План занятия (10.11.17)

- Построение выпуклой оболочки в трехмерном пространстве
- Триангуляция

Выпуклая оболочка в 3D

- Воспользуемся идеей “заворачивания подарка”, использованной в алгоритме Джарвиса
- Пусть $P = \{ p_1, p_2, p_3, \dots , p_n \}$ - точки в трехмерном пространстве. Считаем, что
  - $n >= 3$ и точки не лежат на одной прямой
  - точки не компланарны (не лежат в одной плоскости): в случае их компланарности задача сводится к двумерному случаю

Выпуклая оболочка в 3D

- Выпуклая оболочка - пересечение всех выпуклых подмножеств R^3, содержащих *P*.
- Выпуклая оболочка может быть получена как пересечение конечного числа полупространств, каждое из которых “опирается” на одну из граней

Выпуклая оболочка в 3D

- *H* - полупространство

$$H = \{(x, y, z) \mid ax + by + cz + d \geqslant 0\}$$

- *dH* - граница полупространства

$$\partial H = \{(x, y, z) \mid ax + by + cz + d = 0\}$$

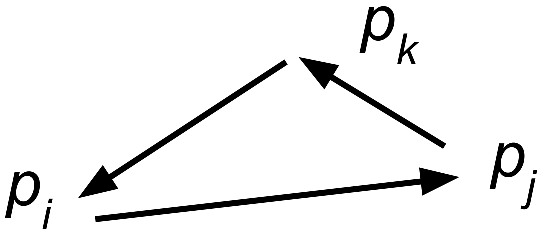
- *H* “опирается” на грань, если *P* содержится в *H* и *dH* содержит как минимум 3 неколлинеарные точки из *H*

Выпуклая оболочка в 3D

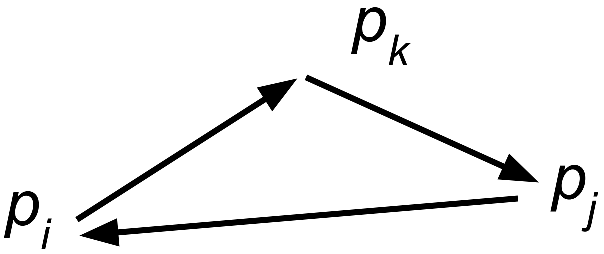
- Считаем, что каждая грань выпуклой оболочки - треугольник.
- В противном случае:
  - можно немного “отклонить” точки
  - в силу ограниченной точности вычислений такая ситуация маловероятна

Ориентация граней. Смотрим снаружи

- Грань face(p_i, p_j, p_k) = face(p_j, p_k, p_i) = face(p_k, p_i, p_j)



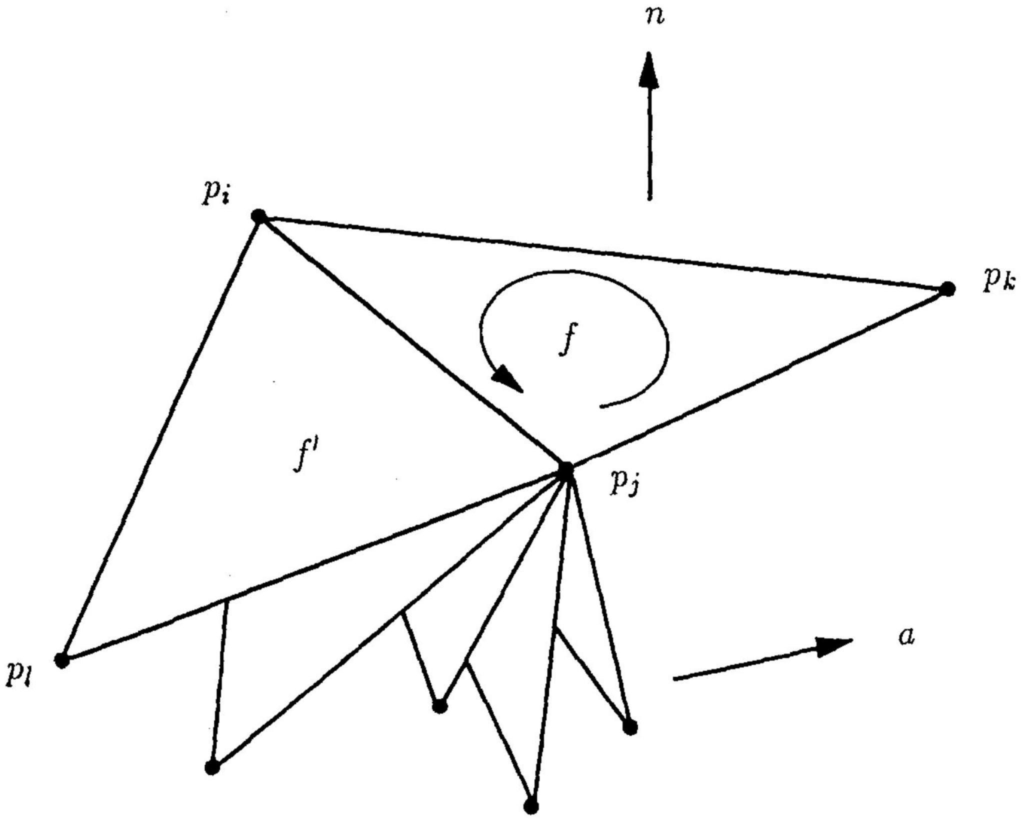
- Грань face(p_i, p_k, p_j) = face(p_k, p_j, p_i) = face(p_j, p_i, p_k)



Итерация алгоритма

- Обрабатываем грань f_1 = face(p_i, p_j, p_k):
- Ищем новую грань f_2, которая разделяет (в смысле “share”) ребро $p_i p_j$ с гранью f_1
    - Ребро уже есть ($p_i p_j$). Осталось найти одну точку
    - Ищем точку, максимизирующую угол между (см. рисунок)

Итерация алгоритма



$$\mathbf{a} = \frac{\mathbf{n} \times \mathbf{p_i p_j}}{|\mathbf{n} \times \mathbf{p_i p_j}|}.$$

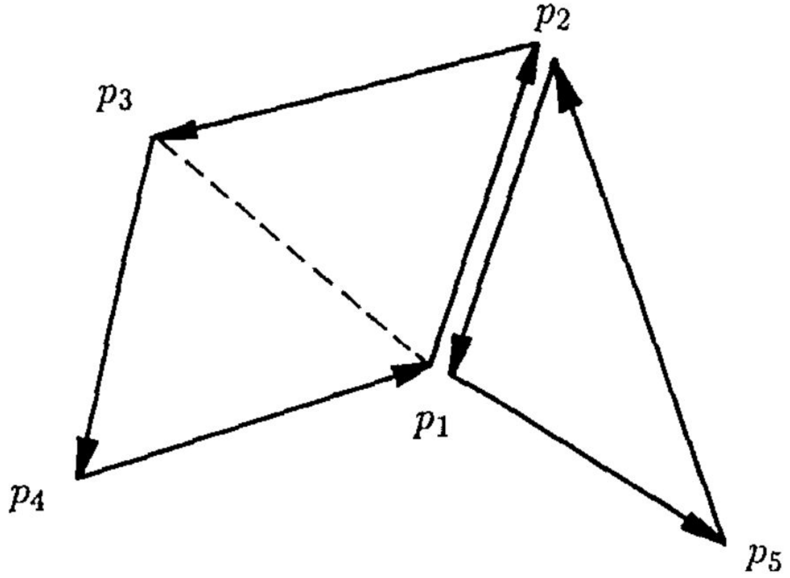
$$g(f, p_l) = \frac{\mathbf{a} \cdot \mathbf{p_i p_l}}{|\mathbf{a} \cdot \mathbf{p_i p_l}|}$$

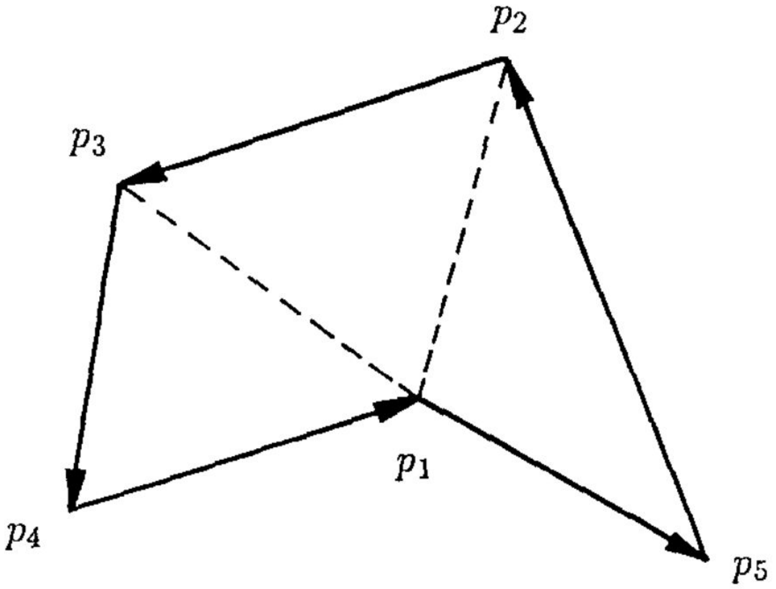
$g(f, p_l)$ -> min

Алгоритм

1. Заводим очередь граней ***Q*** для обработки
2. Поддерживаем контур ***E*** построенной оболочки
3. Достаем грань ***f*** из очереди.
   a. Для каждого ребра **e**, принадлежащего текущей грани ***f*** и контуру ***E:***
      i. находим новую грань ***f_new*** (пред. слайд), добавляем ее в очередь ***Q***
      ii. перестраиваем контур ***E*** в соответствии и ***f_new***
   b. Добавляем ***f*** в ответ.

Перестройка контура





Итог

- Время работы O(n * k), где k - число граней
- http://www.sciencedirect.com/science/article/pii/S002200000580056X

Алгоритм Чана (2D)

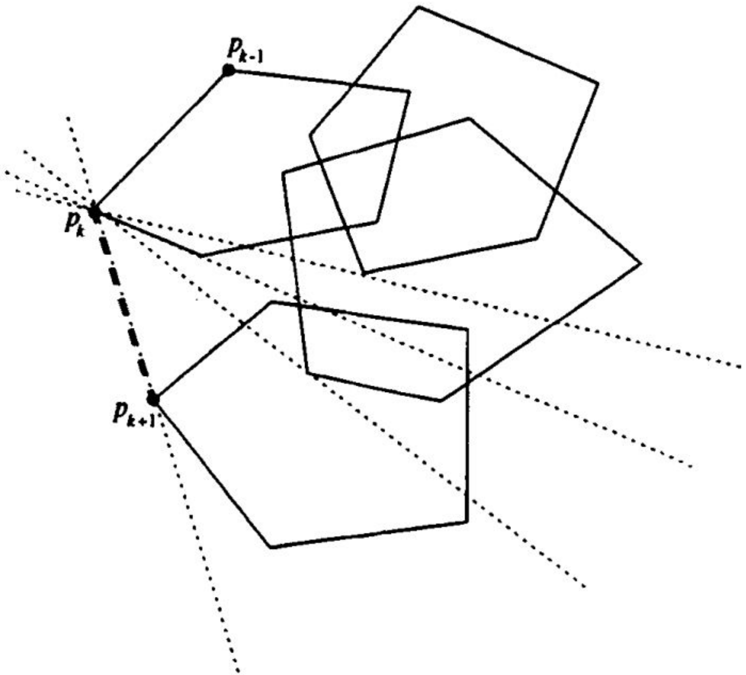
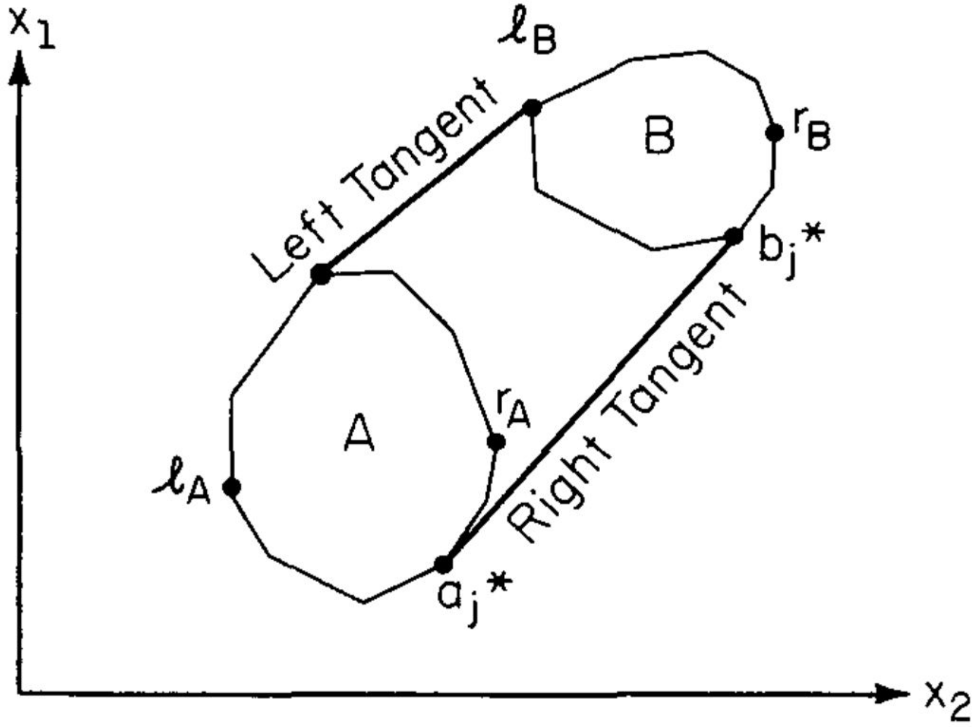


Fig. 1. Wrapping a set of $\lceil n/m \rceil$ convex polygons of size m.

Итог

- Время работы O(n * log(k)), где k - число вершин в.о.
- https://link.springer.com/content/pdf/10.1007/BF02712873.pdf
- https://www.youtube.com/watch?v=--PJ1dxuo-U

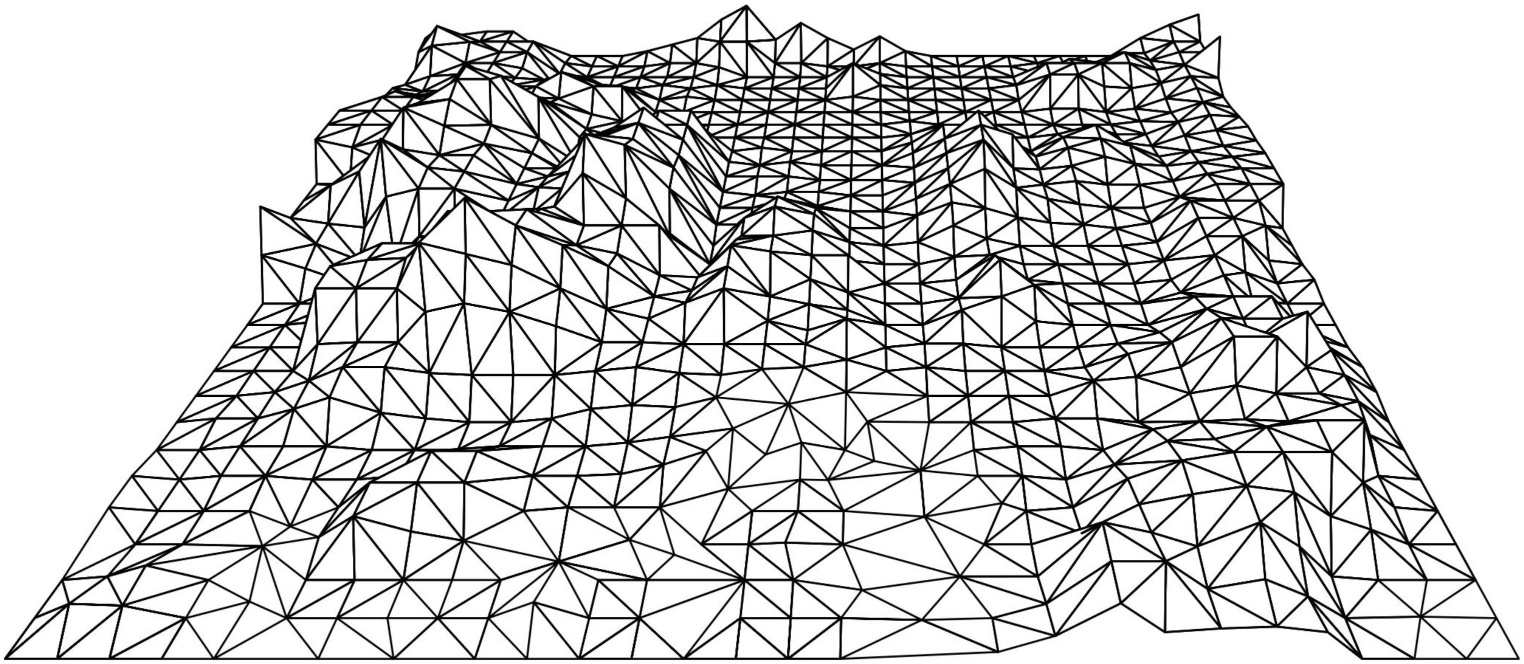
Алгоритм (F. P. Preparata and S. J. Hong)



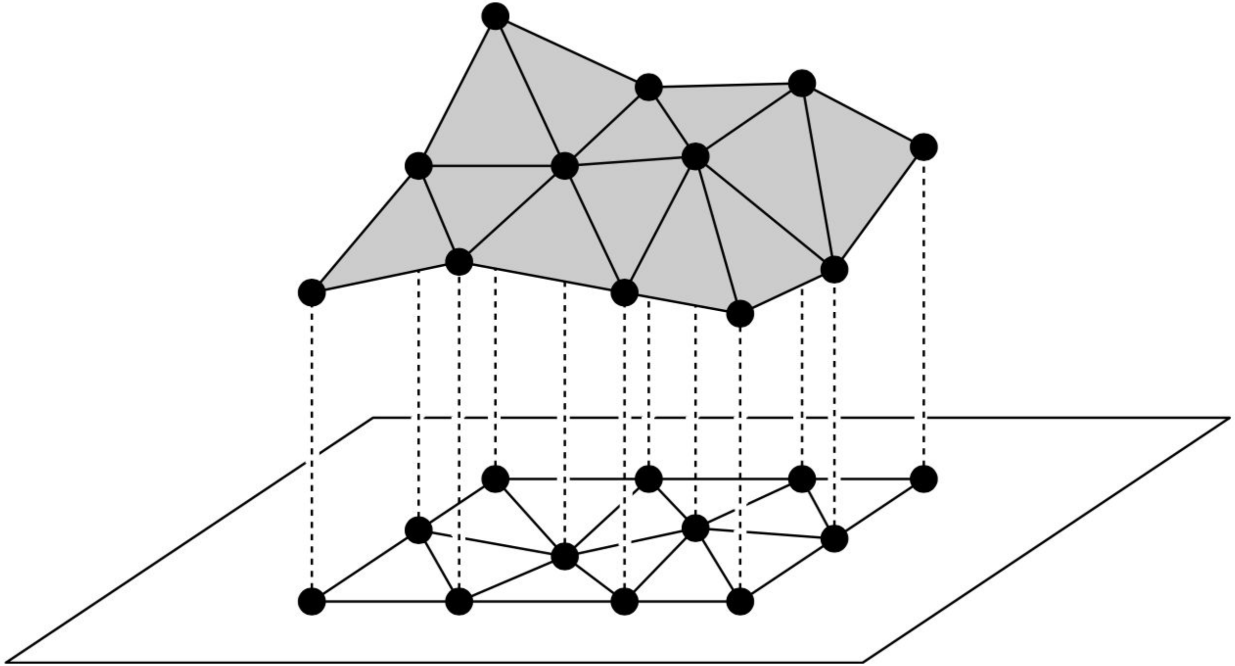
Итог

- Время работы O(n * log(n))
- https://link.springer.com/content/pdf/10.1007/BF02712873.pdf

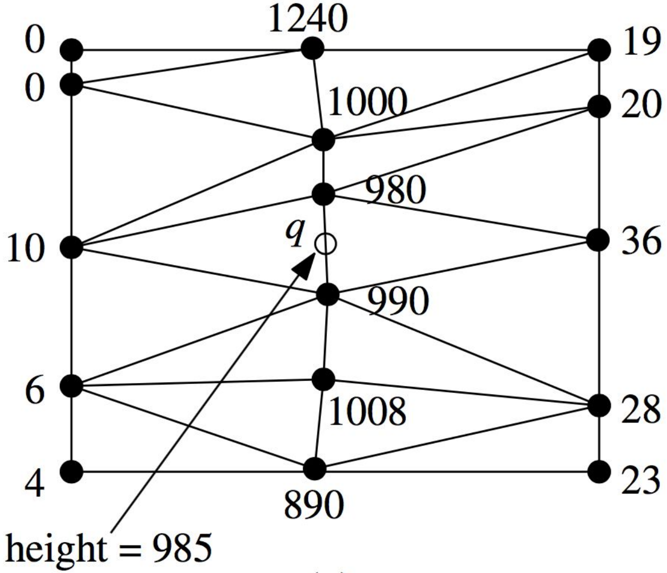
Триангуляция



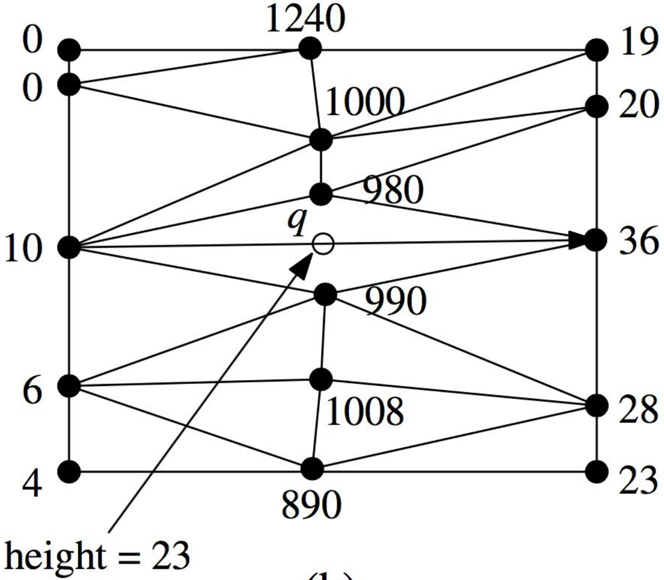
Триангуляция



Какая триангуляция является “хорошей”?

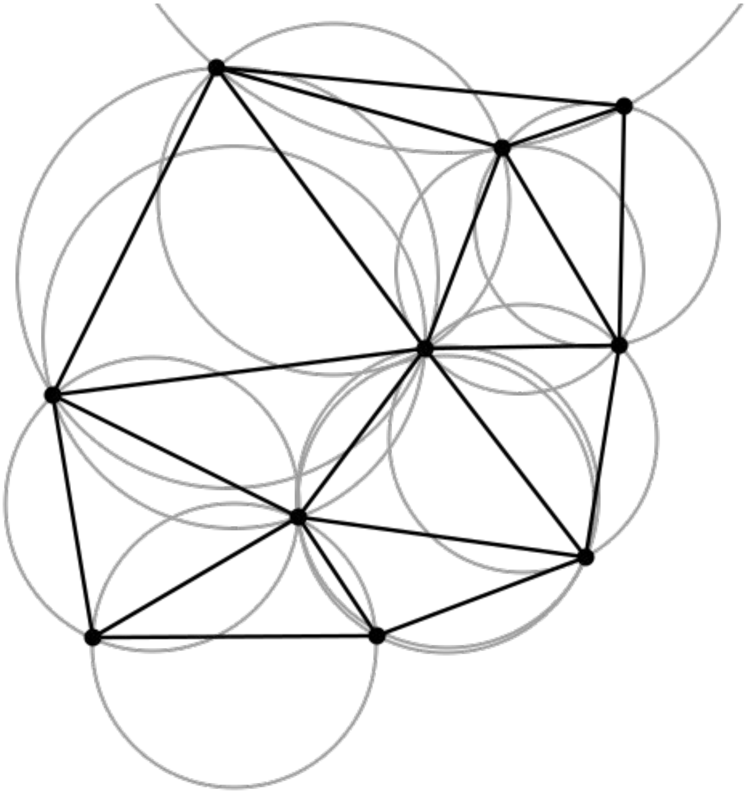


(a)

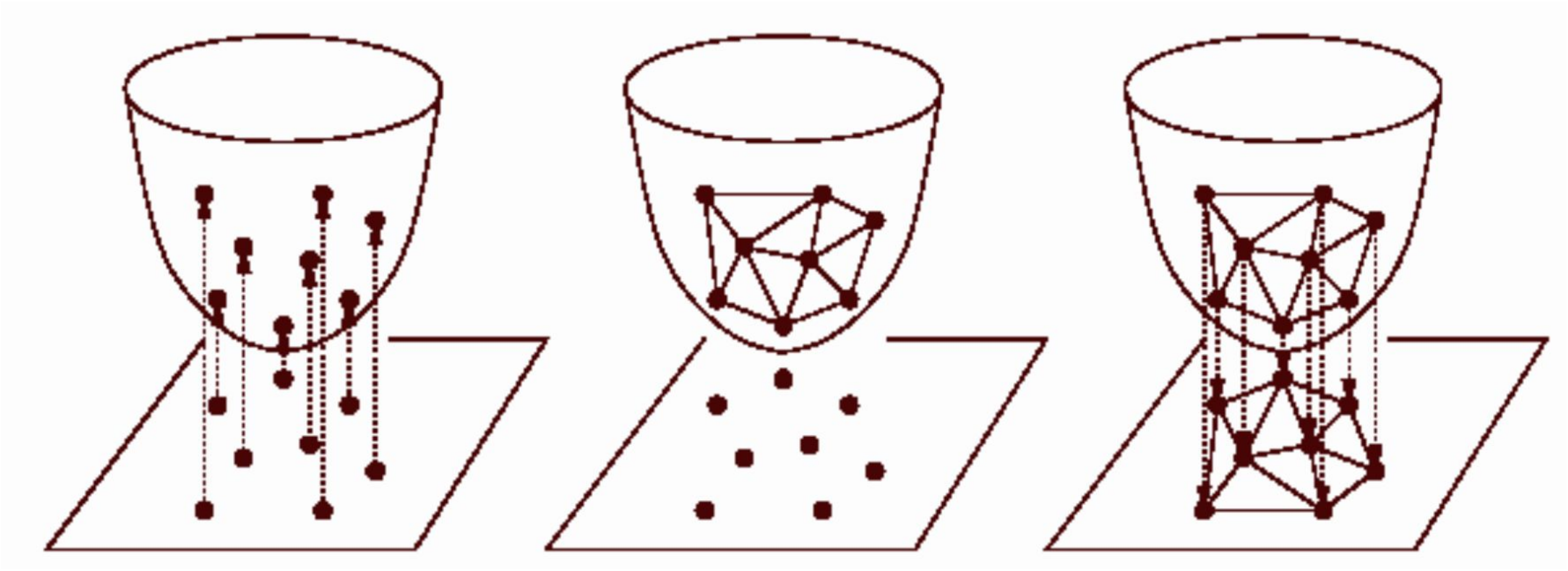


(b)

Триангуляция Делоне



Триангуляция Делоне



$(x, y) \to (x, y, x^2 + y^2)$ В.О. Проекция нижней части оболочки на (x,y)