МНОГОМЕРНЫЙ АЛГОРИТМ ОВЫПУКЛЕНИЯ РОЯ ТОЧЕК, НАХОДЯЩИХСЯ В НЕОБЩЕМ ПОЛОЖЕНИИ

Выполнил: студент гр. МЕНМ-280901 Корабельников А.А.
Научный руководитель: к.ф.-м.н., доцент Кумков С.С

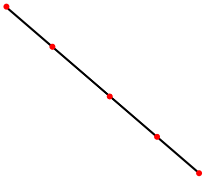
Институт естественных наук и математики

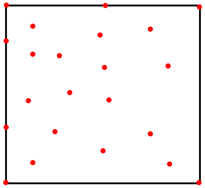
Екатеринбург, 2020

Постановка задачи

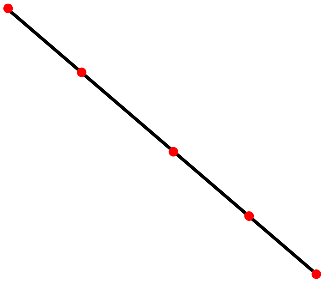
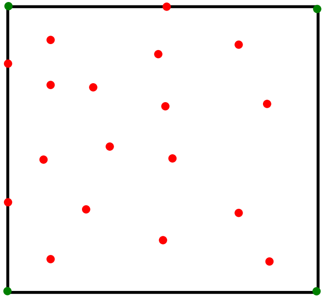
Требуется разработать алгоритм построения выпуклой оболочки многомерного роя точек, находящихся в необщем положении.

Необщее положение точек означает что в гиперплоскости евклидова пространства размерности n лежит больше чем $n+1$ точка.





Необщее положение точек

Проблемы:

- Требуется вычислять вершины ***(гипер)грани***,
- Требуется вычислять ***(гипер)рёбра*** грани.

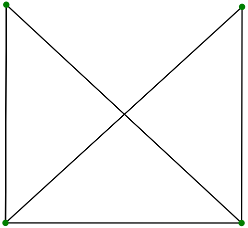
Мне не известны реализации алгоритмов овыпукления, работающих в многомерном пространстве в необщем положении.

Алгоритмы построения выпуклой оболочки в nD

Многие алгоритмы для случая плоскости имеют свои аналоги в 3D, но не в большей размерности. Библиотеки вычислительной геометрии:

- CGAL
- LEDA

Основная проблема алгоритмов, нацеленных на общее положение — несимплициальные грани.



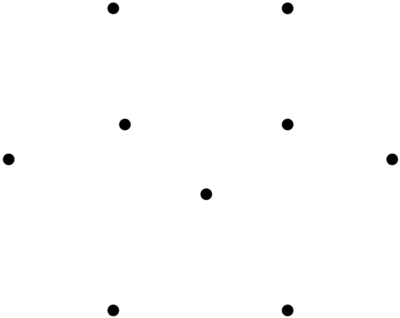
Алгоритмы овыпуклениия на плоскости

Существует множество алгоритмов овыпукления на плоскости:

- **Gift wrapping** — $O(nh)$
- Graham scan — $O(n \log n)$
- Quickhull — $O(n \log n)$
- Divide and conquer — $O(n \log n)$
- Monotone chain — $O(n \log n)$
- Chan's algorithm — $O(n \log n)$

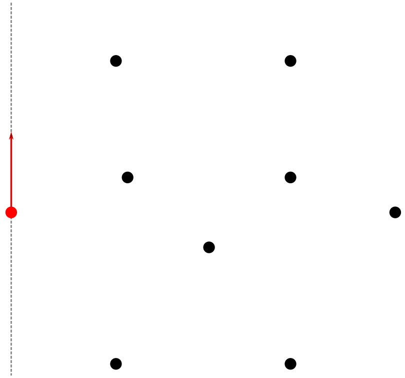
Для развития был взят алгоритм заворачивания подарка, т.к. он менее всего использует специфику плоскости.

Алгоритм Джарвиса на плоскости



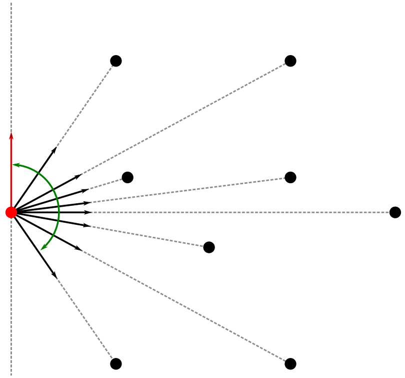
Найти минимальную точку. Провести через точку вертикальный вектор.

Алгоритм Джарвиса на плоскости



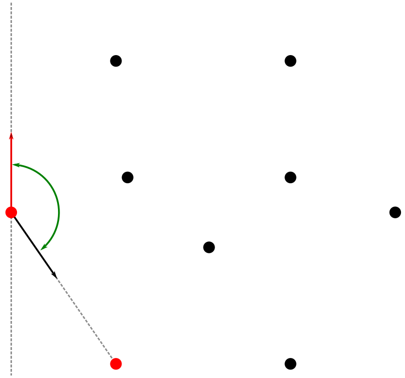
Поочередно проводим векторы к свободным точкам.

Алгоритм Джарвиса на плоскости



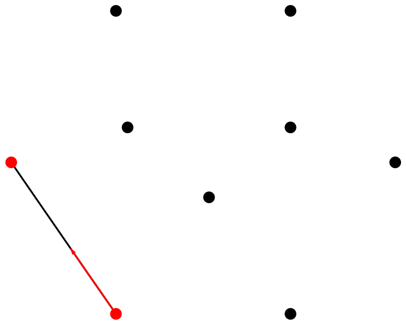
Берется точка, образующая максимальный угол, между векторами.

Алгоритм Джарвиса на плоскости



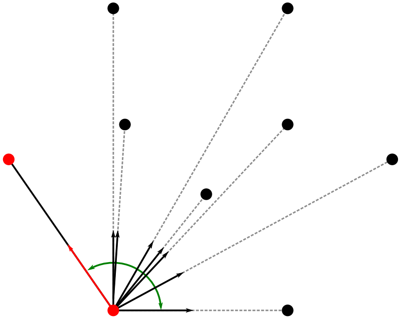
Начальная грань построена. Находим вектор ребра.

Алгоритм Джарвиса на плоскости



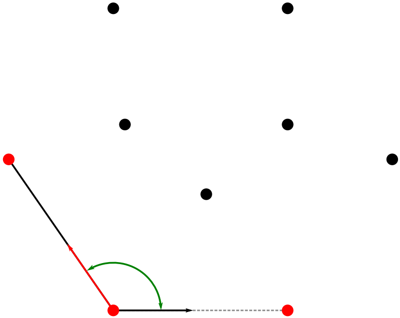
Построение следующего ребра.

Алгоритм Джарвиса на плоскости



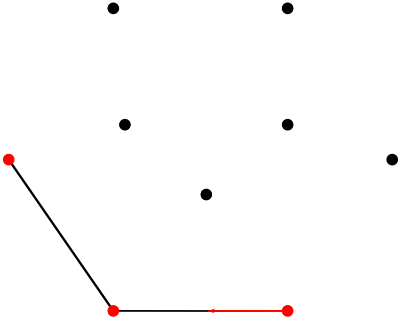
Построение следующего ребра.

Алгоритм Джарвиса на плоскости



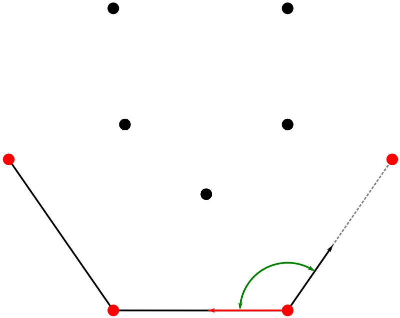
Построение следующего ребра.

Алгоритм Джарвиса на плоскости



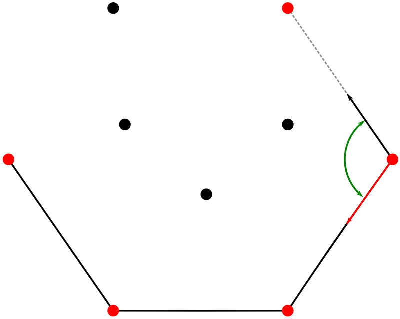
Построение следующего ребра.

Алгоритм Джарвиса на плоскости



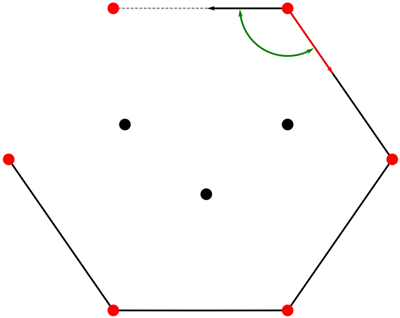
Построение следующего ребра.

Алгоритм Джарвиса на плоскости



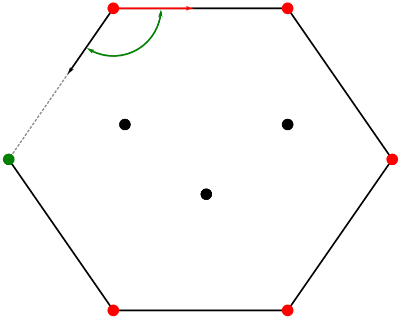
Построение следующего ребра.

Алгоритм Джарвиса на плоскости

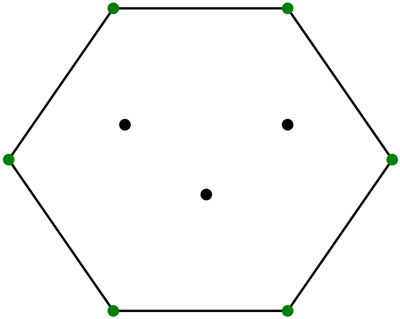


Построение следующего ребра.

Алгоритм Джарвиса на плоскости



Построение следующего ребра.

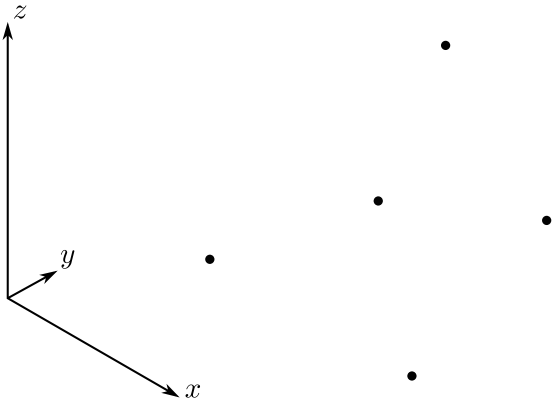


Конец построения

Проблемы расширения:

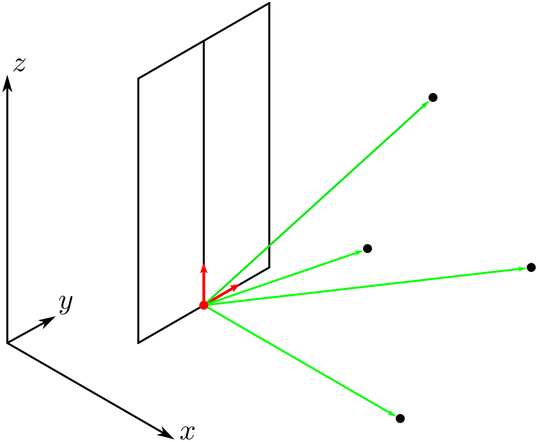
- поиск первой грани;
- обход граней.

Поиск первой грани



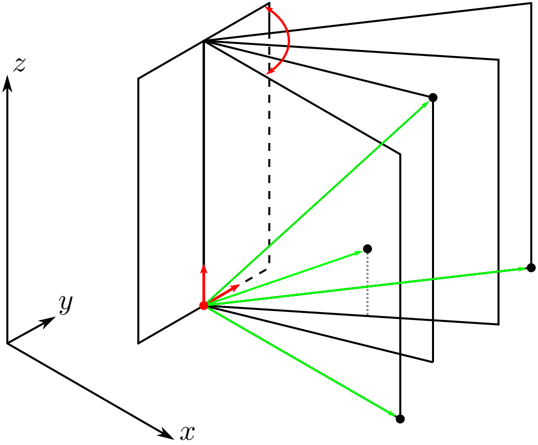
Находим минимальную точку.

Поиск первой грани



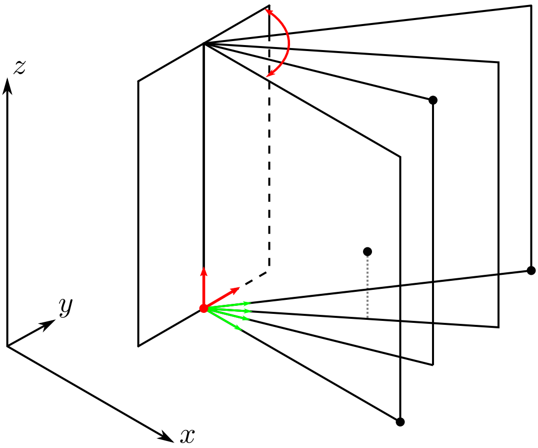
Формируем базис плоскости из векторов базиса пространства. Создаем базис подпространства, исключая первый вектор базиса плоскости.

Поиск первой грани



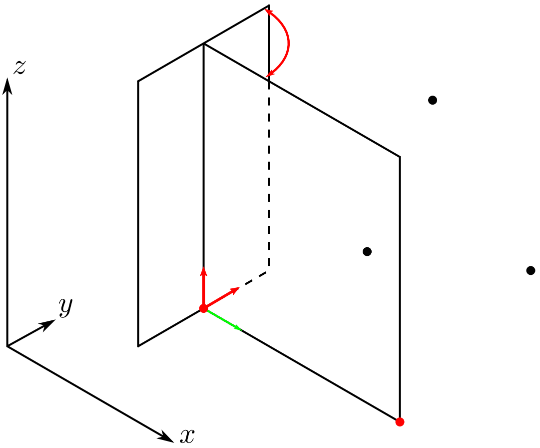
Поочередно проводим векторы к свободным точкам.

Поиск первой грани



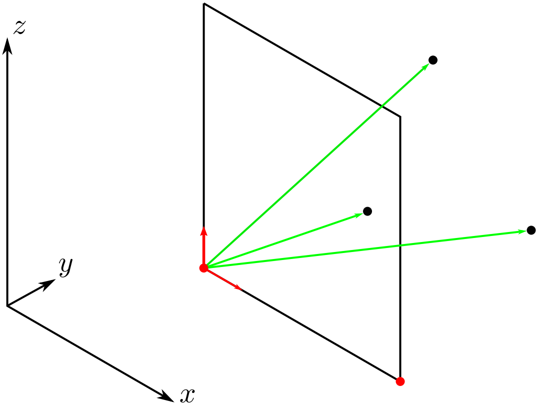
Ортонормируем эти векторы на фоне базиса подпространства
(шаг процедуры Грама–Шмидта)

Поиск первой грани



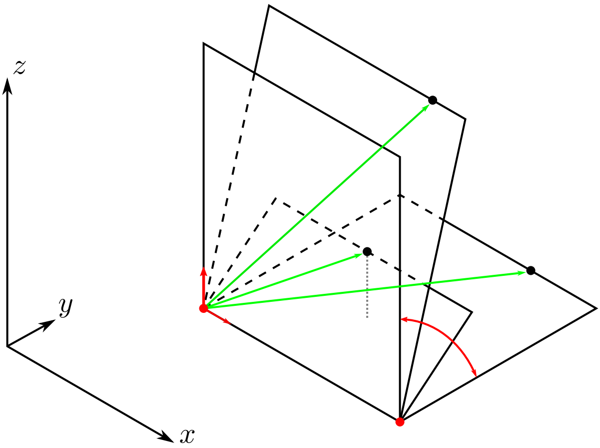
Выбираем вектор, образующий наибольший угол с первым вектором базиса. Заменяем первый вектор базиса. Запоминаем точку.

Поиск первой грани



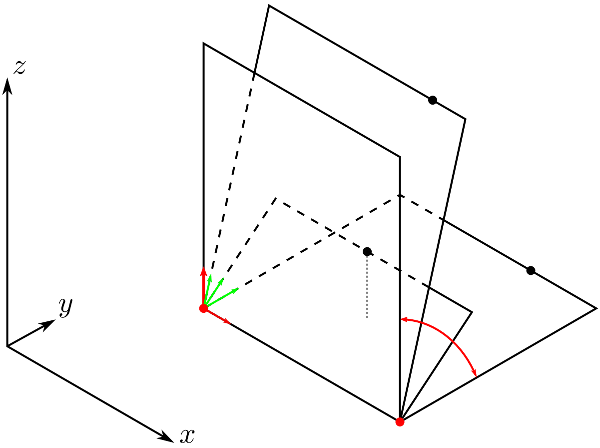
Создаем базис подпространства, исключая второй вектор базиса плоскости.

Поиск первой грани



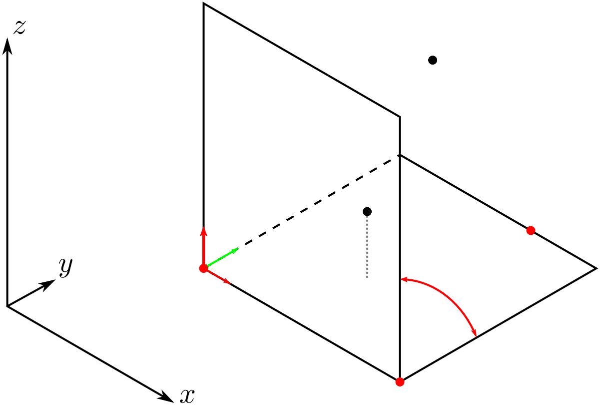
Поочередно проводим векторы к свободным точкам.

Поиск первой грани



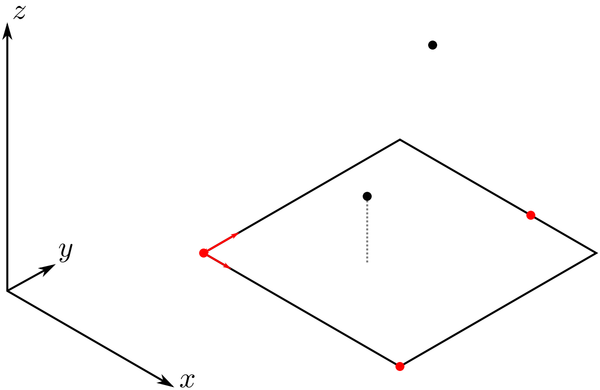
Ортонормируем эти векторы на фоне базиса подпространства (шаг процедуры Грама–Шмидта)

Поиск первой грани



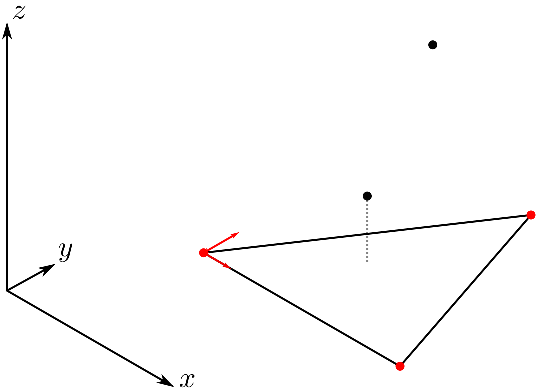
Выбираем вектор, образующий наибольший угол со вторым вектором базиса. Заменяем второй вектор базиса. Запоминаем точку.

Поиск первой грани



Выбираем вектор, образующий наибольший угол со вторым вектором базиса. Заменяем второй вектор базиса. Запоминаем точку.

Поиск первой грани



Начальная грань найдена. Грань добавляется в очередь на рассмотрение.

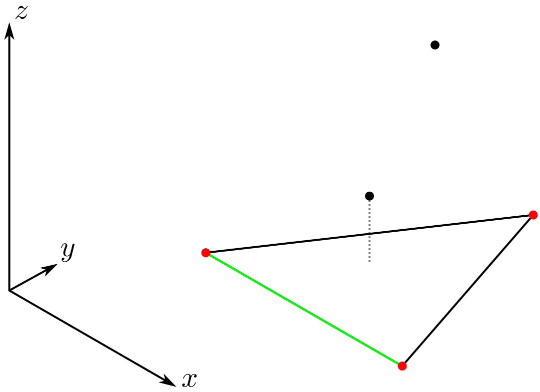
Обход граней

- Перебор ребер - обход в ширину.
- Хранение информации о найденных гранях - хеш-таблица.
- Хеш вычисляется на основе целых чисел, получаемых из коэффициентов уравнения плоскости грани.
  Возможны и другие алгоритмы хэширования граней: на основе точек вершин, на основе индексов точек вершин.

Порядок обхода

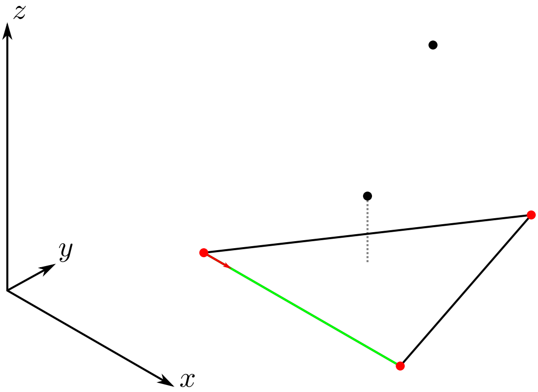
- берем необработанную грань;
- для каждого ребра выполняем переход на соседнюю грань;
- если найденная грань еще не обрабатывалась, добавляем в очередь.

Как происходит переход через ребро



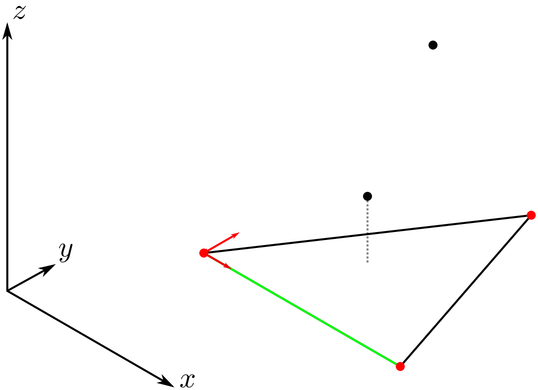
Берем очередное ребро грани

Как происходит переход через ребро



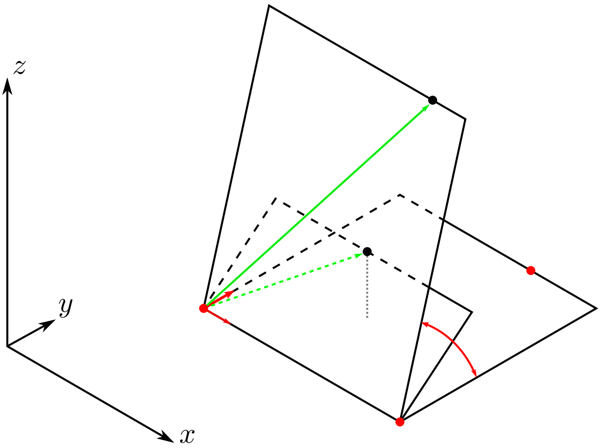
Находим базис ребра.

Как происходит переход через ребро

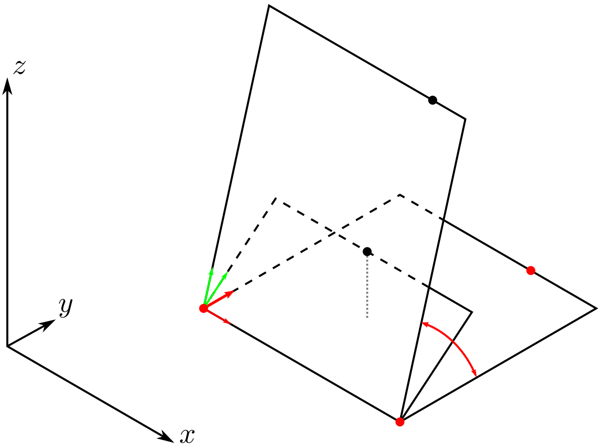


Проводим вектор к свободной точки грани и ортонормируем этот вектор на фоне базиса ребра.
(шаг процедуры Грама–Шмидта)

Как происходит переход через ребро

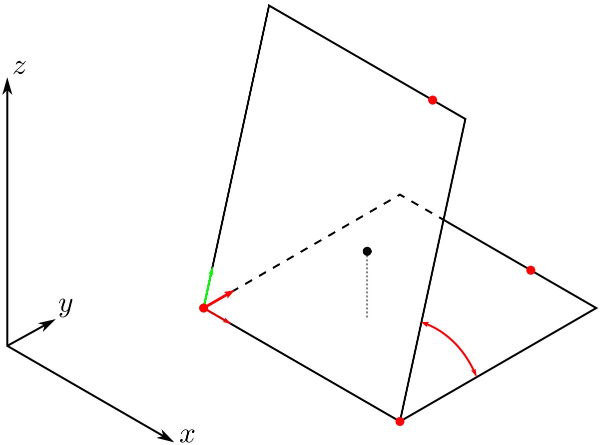


Поочередно проводим вектора к свободным точкам грани.



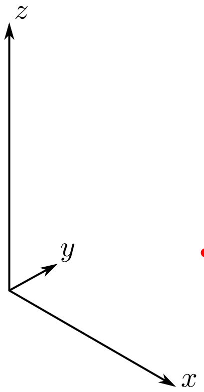
Ортонормируем эти векторы на фоне базиса ребра.

Как происходит переход через ребро



Берем плоскость, образующая максимальный угол между вектором грани и вектором свободной точки

Как происходит переход через ребро



Грань найдена. Если грань новая добавляем в очередь.
Переходим к рассмотрению следующего ребра.

Результат

Сложность — $O(n \cdot F \cdot d^2)$,
где F — количество граней, n — количество точек,
d^2 — порядок количества ребер у d-мерного симплекса.

Алгоритм Джарвиса при необщем положении точек

Проблемы расширения:

- построение грани;

Проблема построения грани

Проблемы построения:

- априори невозможно указать, какие из точек, попавших в плоскость грани, являются ее вершинами;
- грани выпуклой оболочки могут содержать разное количество ребер.

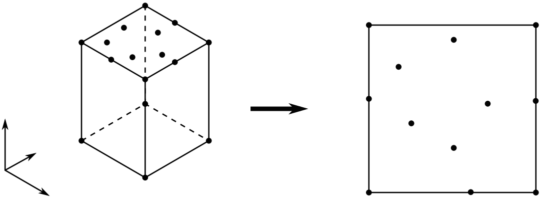
Проблема построения грани

Проблемы построения:

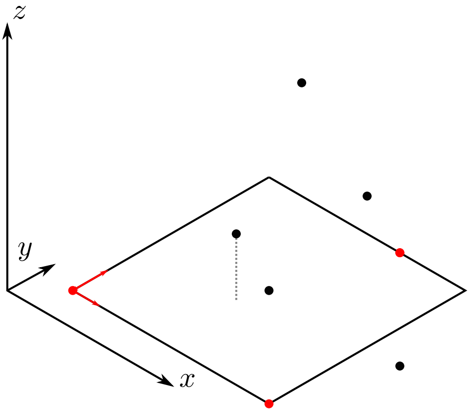
- априори невозможно указать, какие из точек, попавших в плоскость грани, являются ее вершинами;
- грани выпуклой оболочки могут содержать разное количество ребер.

Решение — уход в аффинное подпространство плоскости грани и построение в нем выпуклой оболочки роя точек, попавших в эту плоскость.

Отдельное рассмотрение случая двумерного аффинного подпространства.

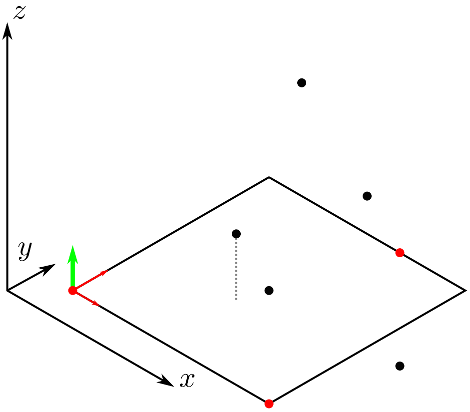


Поиск первой грани



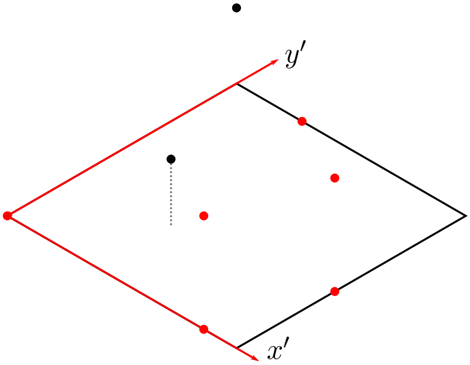
Находими плоскость грани

Поиск первой грани



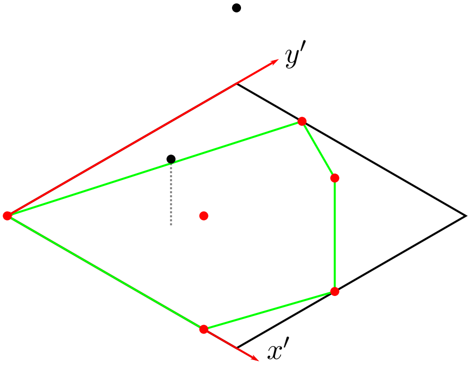
Вычисляем нормаль плоскости

Поиск первой грани



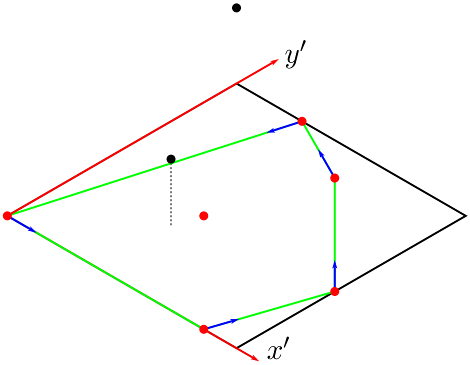
Находим точки и переводим в базис плоскости

Поиск первой грани

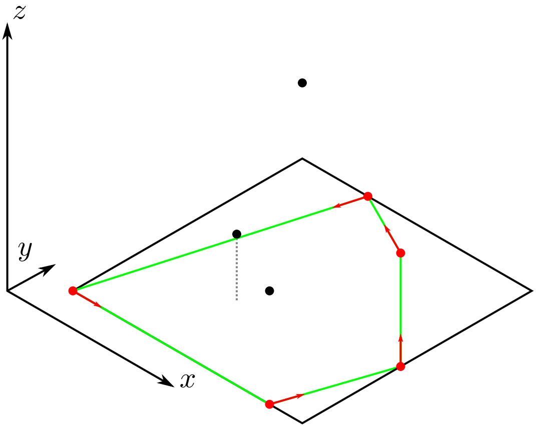


Строим выпуклую оболочку

Поиск первой грани



Если базисы ребер неизвестны, находим. Запоминаем базисы.



Заменяем точки выпуклой оболочки на исходные и пересчитываем базисные векторы ребер в координаты исходного пространства. Добавляем грань в очередь на обработку.

Алгорим Джарвиса при необщем положении точек

Проблемы расширения:

- построение грани;
- хранение выпуклой оболочки;

Хранение выпуклой оболочки

Необходимо хранить:

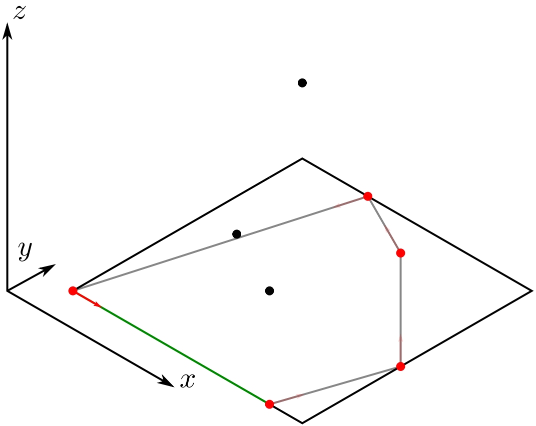
- Информацию о ребрах:
  - Если R^d $(d > 2)$, список ребер, которые могут быть многомерными несимплициальными многогранниками;
  - Если R^2, списки точек и базисов ребер;
- необходимо хранить список соседних граней;
- нужно хранить информацию о плоскости грани.
  - базис и начальную точку;
  - уравнение плоскости;

Алгорим Джарвиса при необщем положении точек

Проблемы расширения:

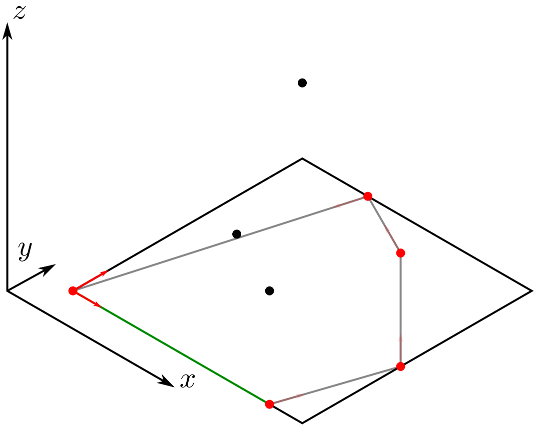
- построение грани;
- хранение выпуклой оболочки;
- обход ребер.

Переход через ребро



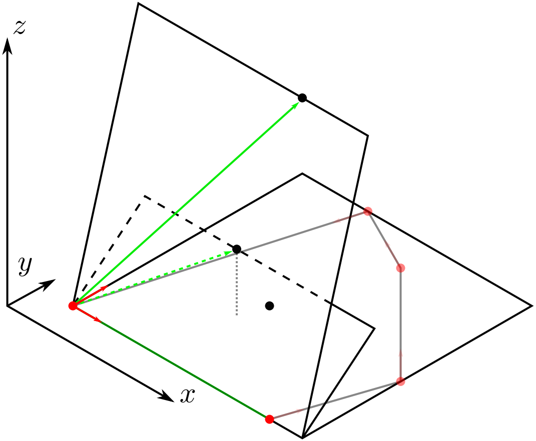
Пока очередь граней на обработку не пуста повторять.
Взять грань из очереди.

Переход через ребро



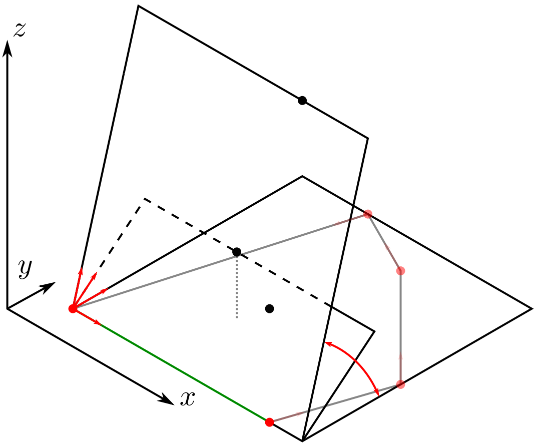
Взять очередное ребро обрабатываемой грани.

Переход через ребро



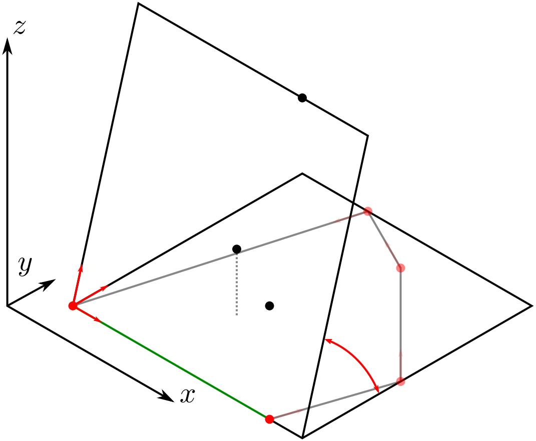
Поочередно провести векторы к свободным точкам.
Ортонормировать эти векторы на фоне базиса ребра
(шаг процедуры Грама–Шмидта)

Переход через ребро



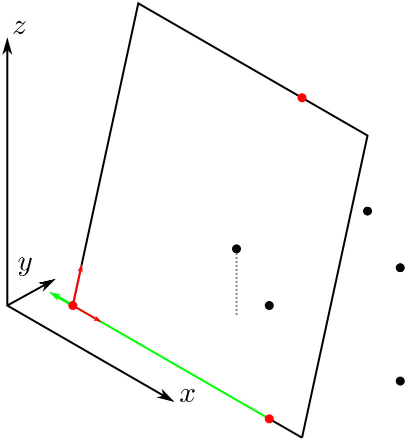
Найти вектор, образующий максимальный угол с вектором грани.

Переход через ребро



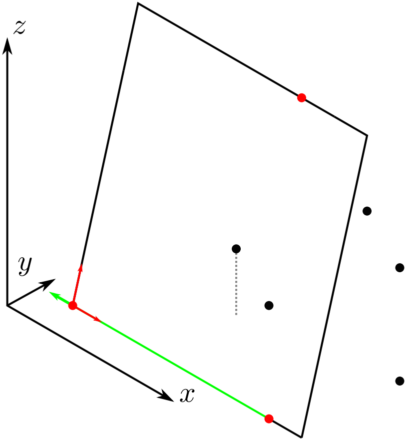
Плоскость грани найдена.

Переход через ребро



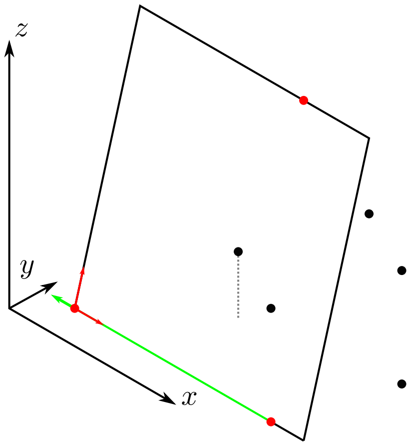
Определить нормаль плоскости. Построить хэш плоскости грани и проверить, обработана ли она уже.

Переход через ребро



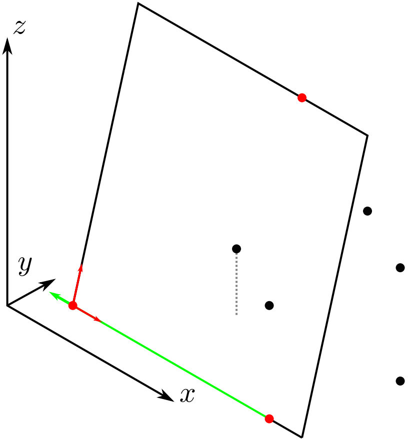
Если обработана, запомнить информацию о соседстве граней и перейти к следующему ребру обрабатываемой грани. Иначе продолжить обработку новой грани.

Переход через ребро



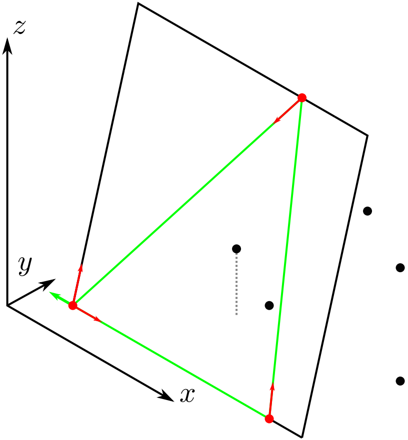
Найти точки, попавшие в плоскость грани.

Переход через ребро



Если их $d+1$ штука, то грань симплициальная и не требует особой обработки. Иначе запустить рекурсивно алгоритм овыпукления в аффинном подпространстве.

Переход через ребро



Грань построена. Запомнить информацию о соседстве.
Добавить в очередь на обработку.

Результат

Сложность — $O\big((hn)^{d-1}\big)$,
где h — количество ребер, n — количество точек,
d — размерность.

Тестовая реализация

- Платформа -.Net Core.
- Язык - C#.

Результаты тестов

- трехмерный тетраэдр
- четырехмерный тетраэдр
- четырехмерный гиперкуб
- трех мерный куб с точками

Спасибо за внимание!