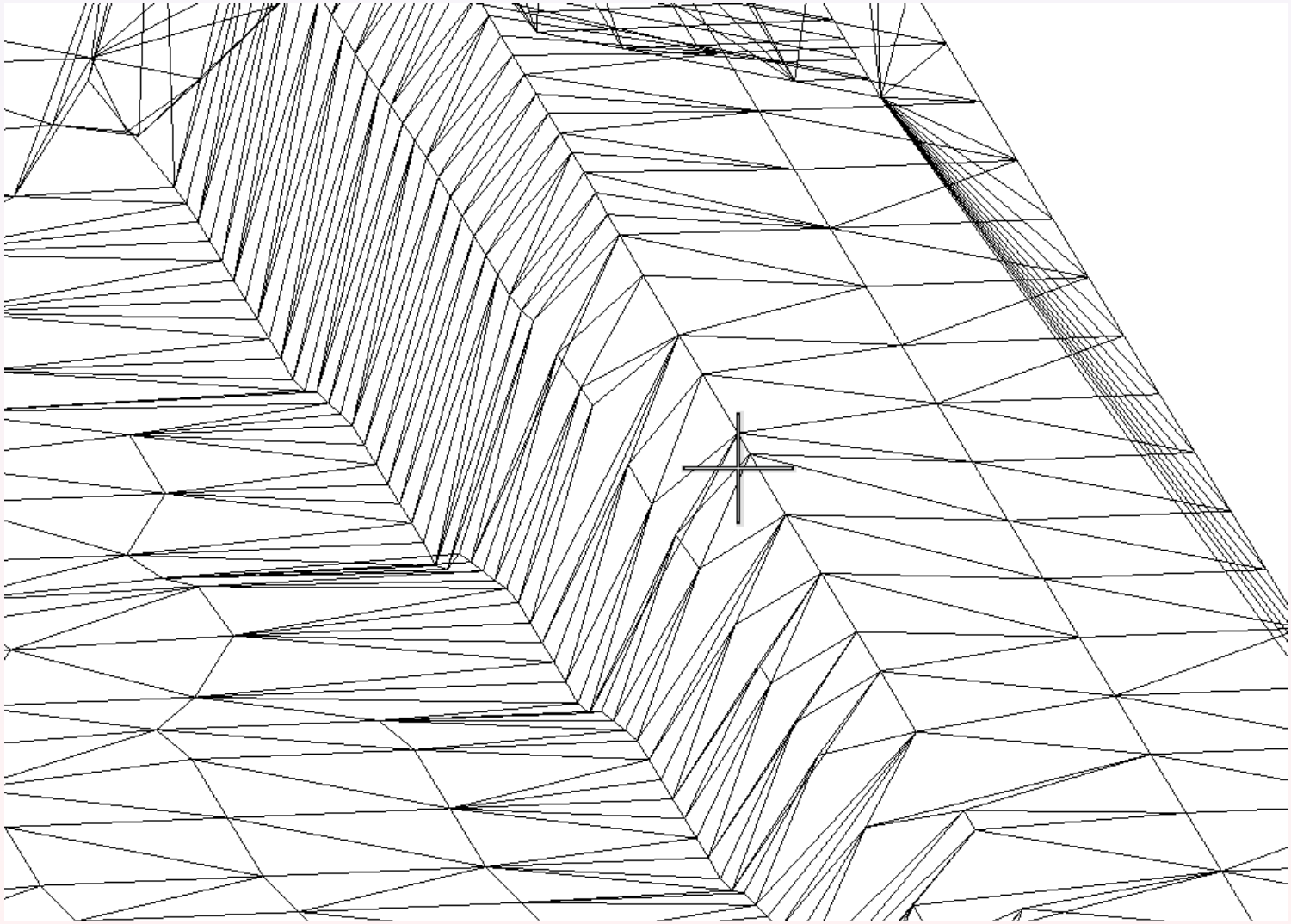
Nevýhody:

- Optimalizace heuristikou rychlá, zlepšení většinou nebývá významné.
- Optimalizace genetickými algoritmy kvalitní, avšak výpočetně náročné, vhodné pouze pro malé množiny ($n < 50000$).

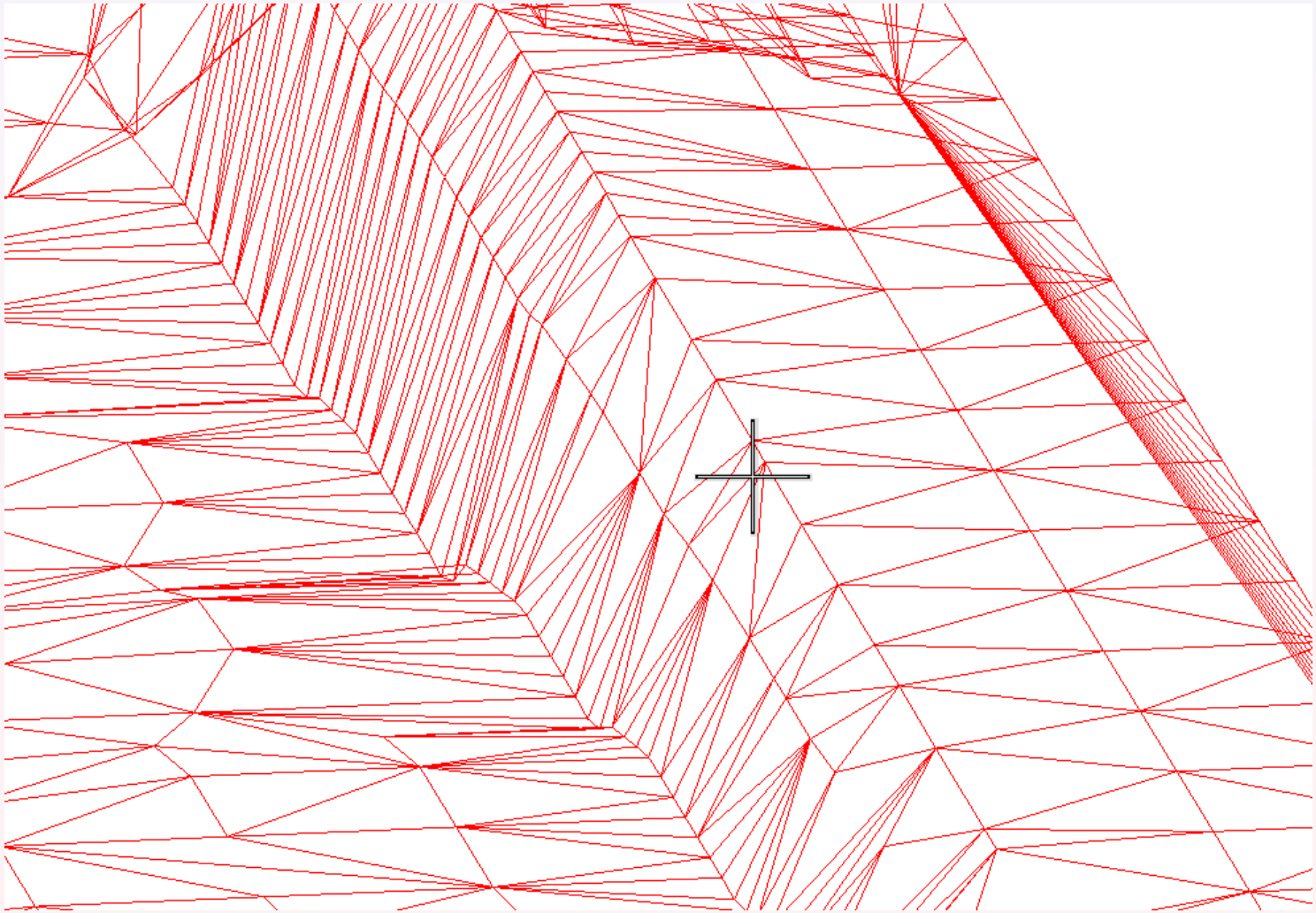
Dvě metody optimalizace:

- lokální optimalizace,
- modifikovaná lokální optimalizace.

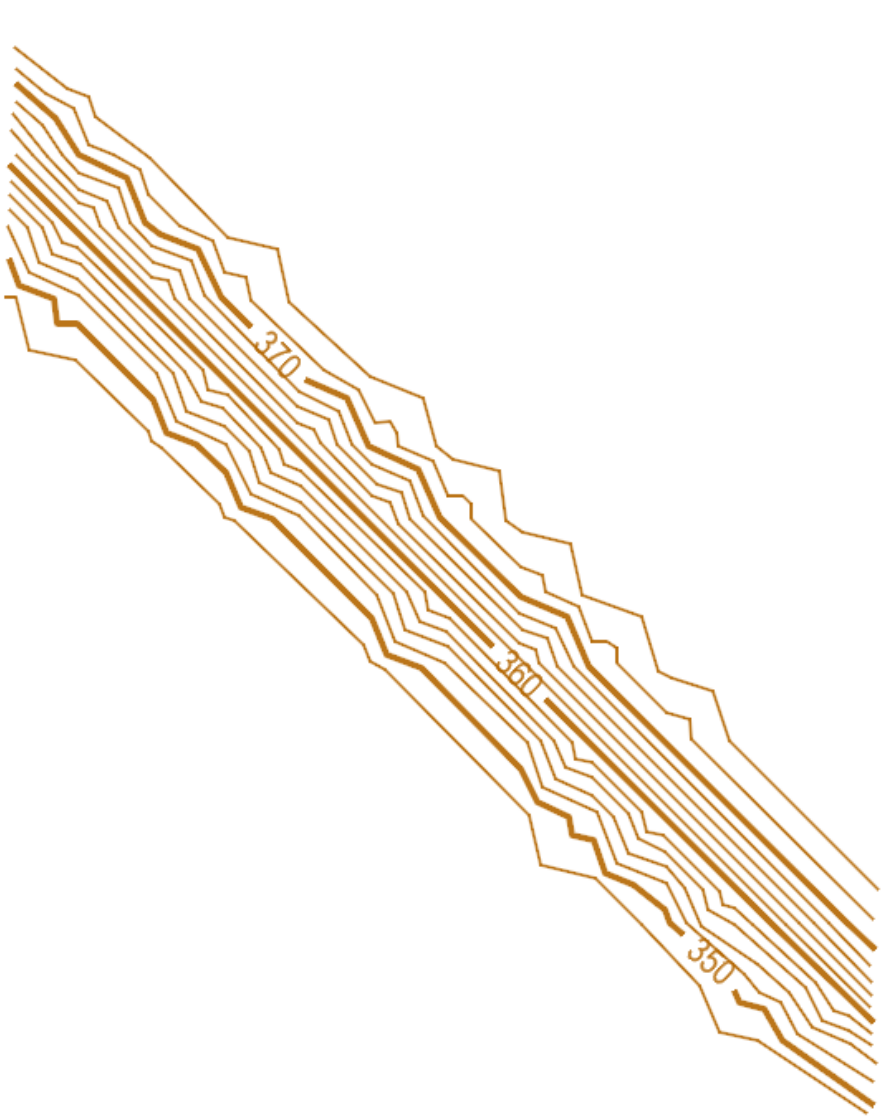
91. DT, nevhodné vystižení terénní hrany

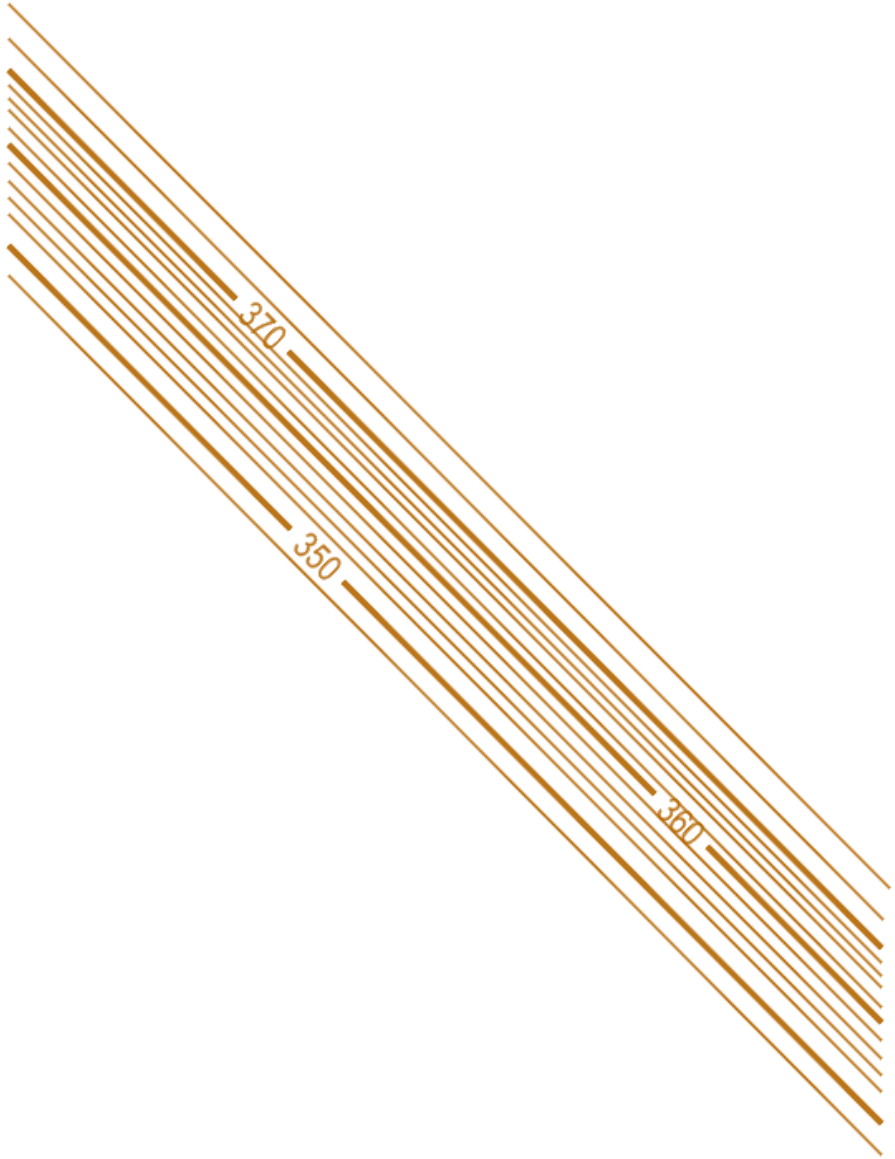


92. DDT, lepší vystižení terénní hrany



93. Srovnání vrstevnic DT a DDT





94. Lokální optimalizace triangulace

Optimalizace triangulace (lokální prohazování hran) vzhledem zadanému kritériu.

Používána heuristika, snaha dosáhnout globálního minima oakovaným hledáním lokálního minima.

V jednom kroku optimalizována “malá” část triangulace:

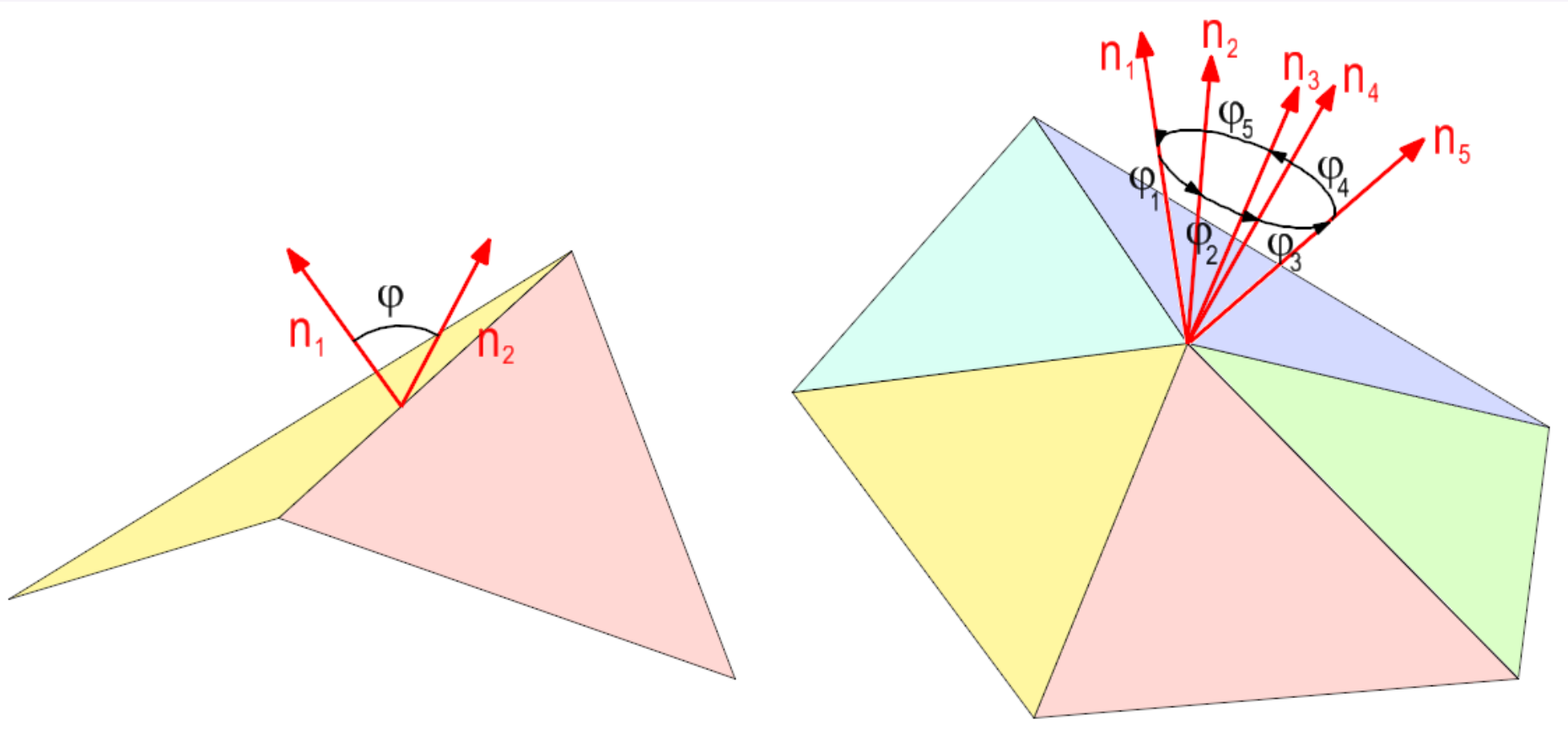
- *Edge Based Optimization*
  Prováděna vzhledem ke každé hraně sdílené dvojicí trojúhelníků.
  Častější varianta.
- *Vertex Based Optimization*
  Prováděna vzhledem ke každému vrcholu sdíleného trojúhelníky.

Rychlé, avšak nepříliš významné zlepšení.

Možné uvíznutí v lokálním minimum, nepřipouští dočasné zhoršení stavu.

Vysoká rychlost konstrukce.

95. Edge Based vs Vertex Based Optimization



96. Edge Based Optimization

Každé hraně e_i trojúhelníka $\triangle$ přiřadí funkce c ohodnocení c_i

$$c_i = c(e_i).$$

Ohodnocovací funkce měří “ostrost přechodu” mezi trojúhelníky (např. úhel normál).

Globální **ohodnocení** triangulace τ s m vnitřními hranami

$$C(\tau) = \sum_{i=1}^{m} c_i = \sum_{i=1}^{m} c(e_i).$$

Optimální triangulace minimalizuje globální kritérium

$$C(\tau) = \min,$$

snaha o co nejvíce “hladkou” triangulaci.

Globální kritérium neumíme exaktně minimalizovat (NP), inkrementální/hladová strategie:

$$C(\tau(k)) = \min \approx C(\tau(k-1)) + \Delta c(e)), \qquad \Delta c(e)) = \min .$$

Minimalizace updatu: kritérium vztažené ke hraně e a blízkému okolí.
Cílem nalézt globální minimum (neúspěšné).

97. Lokální swap kritérium

Výchozí triangulace $\mathcal{T}(P)$: Hrana e_i, inciduje s $\triangle$ t_1, t_2.

2 varianty:

1) Hrana e_i sdílená 2 trojúhelníky
Lokální swapovací kritérium: $\mathcal{T}(P)$ je optimální vzhledem k c právě když

$$c(e_i) < c(e_i').$$

Nízká účinnost, malé území.

2) Hrana e_i sdílená 2 trojúhelníky + incidující hrany.
Celkem 6 trojúhelníků, větší účinnost.
Sousedící hrany v t_1: e_i^1, e_i^2, sousedící hrany v t_2 : e_i^3, e_i^4, ohodnocení

$$C(\tau) = c(e_i) + \sum_{k=1}^{4} c(e_i^k)$$

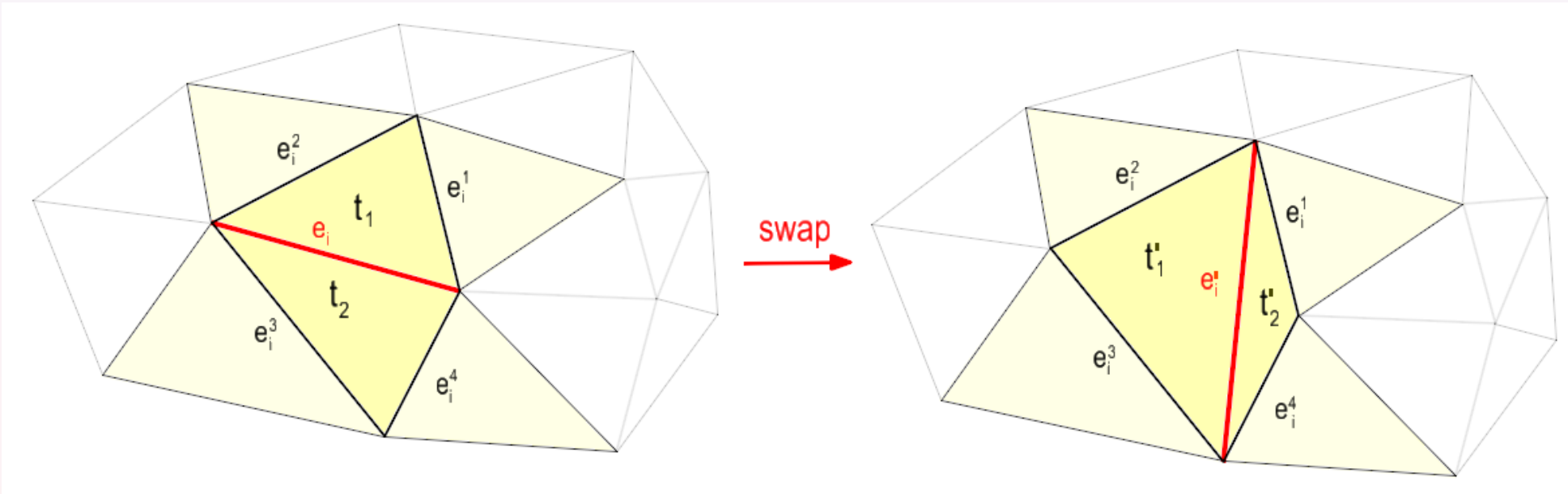
Swap $\Rightarrow$ triangulace $\mathcal{T}'(P)$ s trojúhelníky t_1' a t_2', ohodnocení

$$C(\tau') = c(e_i') + \sum_{k=1}^{4} c(e_i^k).$$

Lokální swapovací kritérium: $\mathcal{T}(P)$ je optimální vzhledem k c právě když

$$C(\tau) < C(\tau') \Leftrightarrow c(e_i) + \sum_{k=1}^{4} c(e_i^k) < c(e_i') + \sum_{k=1}^{4} c(e_i^k).$$

98. Ukázka lokální optimalizace



99. Algoritmus lokální optimalizace

Algoritmus provádí opakované swapování nad všemi hranami triangulace.
Výsledná triangulace nebude globálně optimální k žádnému kritériu.

Algoritmus 7: DDT, LOP (e)

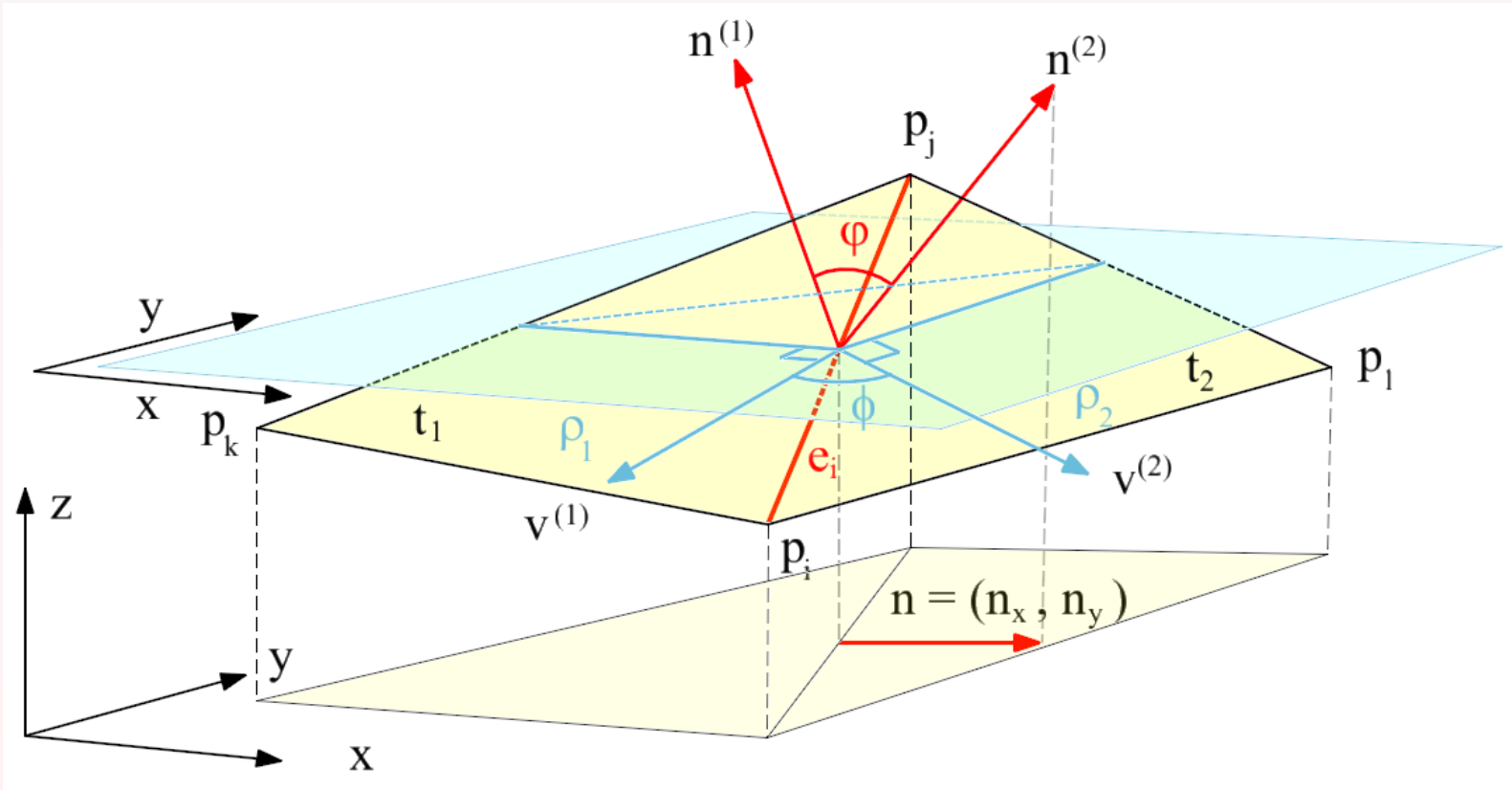
1: Opakuj, dokud existuje alespoň jeden swap
2: Vezmi hranu e_i.
3: Spočti $c_i = c(e_i) + \sum_{k=1}^{4} c(e_i^k)$
4: Swap $(e_i \rightarrow e_i')$
5: Spočti $c_i' = c(e_i') + \sum_{k=1}^{4} c(e_i^k)$
6: if $c_i < c_i'$
7: Swap $(e_i' \rightarrow e_i)$.

---

100. Lokální kritéria pro DDT

Přehled kritérií:

- Angle Between Normals (ABN).
- Distance From Planes (DFP).
- Smoothnes of Contours (SCO).



101. Přehled kritérií

Rovina ϱ_i

$$\varrho_i(x, y) = a_i x + b_i y + c_i z$$

Angle Between Normals:
Úhel φ mezi normálami $n^{(1)}, n^{(2)}$ trojúhelníků t_1, t_2

$$c_{ABN}(e_i) = \varphi = \cos^{-1} \frac{n^{(1)} \cdot n^{(2)}}{\|n^{(1)}\| \, \|n^{(2)}\|}, \text{ kde } n^{(i)} = (\frac{\partial \rho_i}{\partial x_i}, \frac{\partial \rho_i}{\partial y_i}, \frac{\partial \rho_i}{\partial z_i}) = (-a_i, -b_i, 1)$$

Distance From Planes:
Součet vzdálenost bodů p_k, p_l od rovin ϱ_1, ρ_2.

$$c_{DFP}(e_i) = d(p_k, \rho_1)^p + d(p_l, \rho_2)^p, \text{ kde } p = 1, 2$$

Smoothness Of Contours:
Úhel ϕ mezi $v^{(1)}, v^{(2)}$ (průměty $n^{(1)}, n^{(2)}$ v ϱ_0, vodorovná rovina)

$$c_{SCO}(e_i) = \phi = \cos^{-1} \frac{v^{(1)} \cdot v^{(2)}}{\|v^{(1)}\| \, \|v^{(2)}\|} = \cos^{-1} \frac{a_1 a_2 + b_1 b_2}{\sqrt{a_1^2 + b_1^2}\sqrt{a_2^2 + b_2^2}}.$$

102. Zemský povrch a jeho znázornění

Zemský povrch má nepravidelný, komplikovaný průběh:

- *Hladký:*
  Konvexní či konkávní.
  Formován prostřednictvím přírodních jevů.
  Snadnější pro matematické modelování.
- *Ostrý:*
  Zlomy, zářezy, hrany, stupně.
  Formován činností člověka.
  Umělé terénní tvary tvoří *singularity*, obtížnější pro matematické modelování.

Prostorové modely zemského povrchu:

- Digitální model reliéfu (Digital Terrain Model).
- Digitální model povrchu (Digital Surface Model).
- Digitální výškový model (Digital Elevation Model).

103. Digitální model terénu

Digitální model terénu/reliéfu:
Digitální reprezentace reliéfu zemského povrchu v paměti počítače, složená z dat a interpolačního algoritmu, který umožňuje mj. odvozovat výšky mezilehlých bodů. (Terminologický slovník ČÚZK)

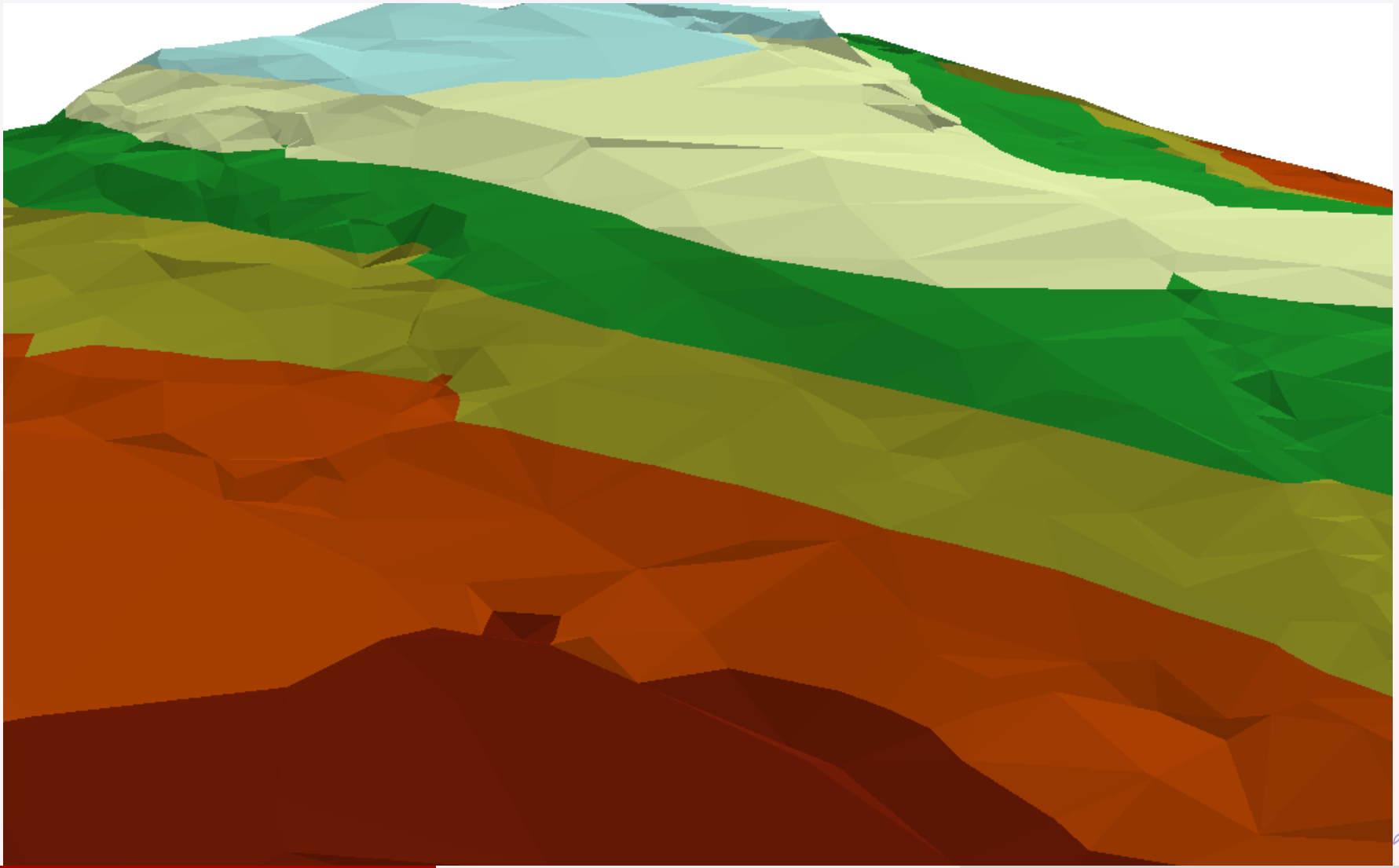
Digitální model povrchu:
Zvláštní případ digitálního modelu reliéfu konstruovaného zpravidla s využitím automatických prostředků (např.obrazové korelace ve fotogrammetrii) tak, že zobrazuje povrch terénu a vrchní plochy všech objektů na něm (střechy, koruny stromů a pod.). (Terminologický slovník ČÚZK)

Digitální výškový model:
Digitální model reliéfu pracující výhradně s nadmořskými výškami bodů.
(Terminologický slovník ČÚZK).

Nad digitální modely lze provádět řadu analýz a výpočtů:
analýza sklonu, osvětlení, expozice, barevná hypsometrie, vrstevnice.

104. Znázornění DMT: trojúhelníková síť

105. Znázornění DMT: barevná hypsometrie (kvantitativní použití barev)



106. Plátování

Základem DMT aproximační plocha procházející všemi zadanými body množiny $P = \{p_i\}_{i=1}^{n}$, kde $p_i = [x_i, y_i, z_i]$.
Mimo tyto body dopočítávána podle specifických matematických postupů, aby se co nejvíce blížila původnímu terénu.
Vede ke vzniku ploch vysokých stupňů, které samovolně oscilují.

Vhodnější použít techniku *plátování*.

Princip plátování:
Rozdělení aproximační plochy na větší množství malých ploch nižších stupňů⇒*pláty*.
Pláty nejčastěji stupně tři⇒kubické pláty (polynomy stupně 3 již věrně aproximují terén, jejich výpočet poměrně snadný).

Hranice plátů jsou vedeny po singularitách.

Digitální model tvořen velkým množstvím plošek (řádově stovky tisíc, milióny), mezi nimi ostré nebo hladké přechody⇒ tímto způsobem lze vyjádřit jakýkoliv terén.

Poprvé použito v 70. letech při konstrukci letadel (Airbus=Bezierovy pláty, Boeing=Coonsovy pláty).

107. Polyedrický model terénu

Plošky jsou představovány trojúhelníky se společnou hranou.
Síť trojúhelníků vytvořena za použití triangulačních algoritmů (Constrained Delaunay Triangulation).

Proložením rovin vrcholy jednotlivých trojúhelníků v $\mathbb{R}^3$ vznikne nepravidelný mnohostěn, tzv. **polyedr**, který se přimyká k terénu.

V trojúhelnících pouze lineární interpolace, což pro řadu účelů nepostačuje.

Do polyedrického modelu lze zadat povinné hrany (hřbetnice, údolnice, spádnice), které zlepšují jeho aproximační vlastnosti.

Vzhledem k nepravidelnému rozložení bodů je nutné při interpolaci/extrapolaci dat či různých analytických operacích z polyedrického modelu používat speciální techniky (např. IDW nebo Krigging).

108. Konstrukce polyedrického modelu

Vrcholy $p_1 = [x_1, y_1, z_1]$, $p_2 = [x_2, y_2, z_2]$, $p_3 = [x_3, y_3, z_3]$ každého trojúhelníku t proložíme rovinu $\mathcal{T}$

$$z = ax + by + c.$$

Koeficienty a, b, c představující složky normálového vektoru roviny

$$a = \frac{\begin{vmatrix} y_1 & z_1 & 1 \\ y_2 & z_2 & 1 \\ y_3 & z_3 & 1 \end{vmatrix}}{\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}} \quad b = \frac{\begin{vmatrix} x_1 & z_1 & 1 \\ x_2 & z_2 & 1 \\ x_3 & z_3 & 1 \end{vmatrix}}{\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}} \quad c = \frac{\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}}{\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}}.$$

Rovnice jednotlivých rovin nejsou udržovány v paměti, jsou podle potřeby operativně určovány (on the fly) $\Rightarrow$ výhoda pro práci s rozsáhlými modely.

109. Interpolace vrstevnic

Lineární interpolační algoritmy:
Spád terénu mezi dvěma body, mezi kterými provádíme interpolaci, je konstantní.
Rozestup vrstevnic mezi dvěna body je také konstantní.
Výpočetně jednoduché, ale nevystihuje realitu.

Nelineární interpolační algoritmy:
Mezi interpolovanými body předpokládáme plynulou změnu sklonu terénu $\Rightarrow$ *geomorfologická* interpolace.

Rozestup vrstevnic mezi dvěma body není konstantní.
Zohledňuje skutečný tvar terénu (sklon okolních plošek).
Využití kvadratické či kubické interpolace.

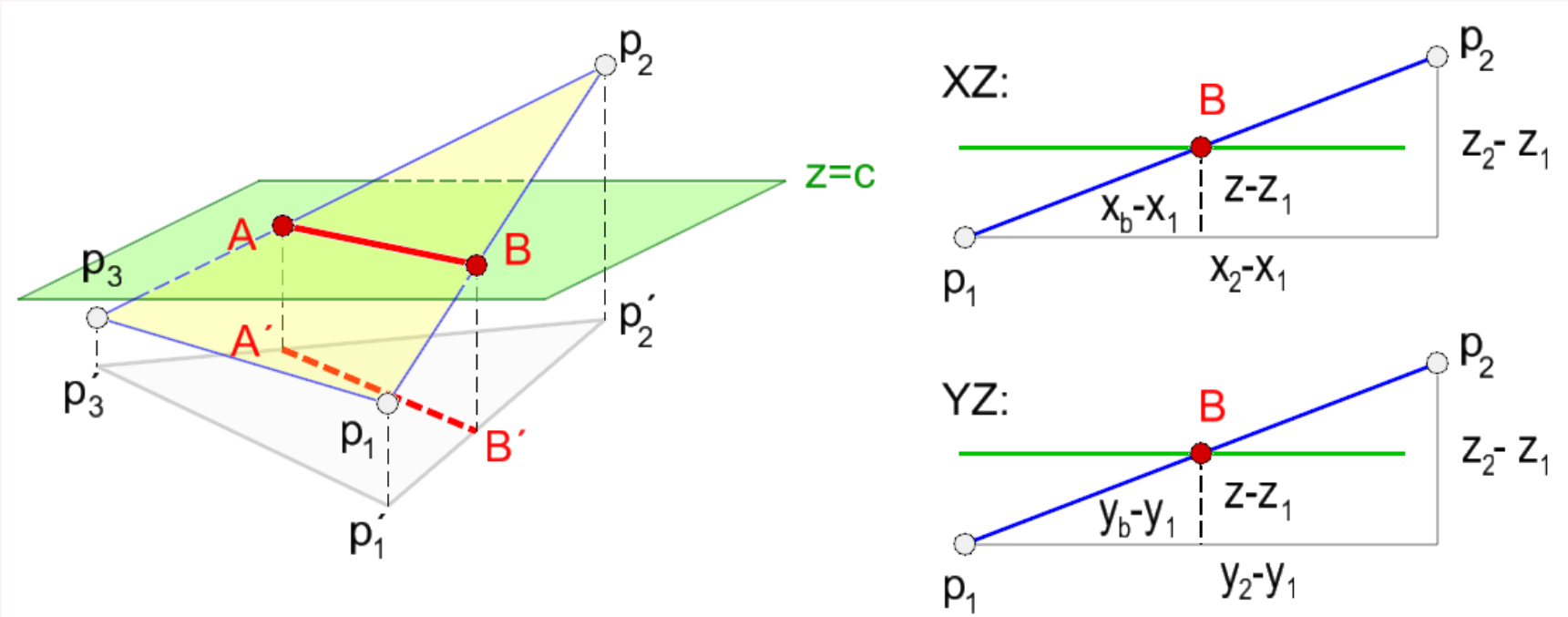
Používá se v mapách velkých a středních měřítek.
Postup je značně složitý a obtížně se algoritmizuje.

110. Konstrukce vrstevnic lineární interpolací

Dáno:	Rovina plátu $\mathcal{T}(p_1, p_2, p_3)$, rovina $\rho : z = h$.
Hledáme:	Průsečnice $\overline{AB}$ rovin ρ a $\mathcal{T}$.

Založena na analytické geometrii: hledání průsečnice roviny $\mathcal{T}$ určené trojúhelníkem $t \in \mathcal{DT}$ a vodorovné roviny s výškou h.

Opakováno nad všemi t.



111. Výpočet souřadnic bodů A, B

Varianty vzájemné polohy ϱ a τ:
Nemají žádný společný bod (neřešíme), průsečnice tvoří 1 bod (neřešíme), průsečnice je úsečka.

Z podobnosti trojúhelníků představujících průměty do roviny XZ a YZ platí:

$$\frac{x_2 - x_1}{z_2 - z_1} = \frac{x_b - x_1}{z - z_1} \qquad \frac{y_2 - y_1}{z_2 - z_1} = \frac{y_b - y_1}{z - z_1},$$
$$\frac{x_3 - x_1}{z_3 - z_1} = \frac{x_a - x_1}{z - z_1} \qquad \frac{y_2 - y_1}{z_2 - z_1} = \frac{y_a - y_1}{z - z_1}.$$

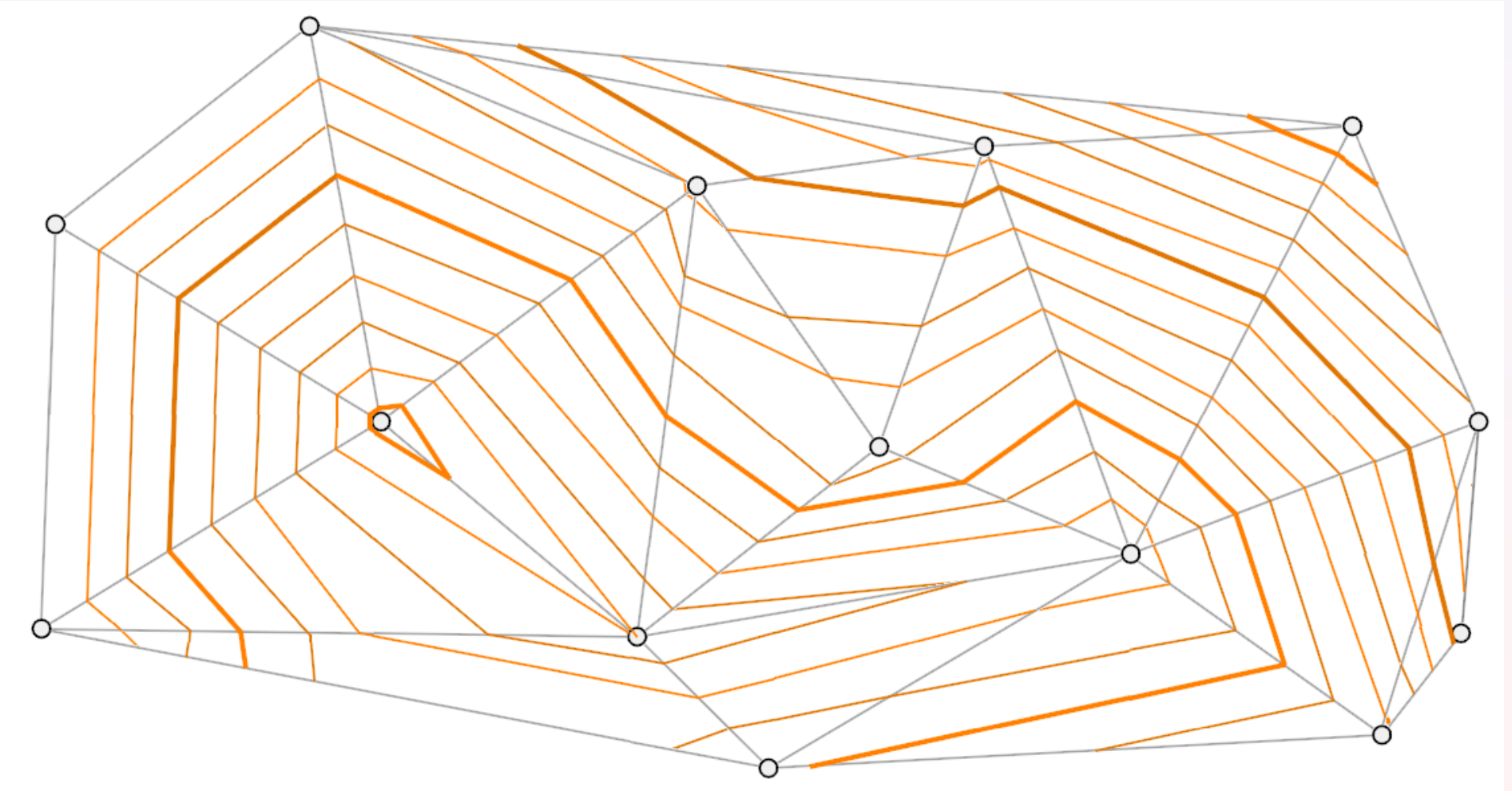
Výsledné souřadnice koncových bodů A, B průsečnice určíme ze vztahů

$$x_a = \frac{x_3 - x_1}{z_3 - z_1}(z - z_1) + x_1 \qquad x_b = \frac{x_2 - x_1}{z_2 - z_1}(z - z_1) + x_1,$$
$$y_a = \frac{y_3 - y_1}{z_3 - z_1}(z - z_1) + y_1 \qquad y_b = \frac{y_2 - y_1}{z_2 - z_1}(z - z_1) + y_1.$$

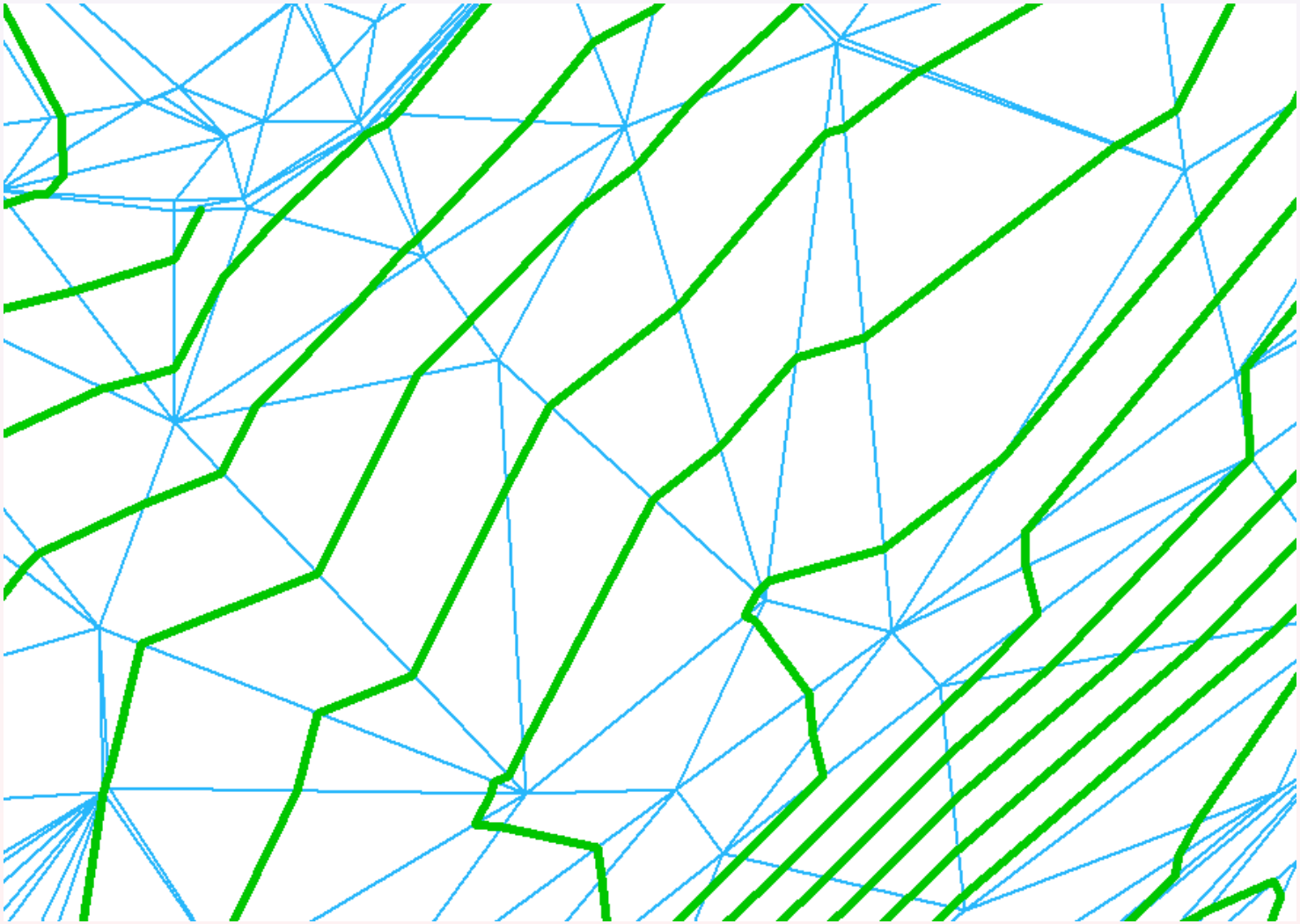
Test, zda rovina ϱ protíná stranu (p_i, p_{i+1}) trojúhelníku:

$$(z - z_i)(z - z_{i+1}) \quad < \quad 0.$$

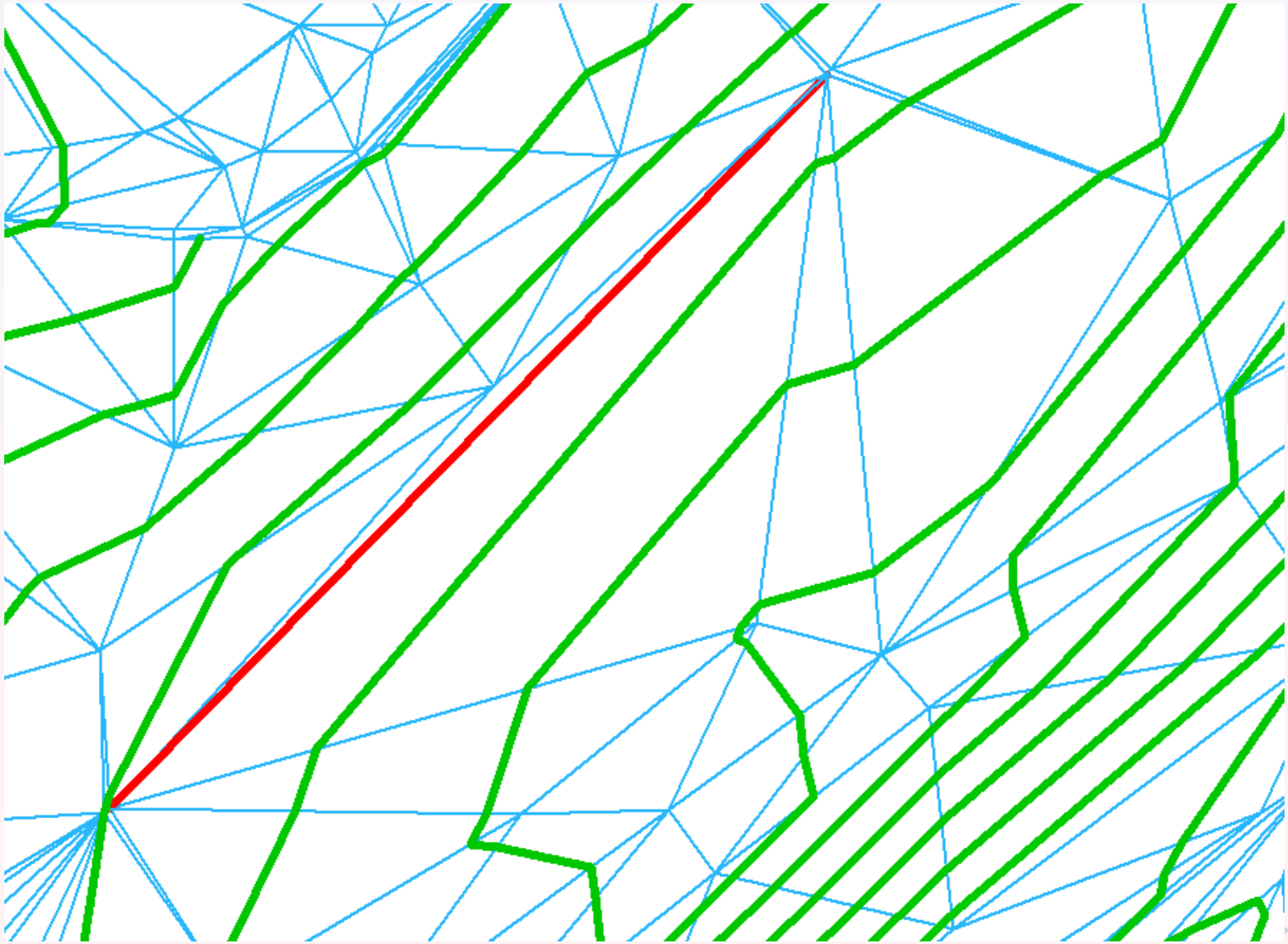
112. Ukázka výpočtu vrstevnic lineární interpolací (DT)



113. Vliv vložení povinné hrany do triangulace: výchozí situace



114. Vliv vložení povinné hrany do triangulace: lokální změna průběhu DMT



115. Analýza sklonu terénu

Analytická úloha realizovaná nad DMT.
Použití pro analýzu hydrologických poměrů, sesuvů, lavin, návrhy komunikací, stavebních objektů.

Zprostředkující hodnotou je *gradient* (tj. vektor max. spádu).
Gradient $\nabla f(x_0, y_0, z_0)$ funkce $f(x, y, z)$ v bodě $p = [x_0, y_0 z_0]$

$$\nabla f(x_0, y_0, z_0) = (\frac{\partial f}{\partial x}(x_0), \frac{\partial f}{\partial y}(y_0), \frac{\partial f}{\partial z}(z_0)).$$

Rovnice roviny ϱ

$$\rho : ax + by + cz + d = 0.$$

Gradient $\nabla \rho(x_0, y_0, z_0)$ roviny ρ

$$\nabla \rho(x_0, y_0, z_0) = (\frac{\partial \rho}{\partial x}(x_0), \frac{\partial \rho}{\partial y}(y_0), \frac{\partial \rho}{\partial z}(z_0)) = (a, b, c).$$

116. Analýza sklonu terénu

Rovina ρ procházející 3 body

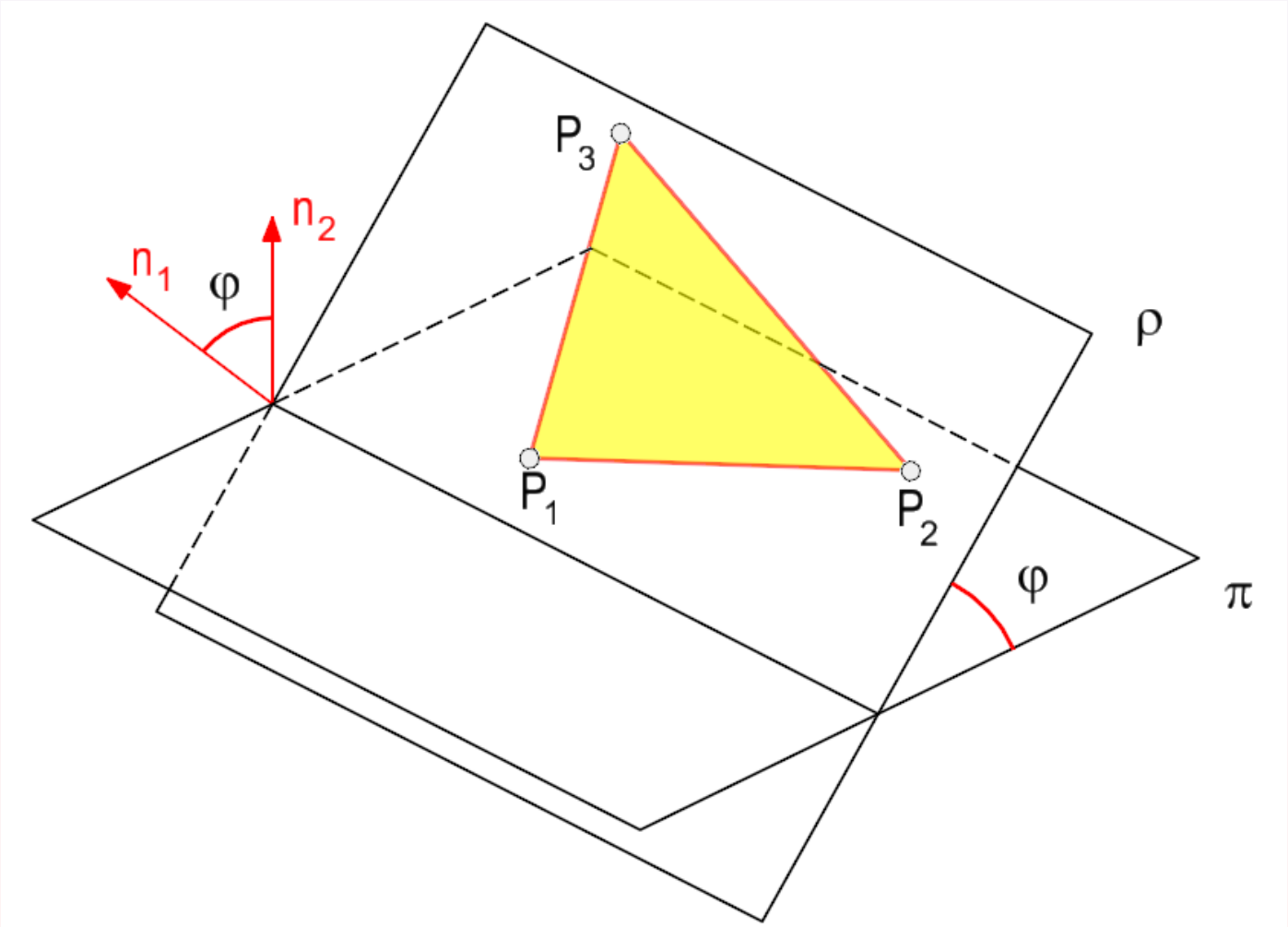
$$\begin{vmatrix} x - x_1 & y - y_1 & z - z_1 \\ x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \end{vmatrix} = 0.$$

Odchylka φ rovin ρ a π:

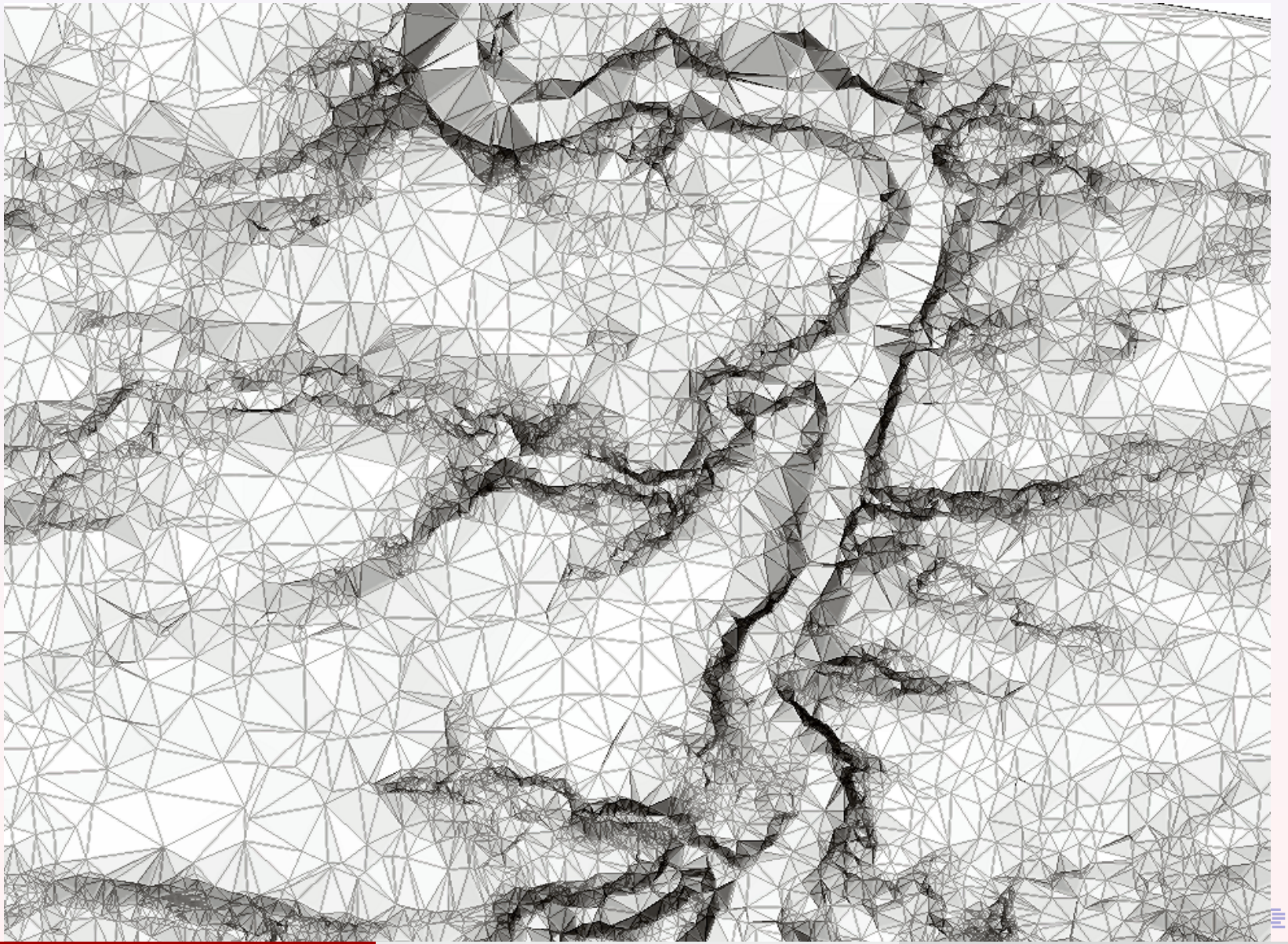
$$\begin{aligned} \varphi &= \arccos \left| \frac{n_1 . n_2}{\|n_1\| \, \|n_2\|} \right|, \; \varphi \in \left\langle 0, \frac{\pi}{2} \right\rangle, \\ n_1 &= (a, b, c), \\ n_2 &= (0, 0, 1). \end{aligned}$$

Výpočet prováděn nad každým trojúhelníkem t DMT.
Vizualizace trojúhelníků (tj. vyplnění barvou) na základě hodnoty φ.

117. Sklon terénu



118. Vizualizace sklonu terénu



119. Analýzy orientace terénu

Využití ve stavebnictví, zemědělství, hydrologii.
Sluneční svit ovliňuje množství tepla dopadajícího na zemský povrch, hydrologické poměry, podmínky pro růst zemědělských plodin.

Orientace v bodě definována jako azimut průmětu gradientu $\nabla\rho$ do roviny x, y.

Vektor v průmětem gradientu $\nabla\rho(x_0, y_0, z_0)$ do roviny xy

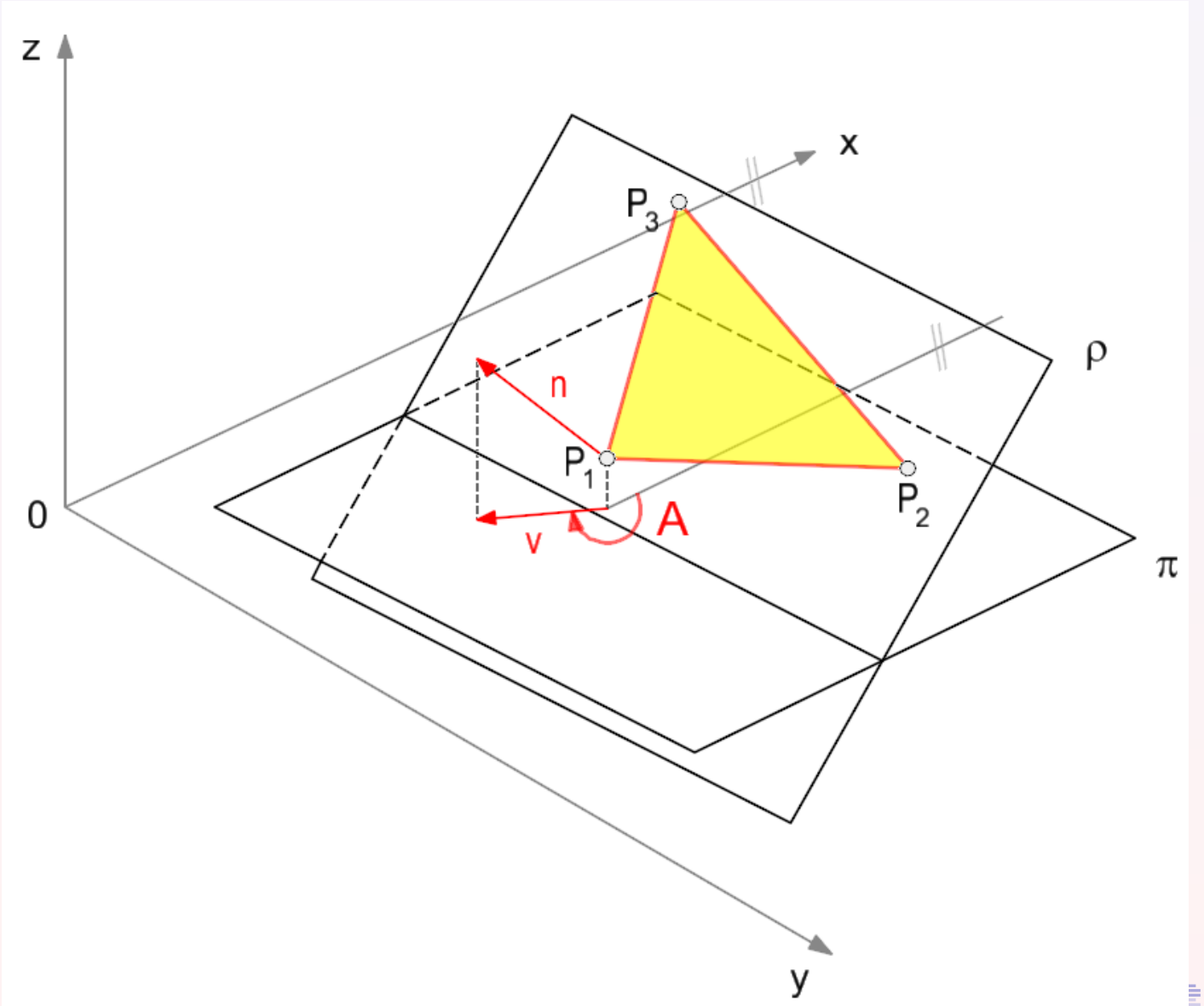
$$v = (\frac{\partial\rho}{\partial x}(x_0), \frac{\partial\rho}{\partial y}(y_0), 0) = (a, b, 0).$$

Azimut α vektoru v

$$A = \arctan\left(\frac{b}{a}\right), \ A \in \langle 0, 2\pi\rangle .$$

Výpočet prováděn nad každým trojúhelníkem DMT.
Pozor na správnou detekci kvadrantů.

120. Orientace terénu



121. Vizualizace orientace terénu