

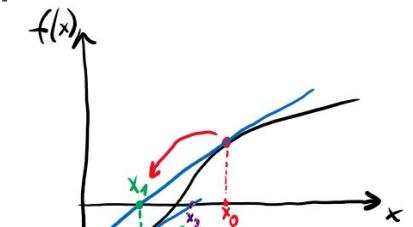
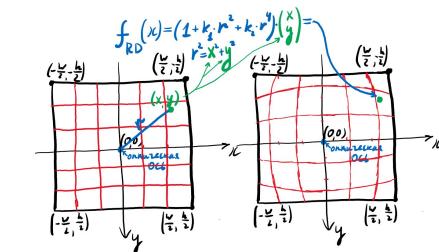
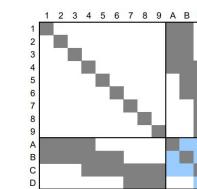
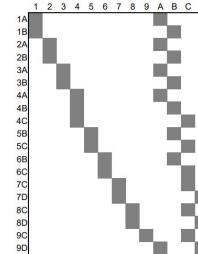
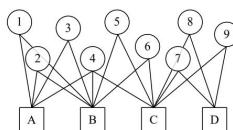
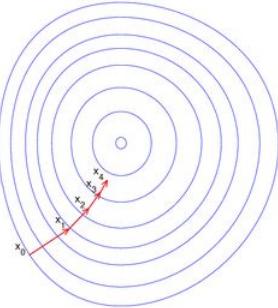
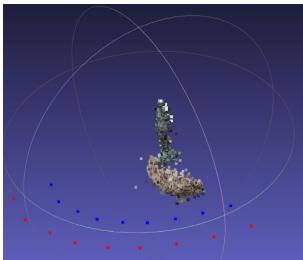
Введение в фотограмметрию

Уточнение параметров камер

Bundle Adjustment

Фотограмметрия. Лекция 9

- Методы оптимизации
- Градиентный спуск
- Метод Ньютона, Гаусс-Ньютона
- Метод Левенберга - Марквардта
- Bundle Adjustment



Полярный Николай
polarnick239@gmail.com

Мотивация

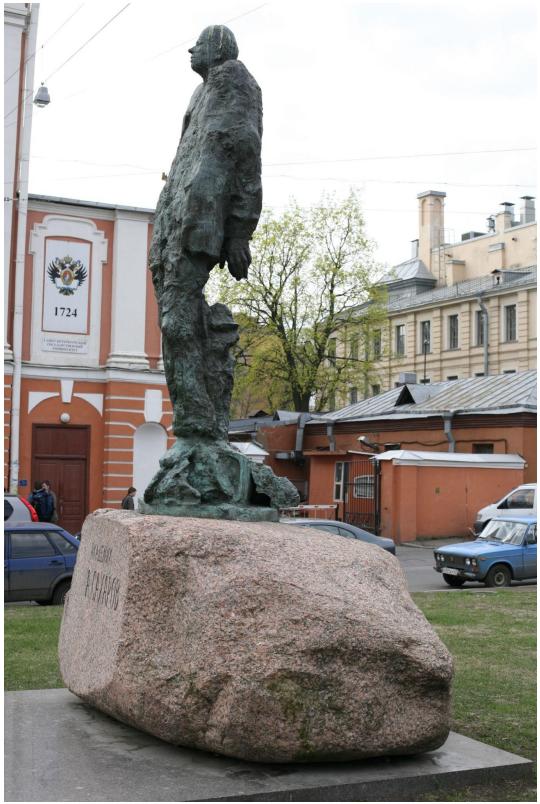
- 1) Научились с помощью решения линейных систем находить облака точек и положения камер

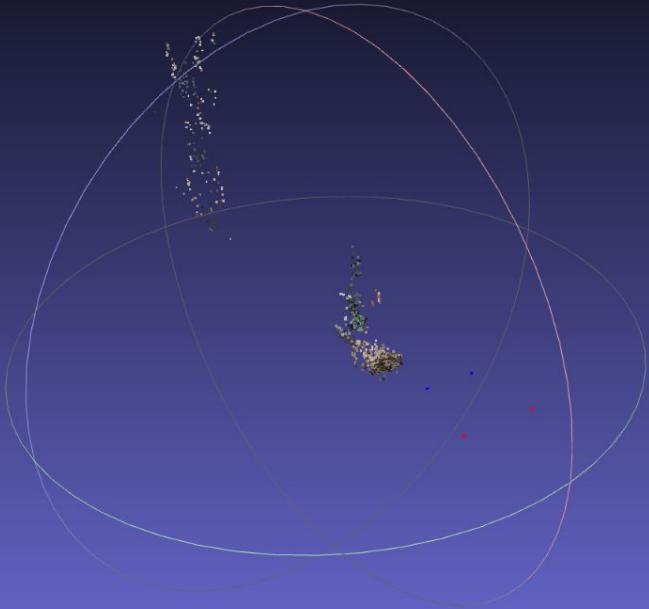
Мотивация

- 1) Научились с помощью решения линейных систем находить облака точек и положения камер
- 2) Из-за накопления ошибки и незнания калибровочных параметров камеры сцена очень быстро “разваливается” (не можем выполнить резекцию для новой камеры и триангулировать новые связующие точки)

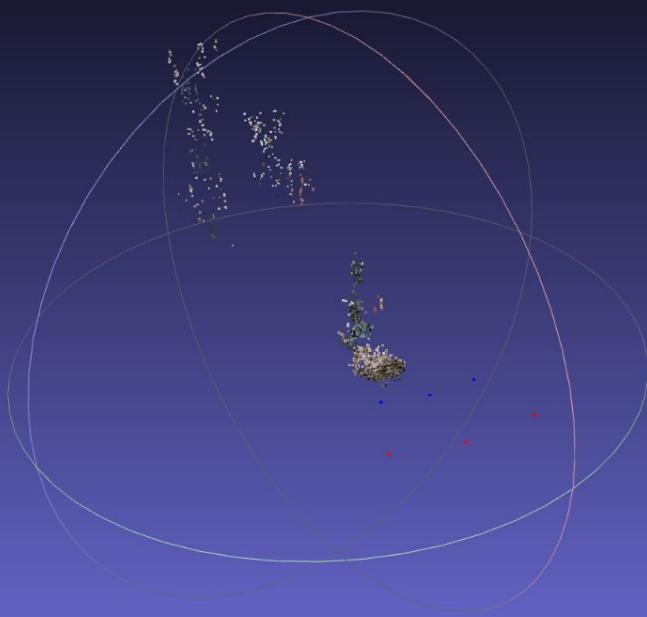
Мотивация

- 1) Научились с помощью решения линейных систем находить облака точек и положения камер
- 2) Из-за накопления ошибки и незнания калибровочных параметров камеры сцена очень быстро “разваливается” (не можем выполнить резекцию для новой камеры и триангулировать новые связующие точки)
- 3) Хотим сбрасывать накапливающуюся ошибку по мере добавления новых камер

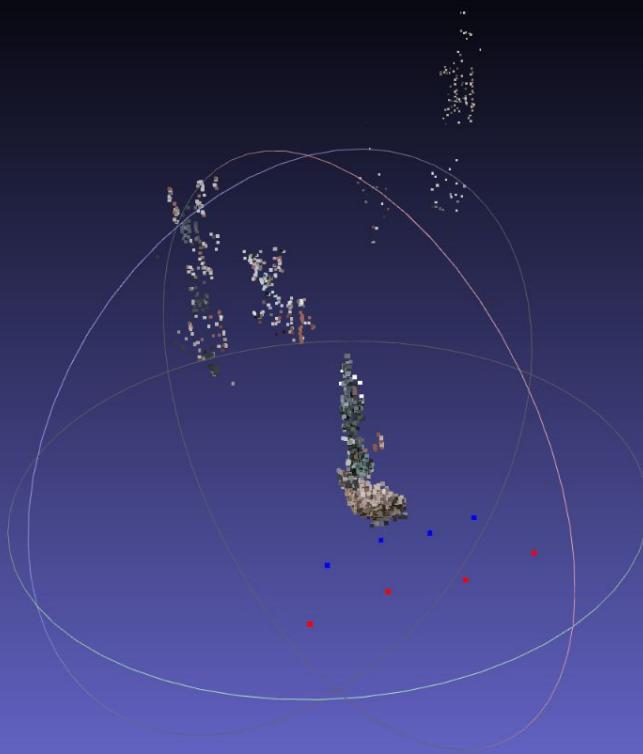




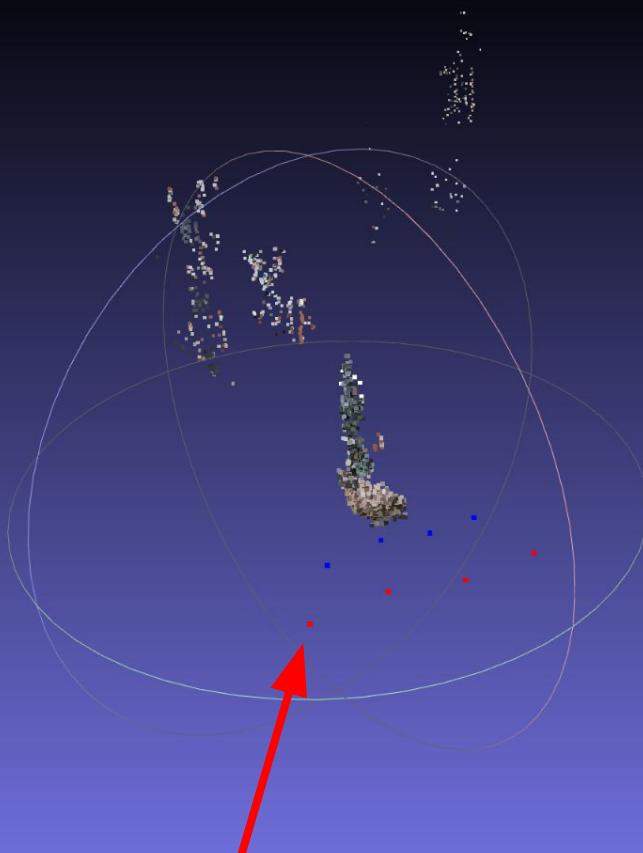
Триангуляция + резекция



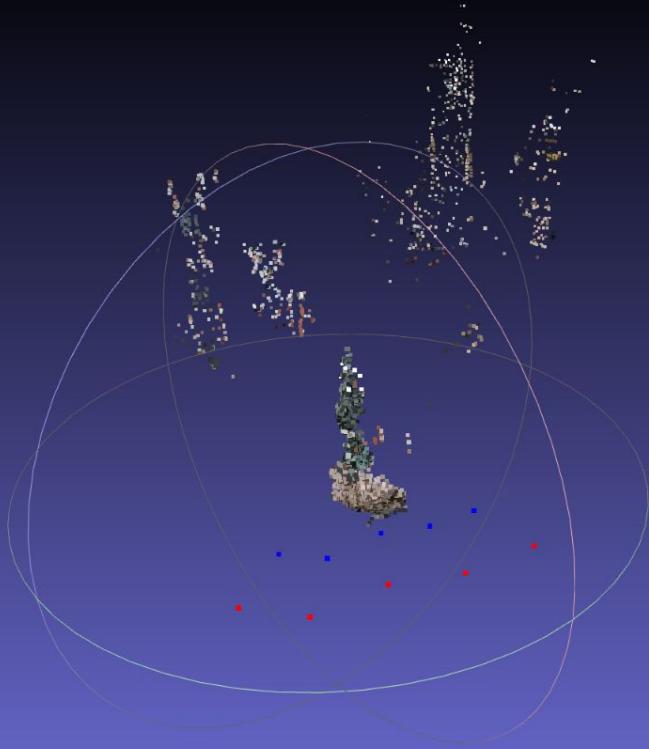
Триангуляция + резекция



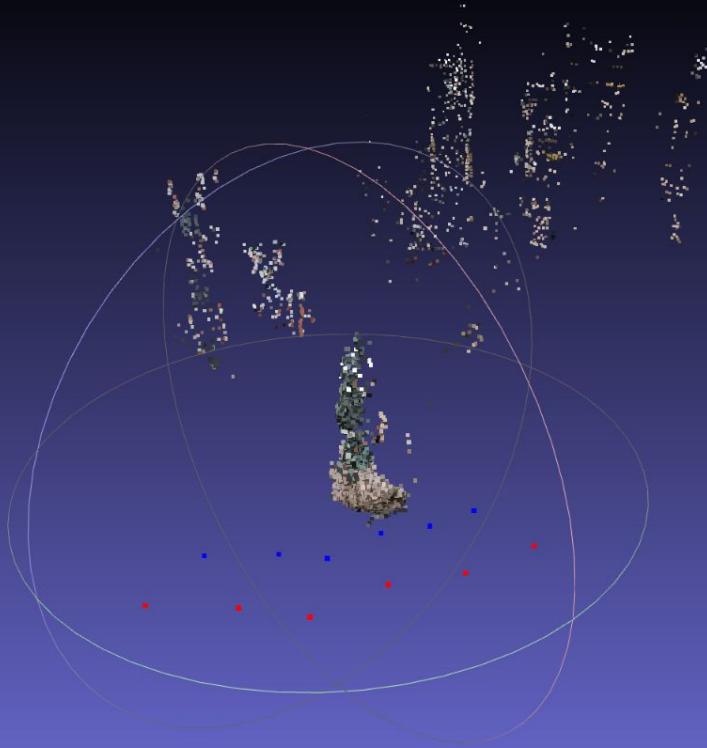
Триангуляция + резекция



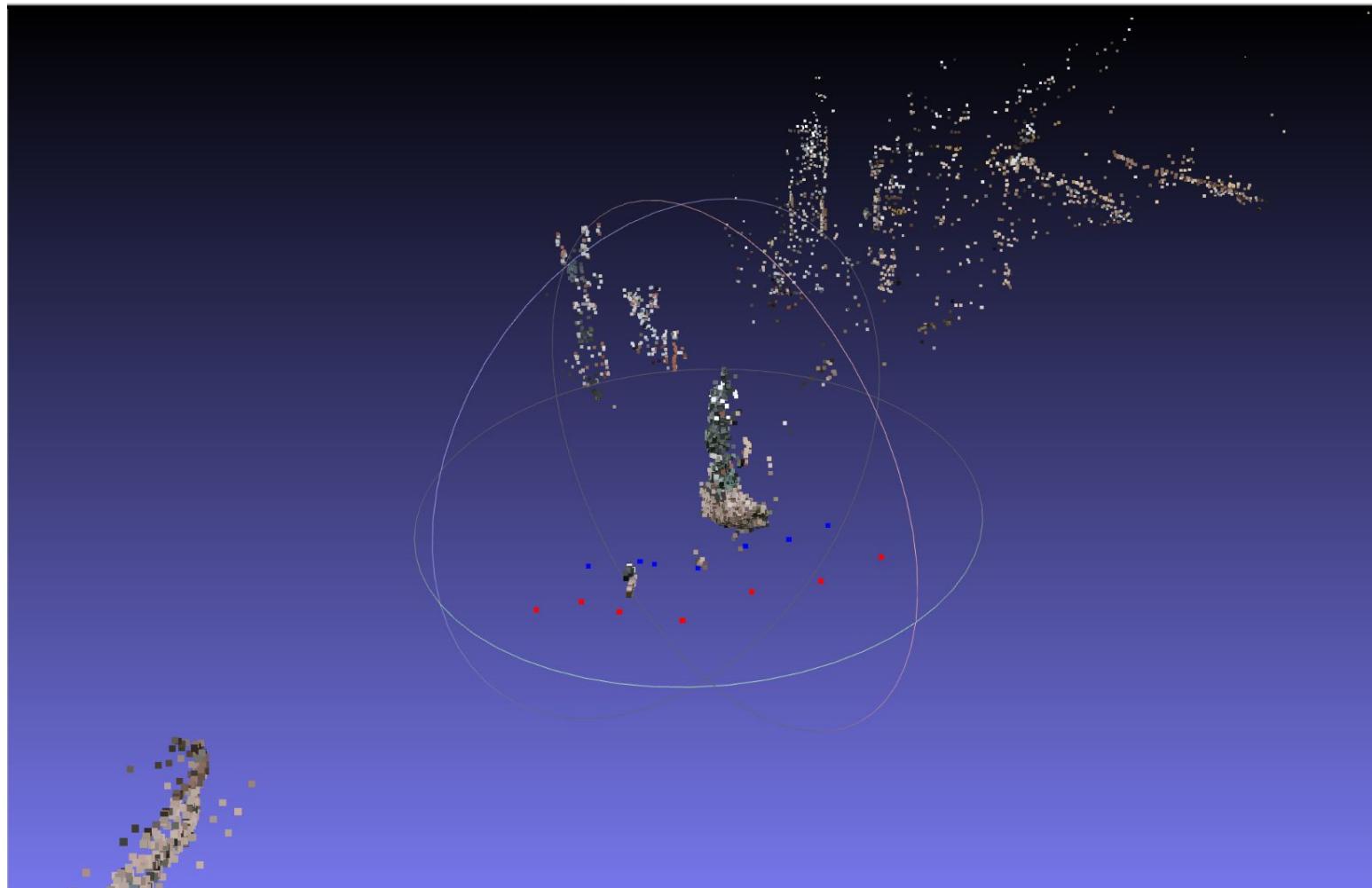
Триангуляция + резекция



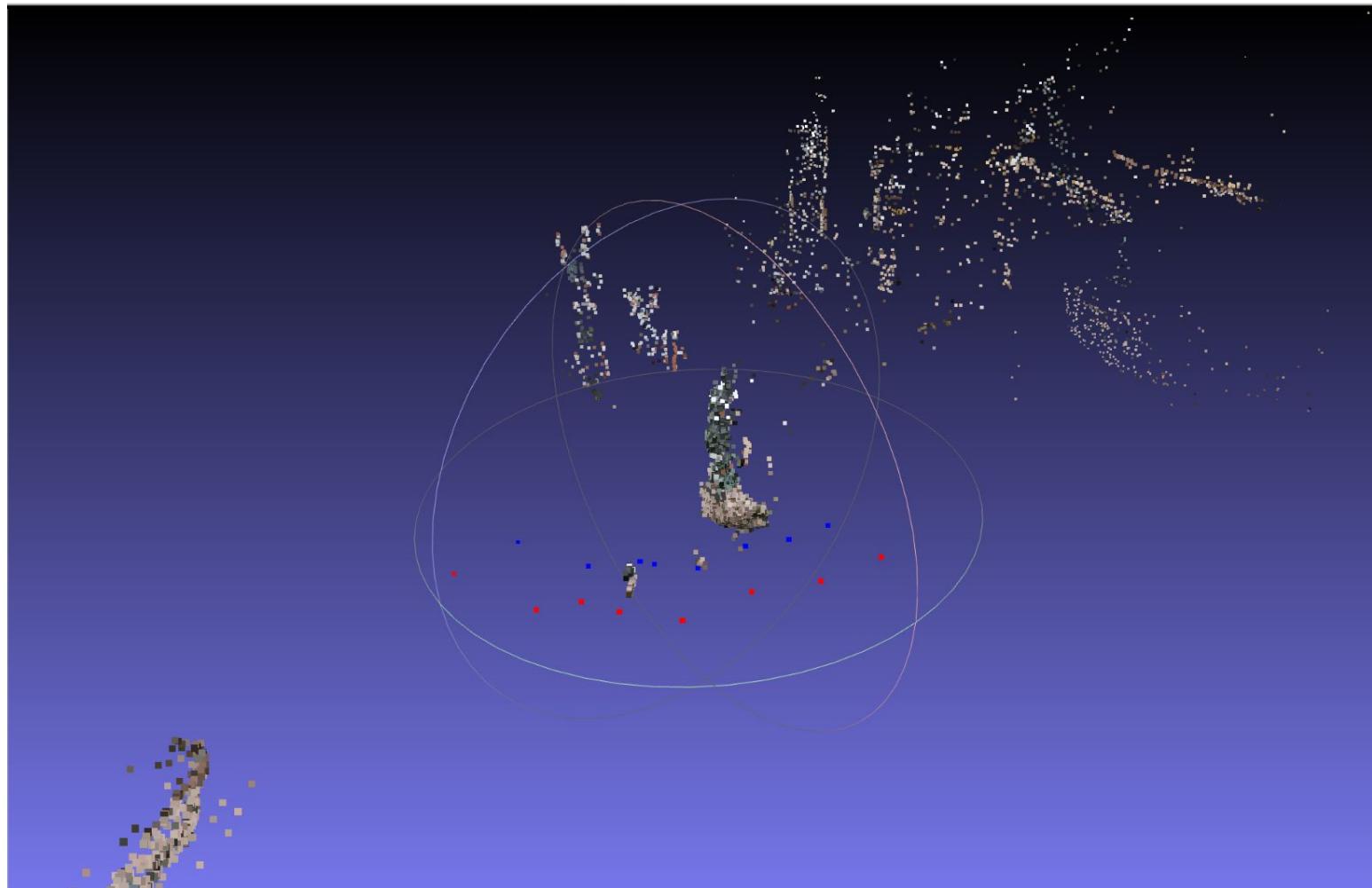
Триангуляция + резекция



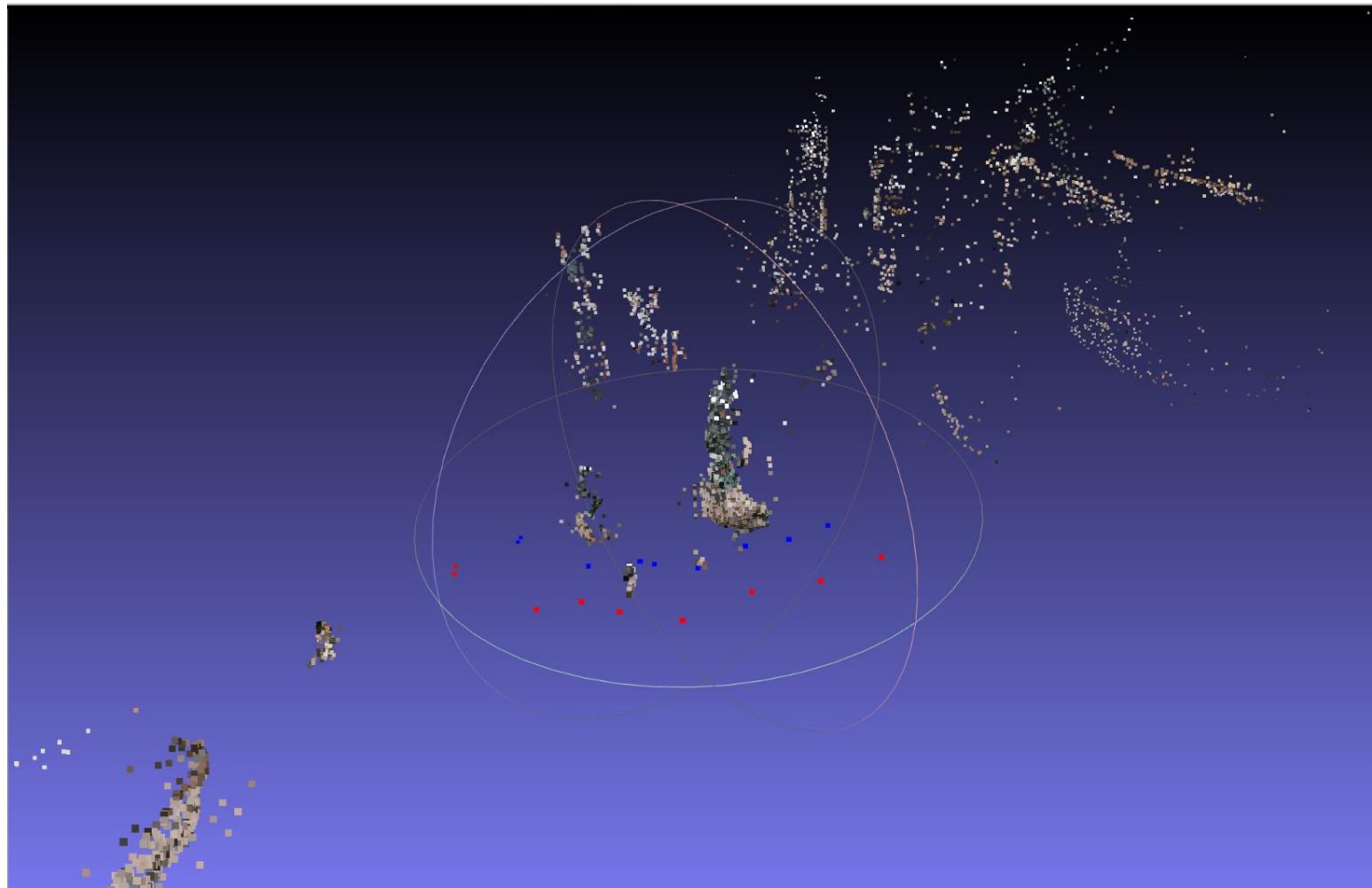
Триангуляция + резекция



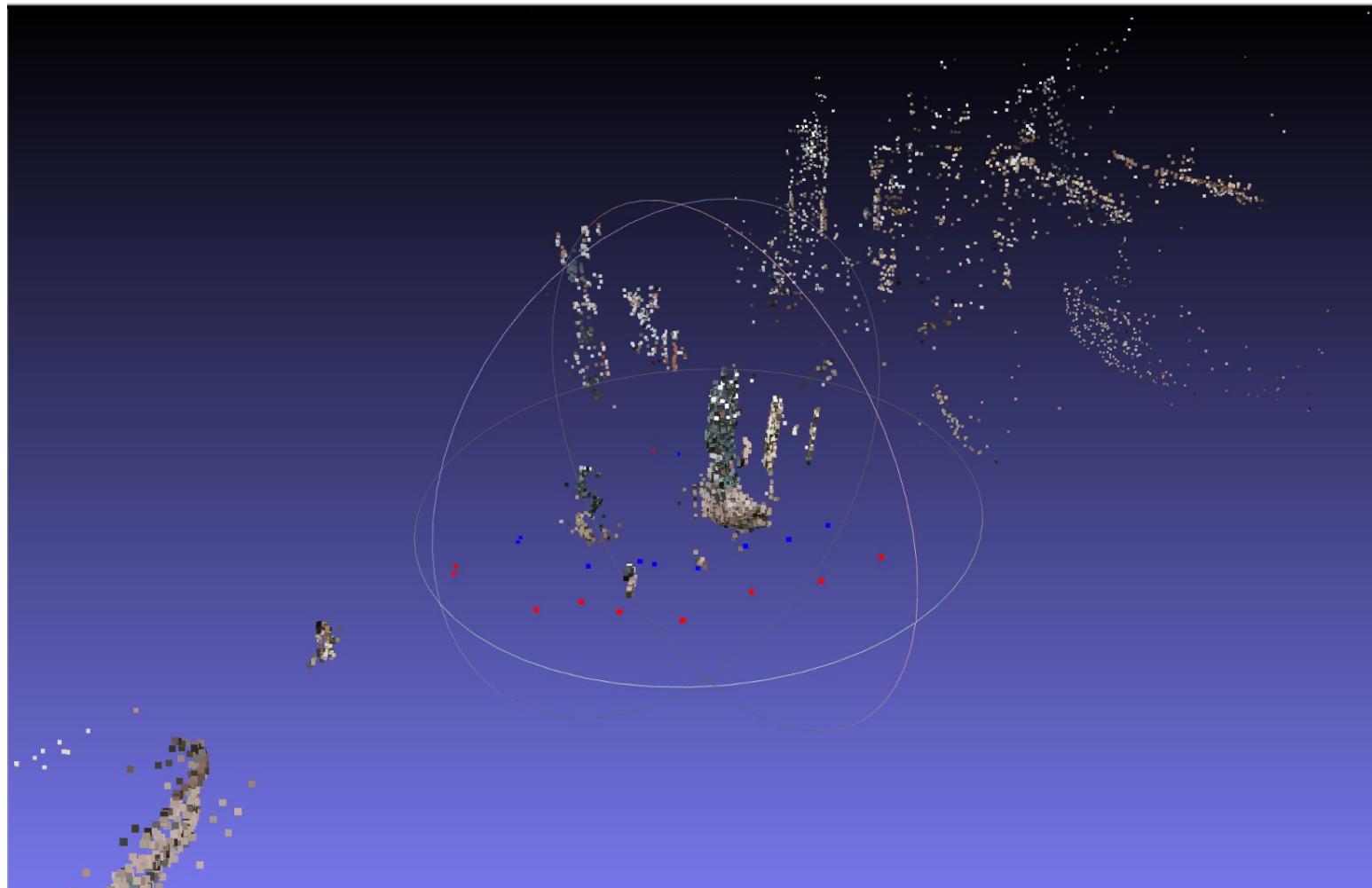
Триангуляция + резекция



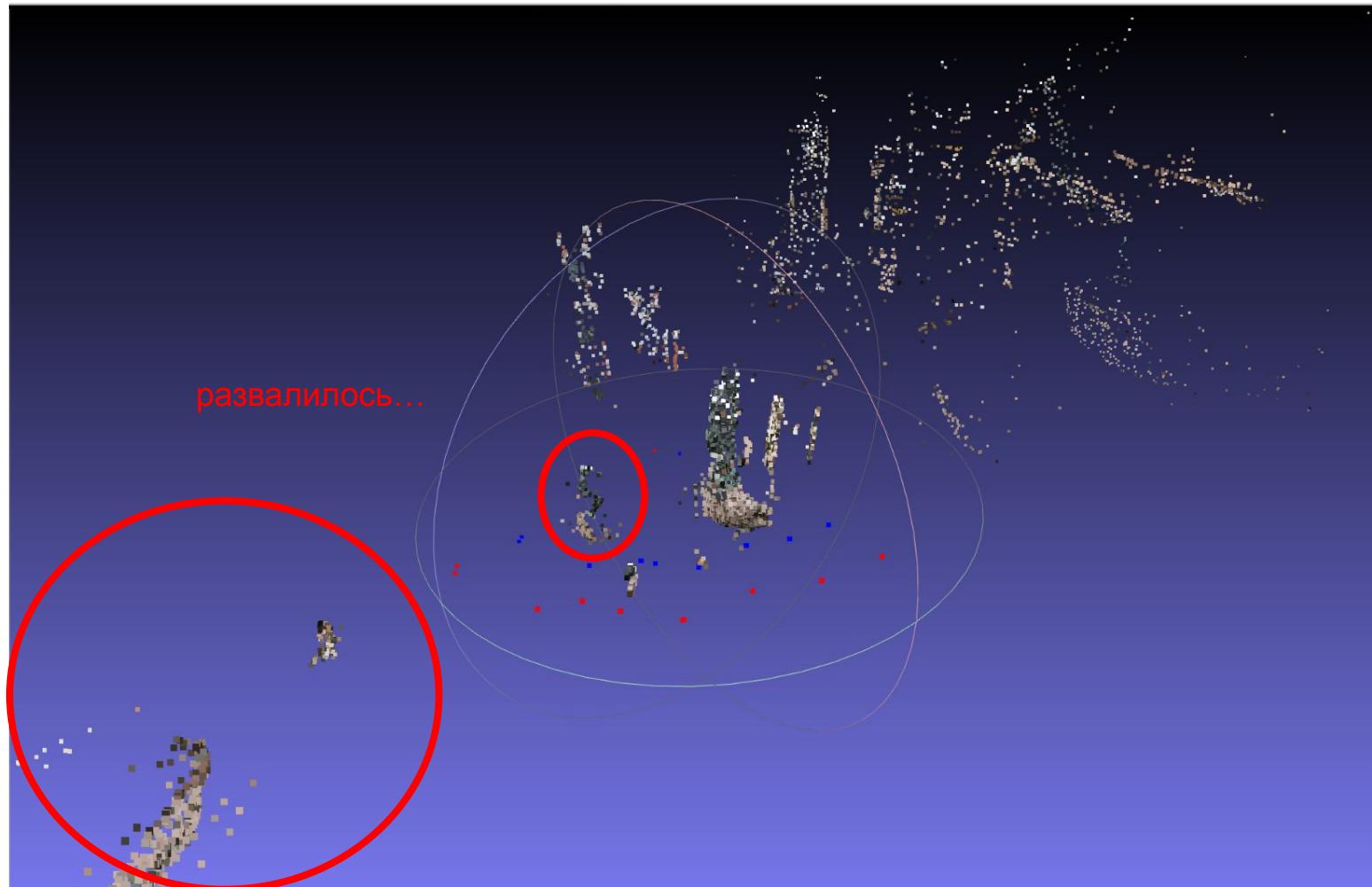
Триангуляция + резекция



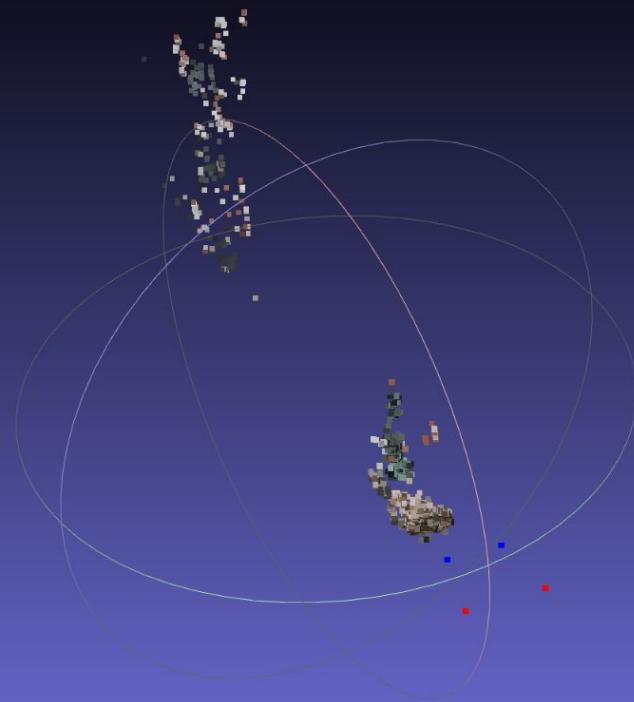
Триангуляция + резекция



Триангуляция + резекция

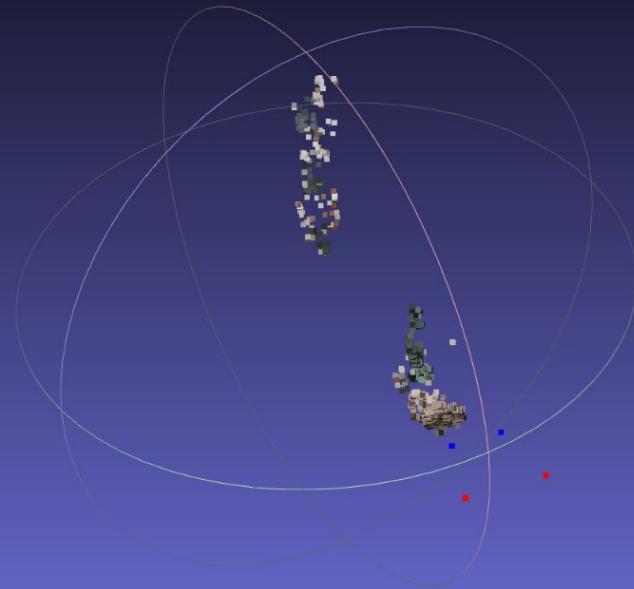


Триангуляция + резекция

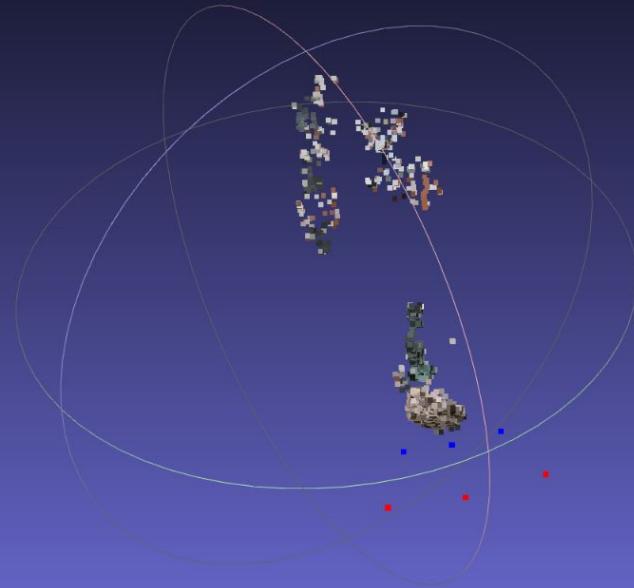


Триангуляция, резекция + уточнение положения камер и точек

BA

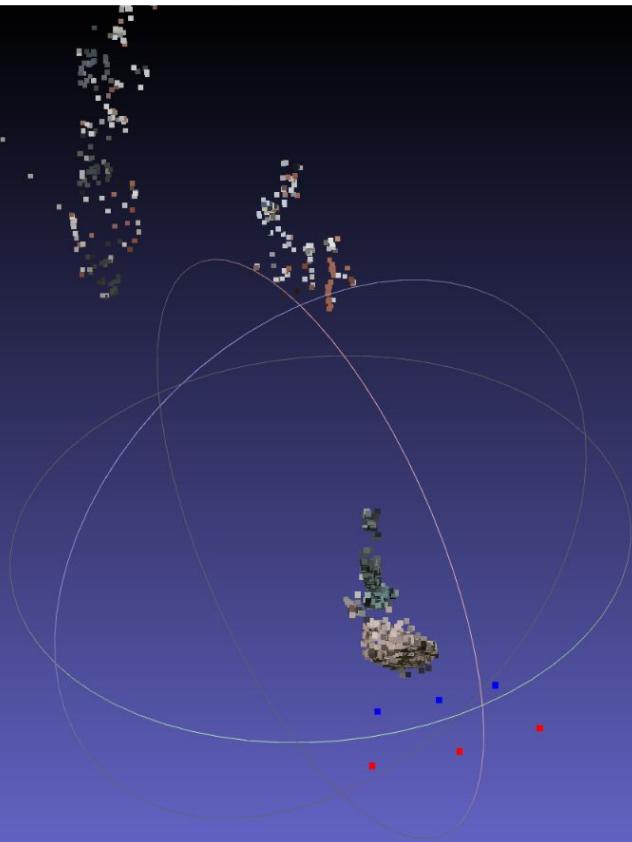


Триангуляция, резекция + уточнение положения камер и точек

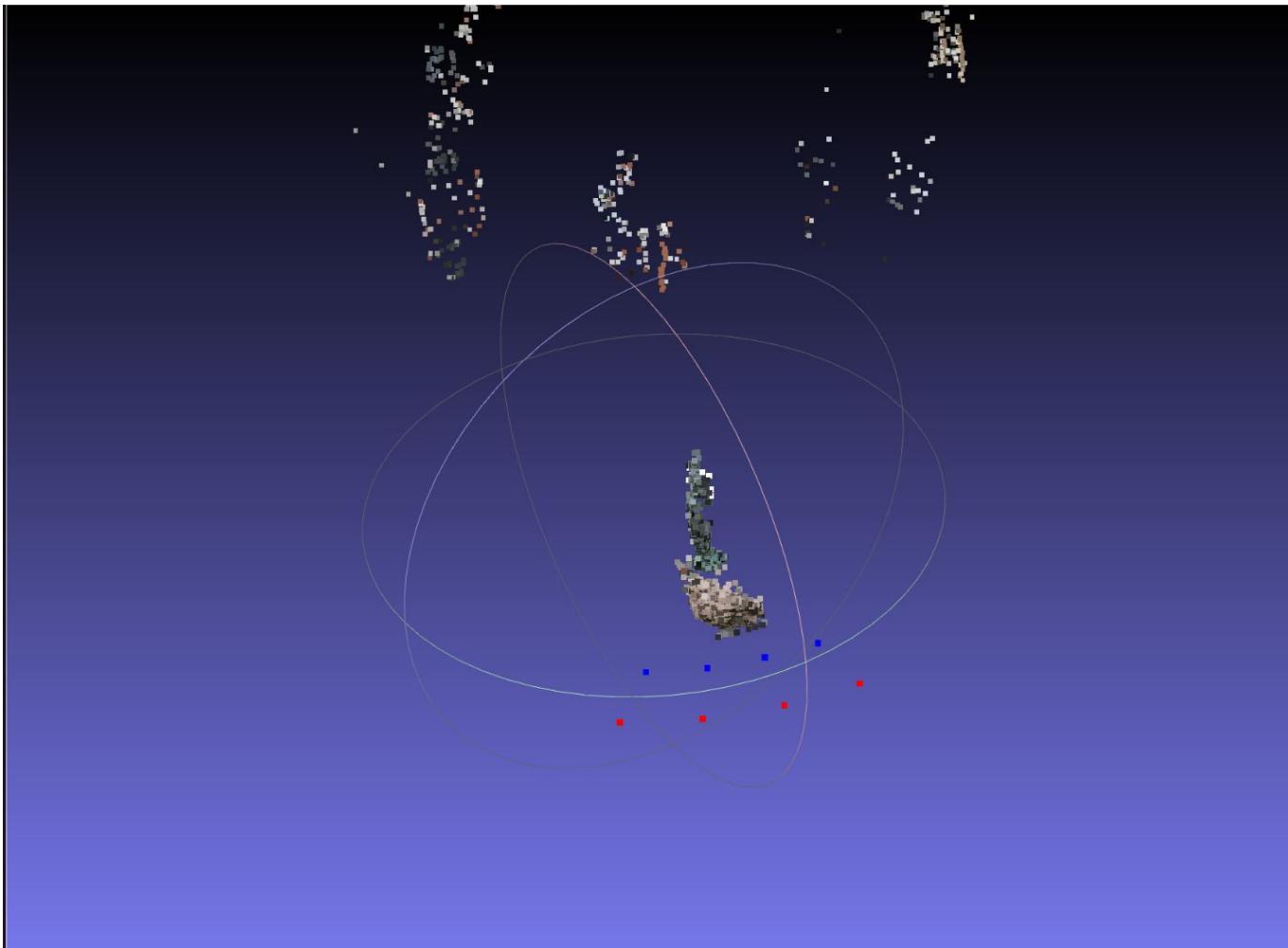


Триангуляция, резекция + уточнение положения камер и точек

BA

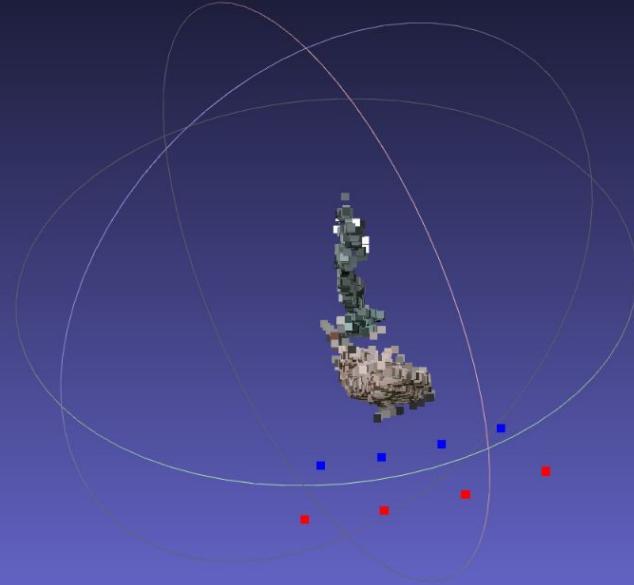


Триангуляция, резекция + уточнение положения камер и точек

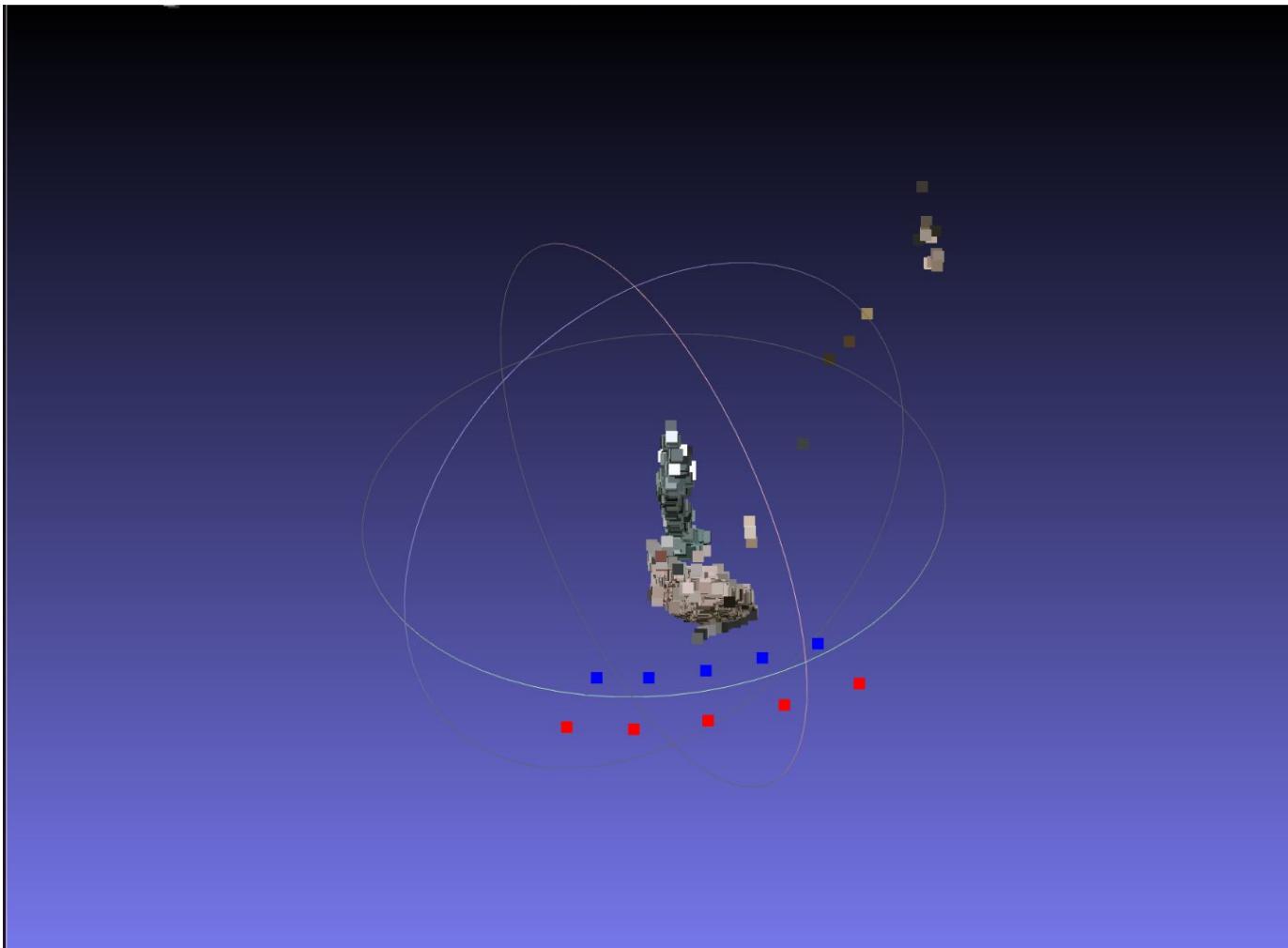


Триангуляция, резекция + уточнение положения камер и точек

BA

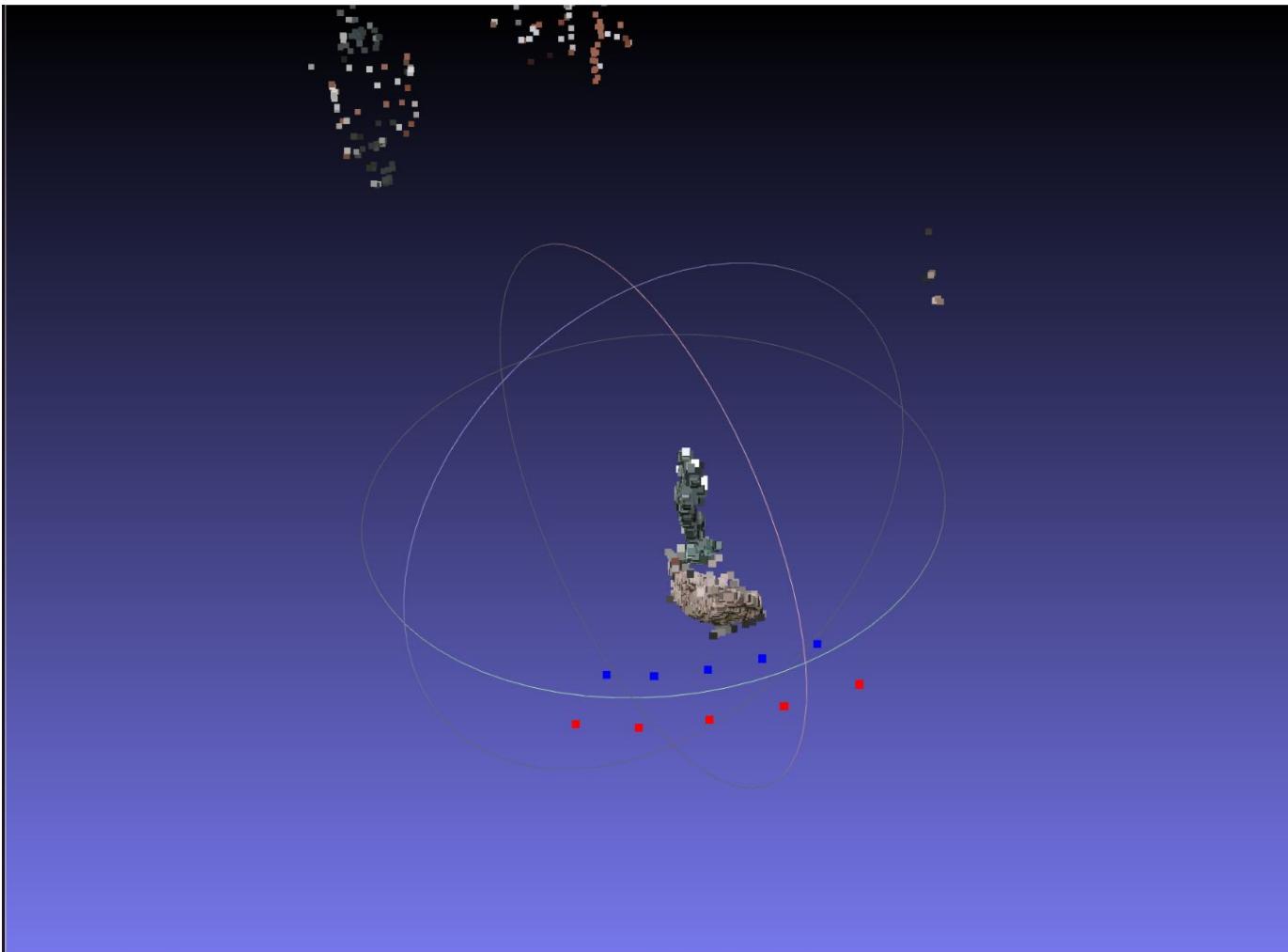


Триангуляция, резекция + уточнение положения камер и точек

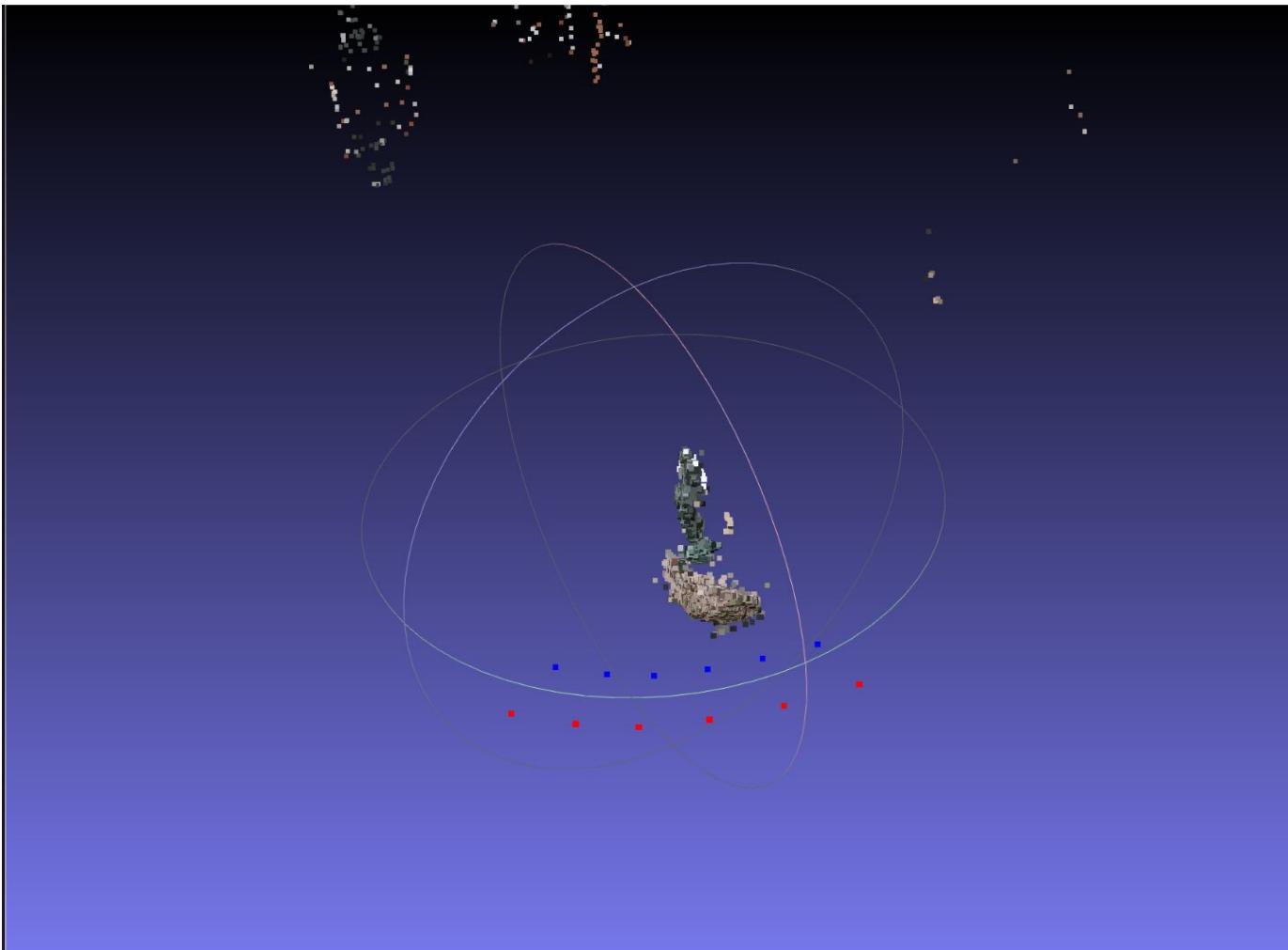


Триангуляция, резекция + уточнение положения камер и точек

BA

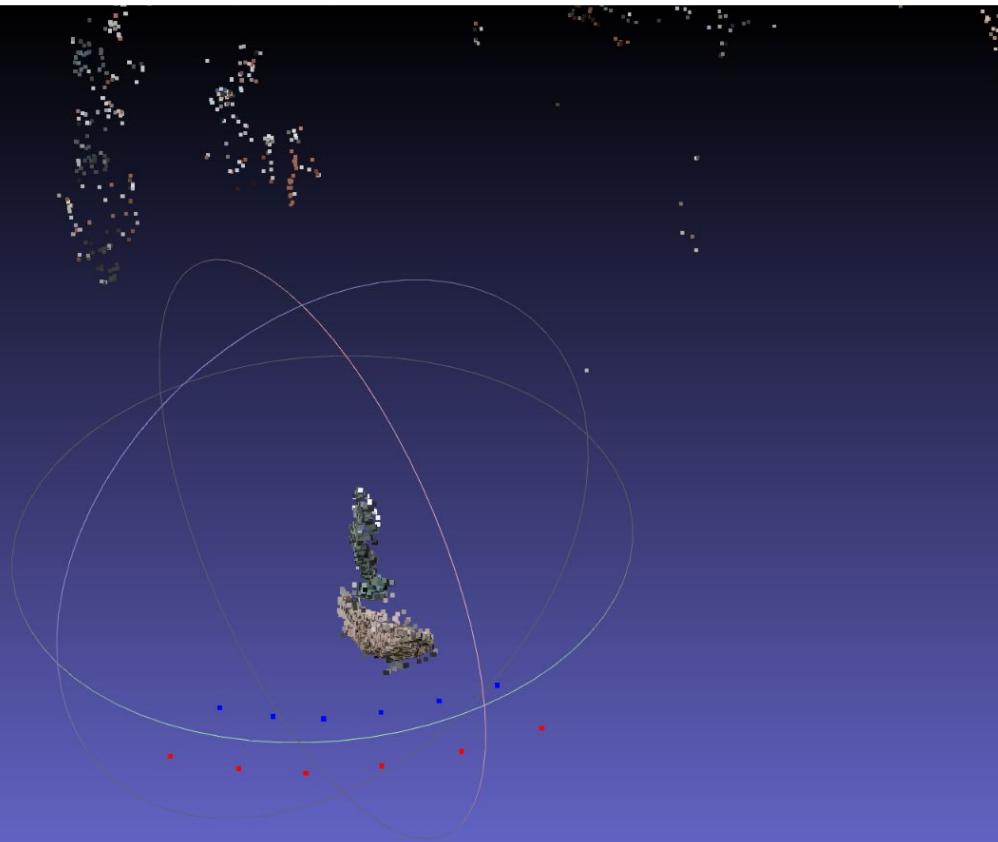


Триангуляция, резекция + уточнение положения камер и точек

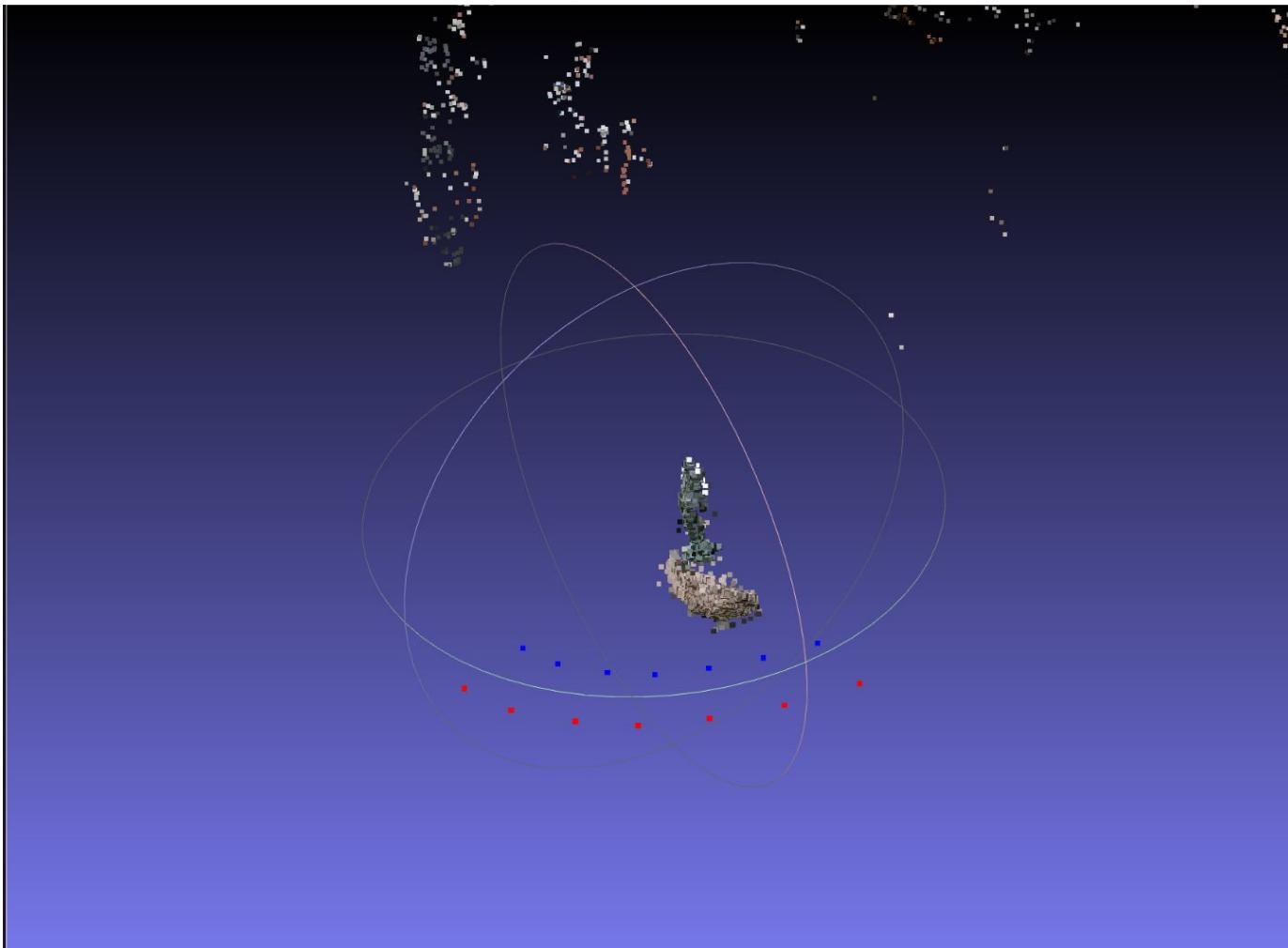


Триангуляция, резекция + уточнение положения камер и точек

BA

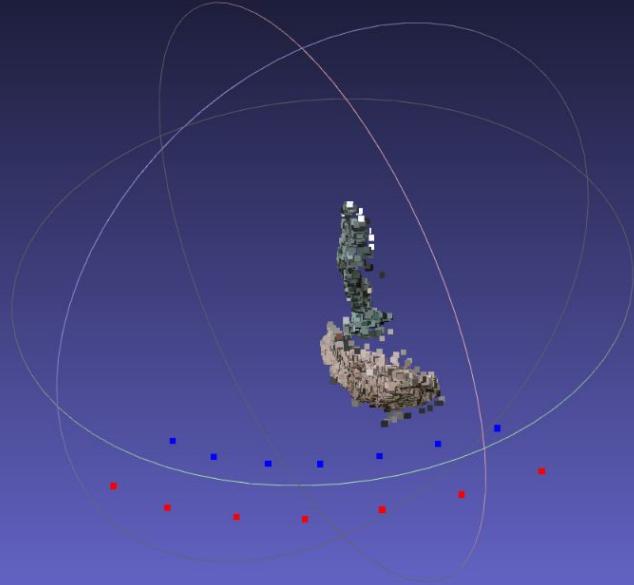


Триангуляция, резекция + уточнение положения камер и точек

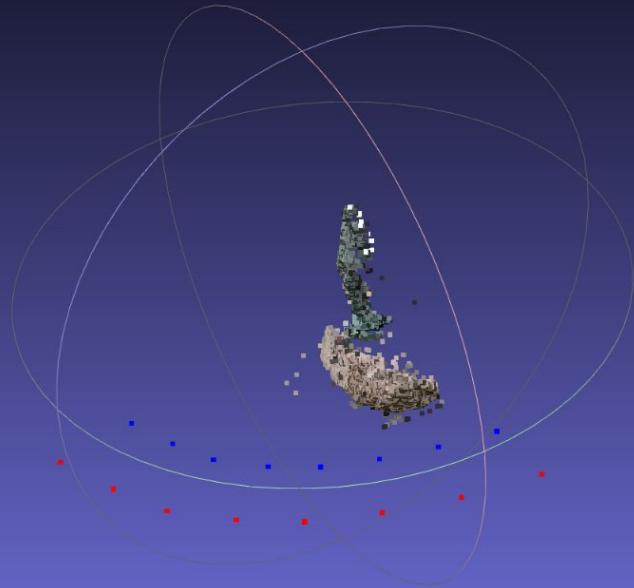


Триангуляция, резекция + уточнение положения камер и точек

BA

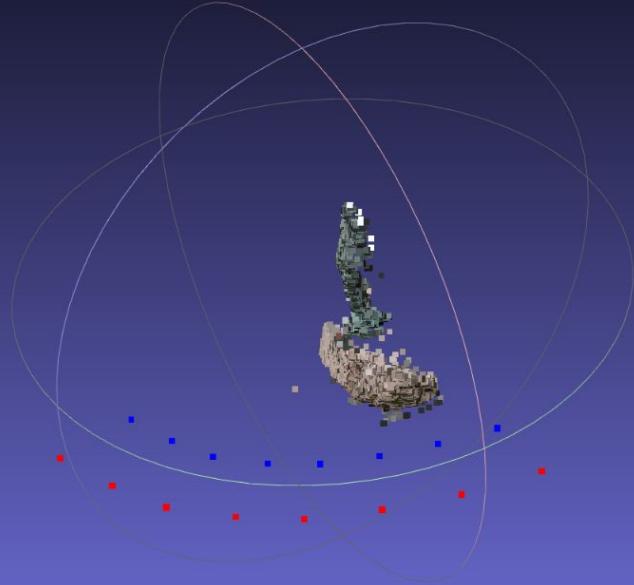


Триангуляция, резекция + уточнение положения камер и точек

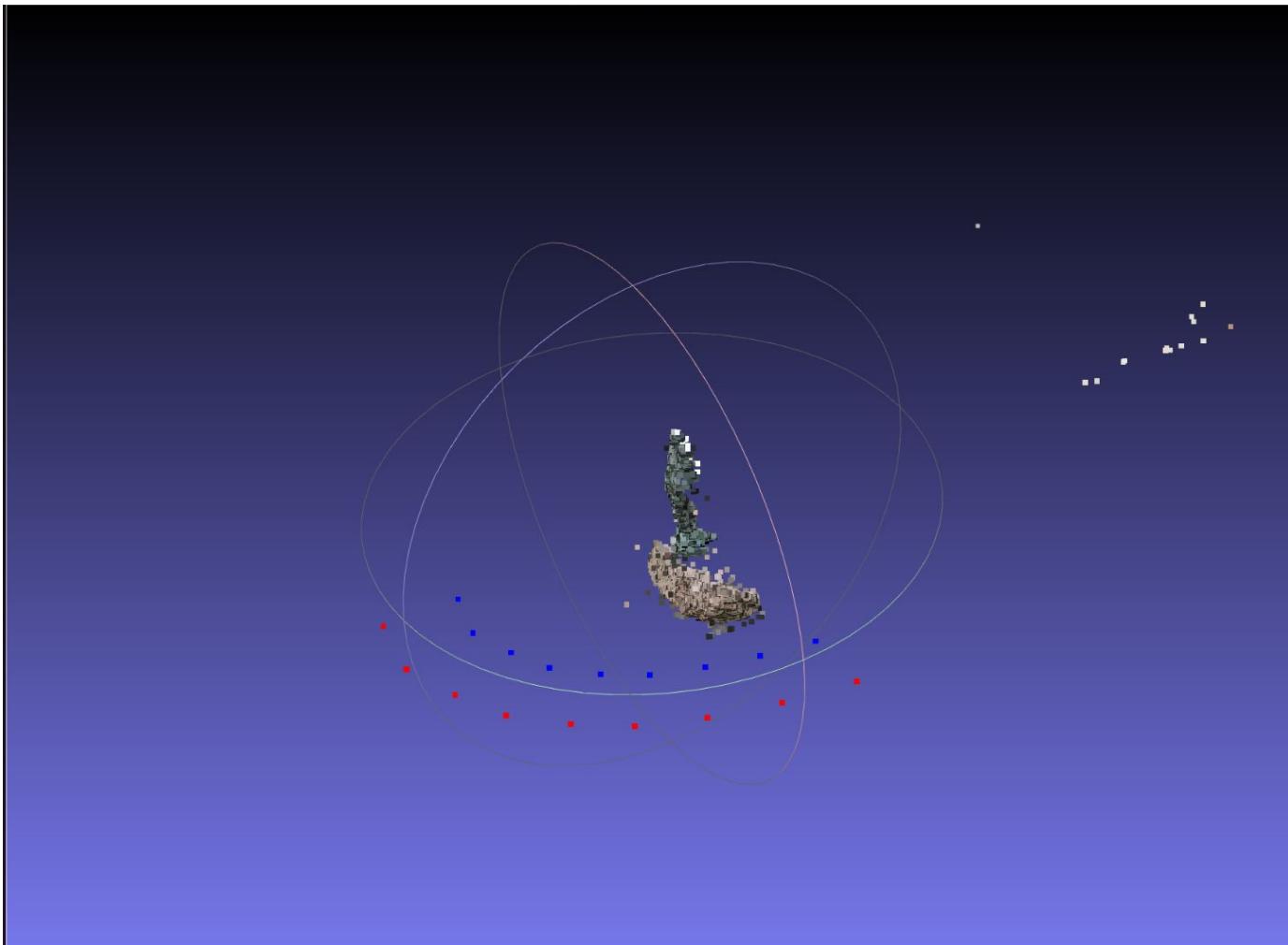


Триангуляция, резекция + уточнение положения камер и точек

BA

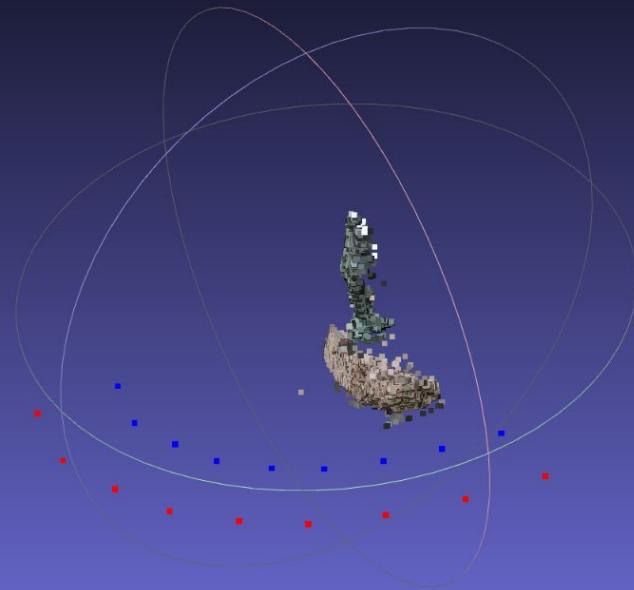


Триангуляция, резекция + уточнение положения камер и точек

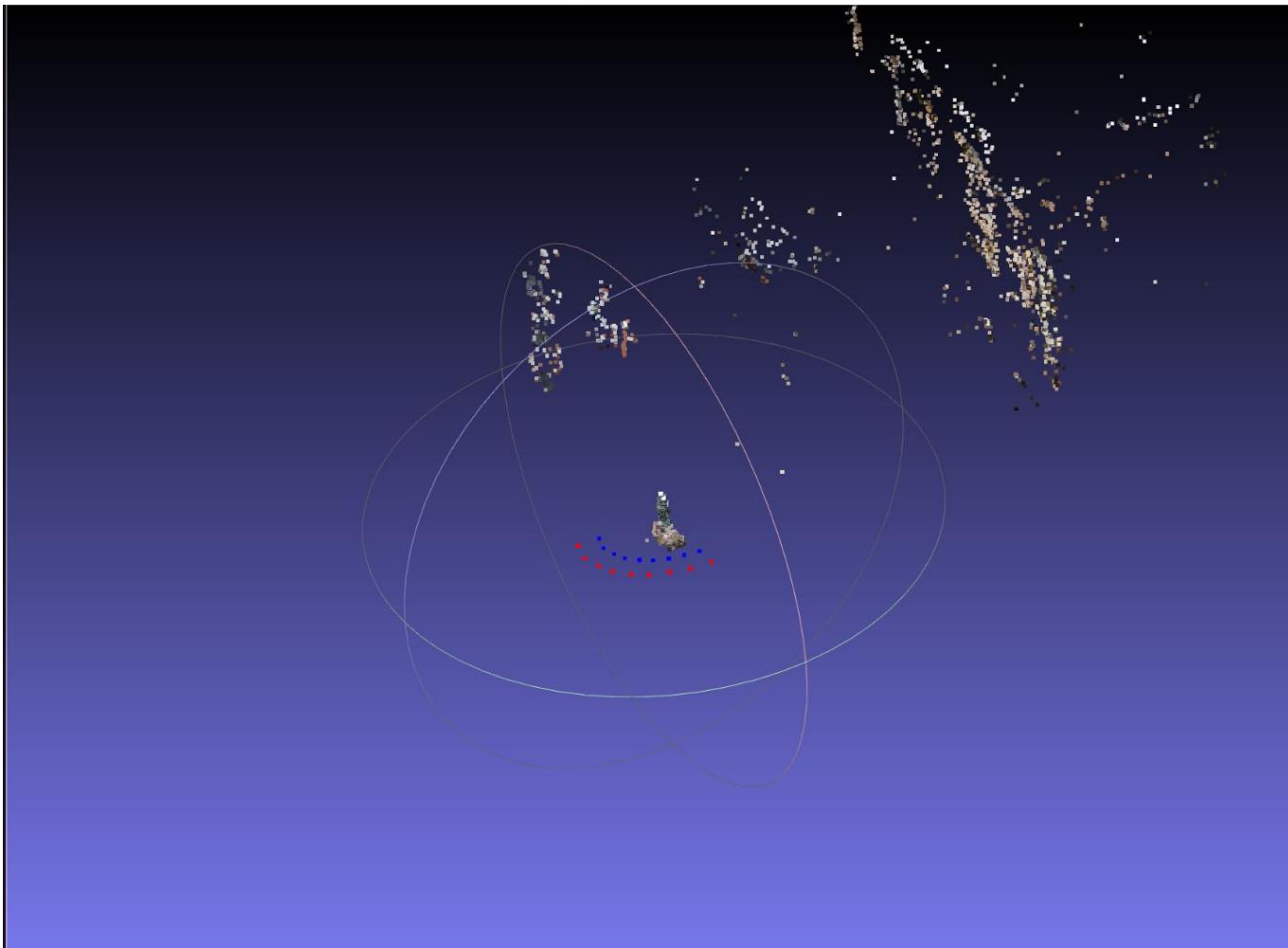


Триангуляция, резекция + уточнение положения камер и точек

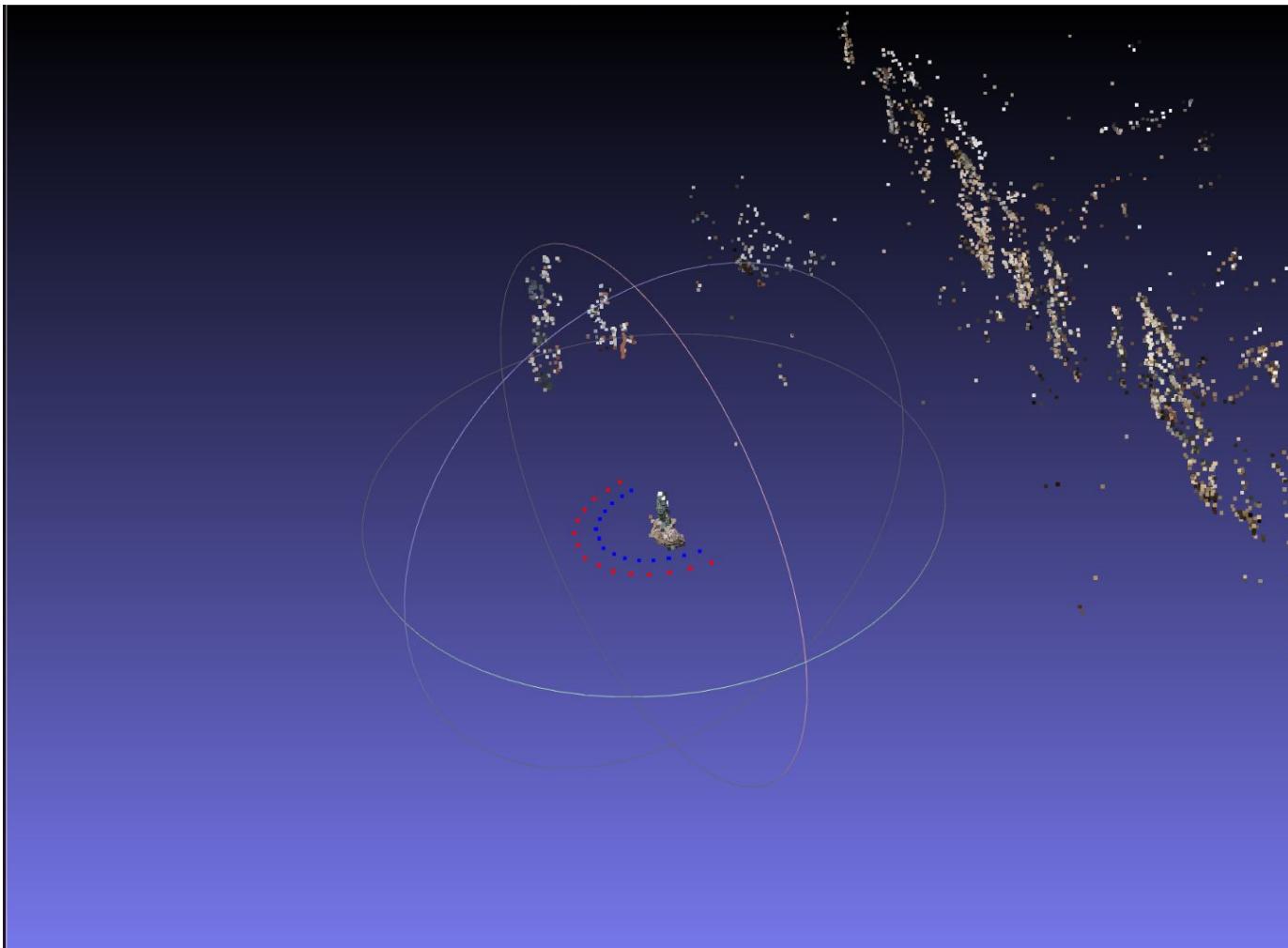
BA



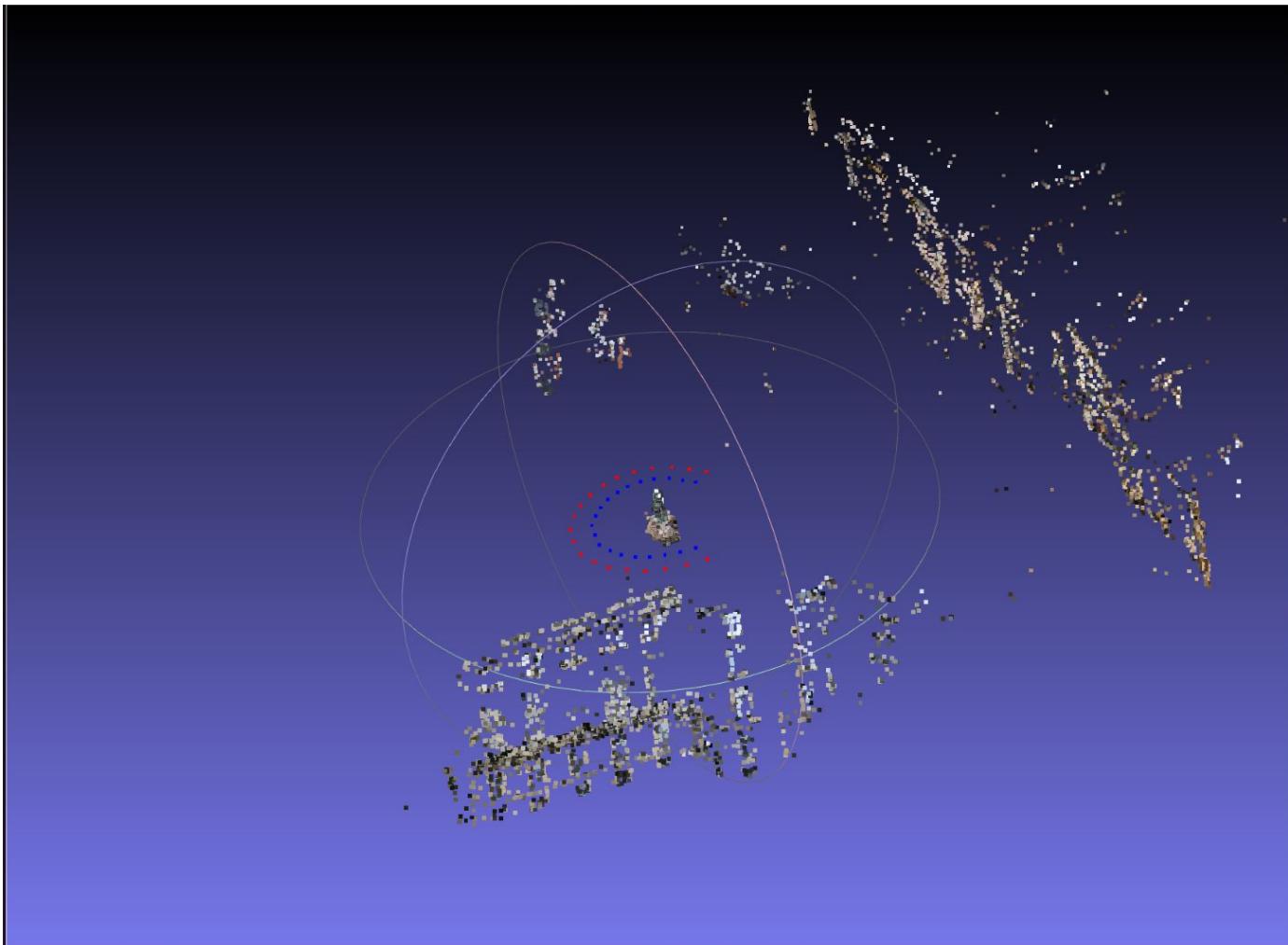
Триангуляция, резекция + уточнение положения камер и точек



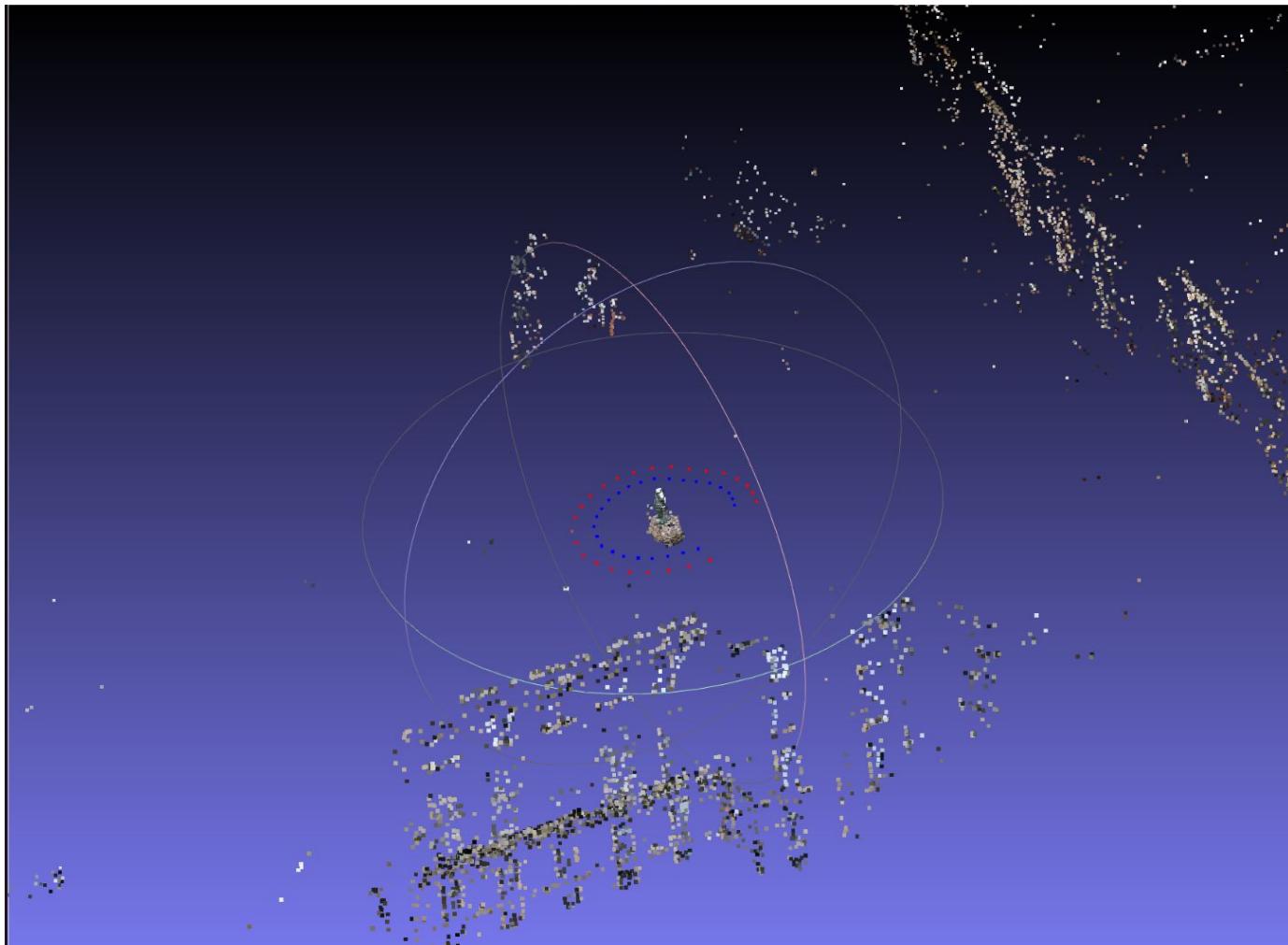
Триангуляция, резекция + уточнение положения камер и точек



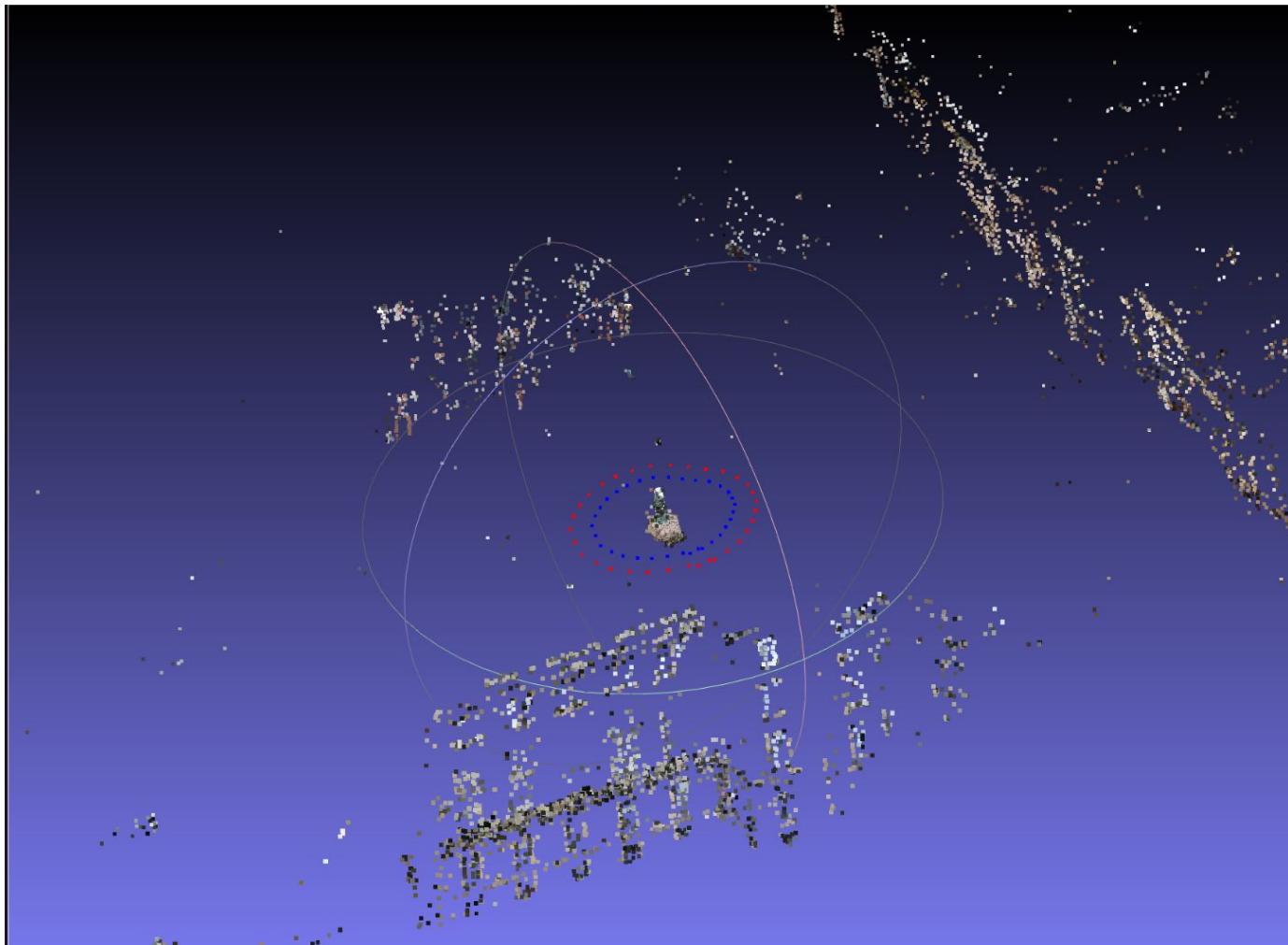
Триангуляция, резекция + уточнение положения камер и точек



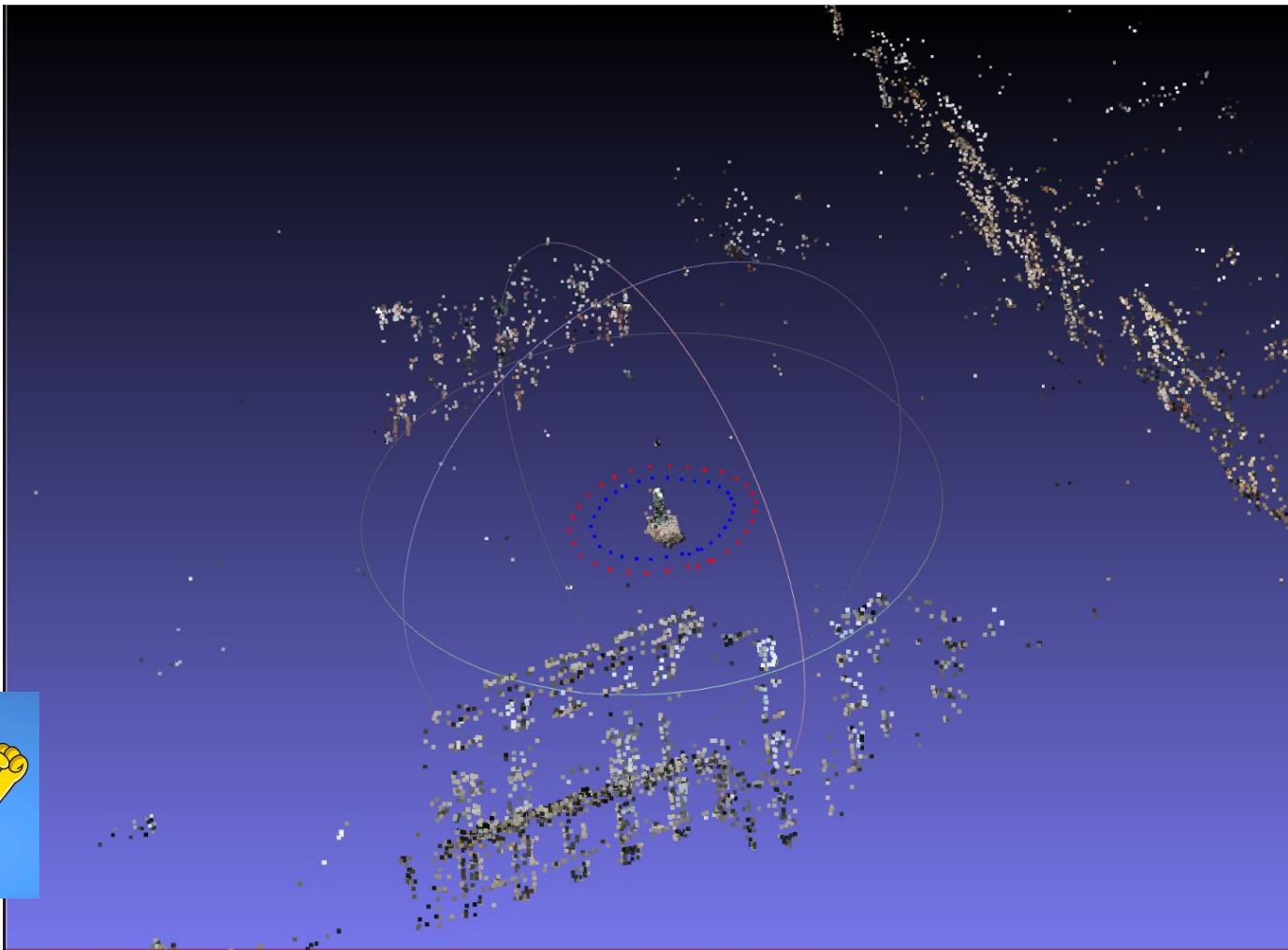
Триангуляция, резекция + уточнение положения камер и точек



Триангуляция, резекция + уточнение положения камер и точек



Триангуляция, резекция + уточнение положения камер и точек



Триангуляция, резекция + уточнение положения камер и точек

Методы оптимизации функционалов

- 1) Поиск экстремума 1: градиентный спуск
- 2) Поиск корня: алгоритм Ньютона
- 3) Поиск экстремума 2: алгоритм Ньютона
- 4) Поиск экстремума 3: алгоритм Гаусса-Ньютона
- 5) Поиск экстремума 4: алгоритм Левенберга — Марквардта

Поиск экстремума 1: градиентный спуск

Есть функция $f(x)$ хотим найти точку экстремума, например $\underset{x}{\operatorname{argmin}} f(x)$

Поиск экстремума 1: градиентный спуск

Есть функция $f(x)$ хотим найти точку экстремума, например $\underset{x}{\operatorname{argmin}} f(x)$

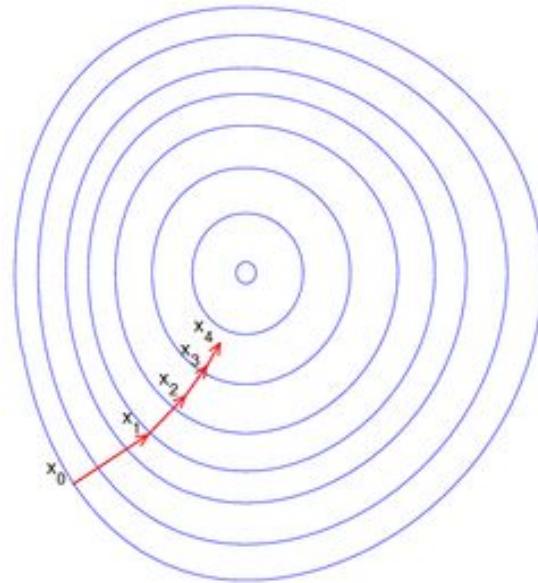
Есть первое приближение x_0 , давайте решим в какую новую точку стоит попробовать сходить?

Поиск экстремума 1: градиентный спуск

Есть функция $f(x)$ хотим найти точку экстремума, например $\underset{x}{\operatorname{argmin}} f(x)$

Есть первое приближение x_0 , давайте решим в какую новую точку стоит попробовать сходить?

Аналогия: вы в лесу на горе, вокруг туман, как дойти до вершины?



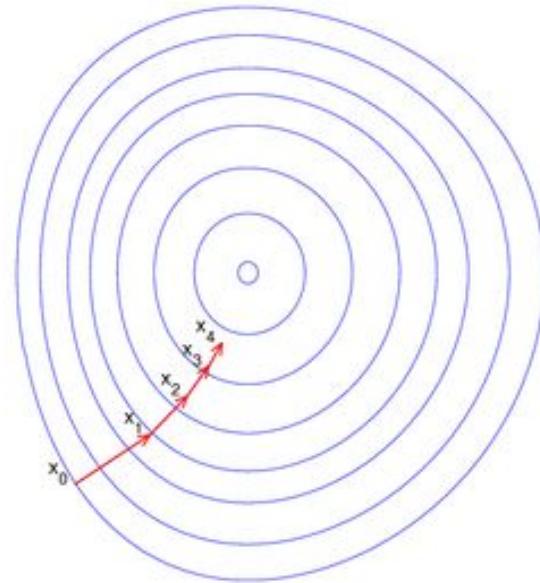
Поиск экстремума 1: градиентный спуск

Есть функция $f(x)$ хотим найти точку экстремума, например $\underset{x}{\operatorname{argmin}} f(x)$

Есть первое приближение x_0 , давайте решим в какую новую точку стоит попробовать сходить?

Аналогия: вы в лесу на горе, вокруг туман, как дойти до вершины?

Почему мы говорим об экстремуме а не о максимуме? Работает ли это для минимума?



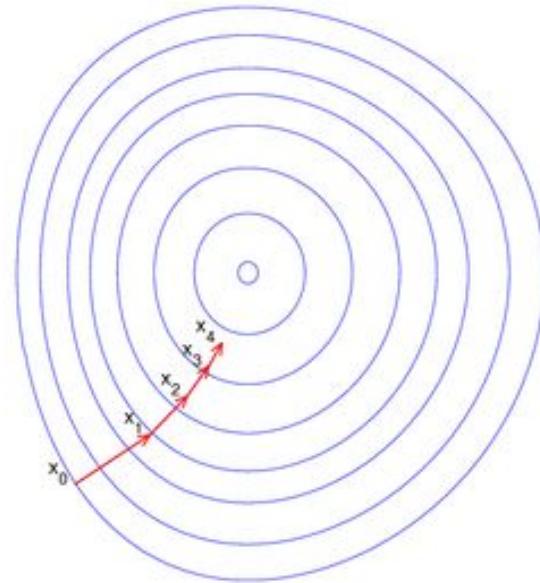
Поиск экстремума 1: градиентный спуск

Есть функция $f(x)$ хотим найти точку экстремума, например $\underset{x}{\operatorname{argmin}} f(x)$

Есть первое приближение x_0 , давайте решим в какую новую точку стоит попробовать сходить?

$$x_{h+1} = x_h - \gamma \nabla f(x_h)$$

Сдвигаемся по направлению наискильнейшего убывания $f(x)$



Поиск экстремума 1: градиентный спуск

Есть функция $f(x)$ хотим найти точку экстремума, например $\underset{x}{\operatorname{argmin}} f(x)$

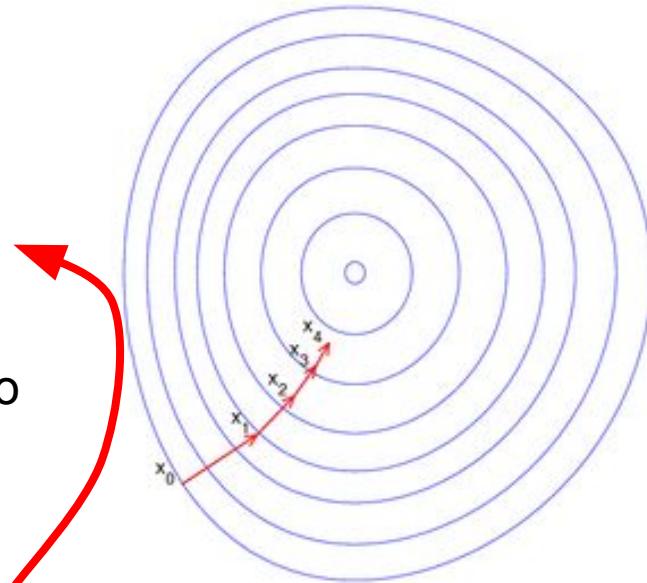
Есть первое приближение x_0 , давайте решим в какую новую точку стоит попробовать сходить?

$$x_{h+1} = x_h - \gamma \nabla f(x_h)$$

Сдвигаемся по направлению наисклонейшего

убывания $f(x)$

А как нам найти минимум изменив формулу?



Поиск экстремума 1: градиентный спуск

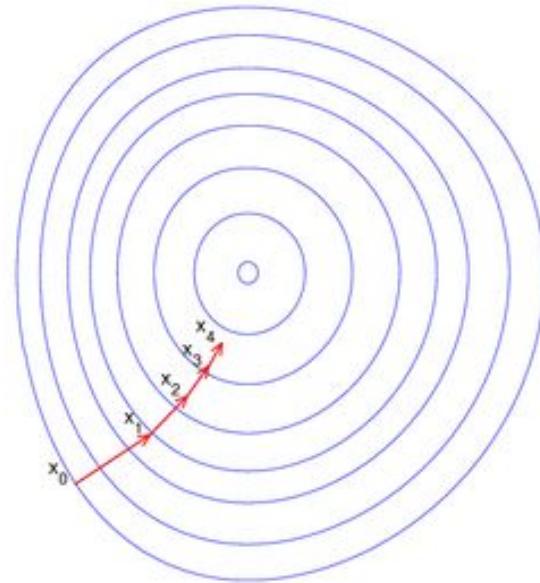
Есть функция $f(x)$ хотим найти точку экстремума, например $\underset{x}{\operatorname{argmin}} f(x)$

Есть первое приближение x_0 , давайте решим в какую новую точку стоит попробовать сходить?

$$x_{h+1} = x_h - \gamma \nabla f(x_h)$$

Сдвигаемся по направлению наискильнейшего
убывания $f(x)$

Какие могут быть проблемы?



Поиск экстремума 1: градиентный спуск

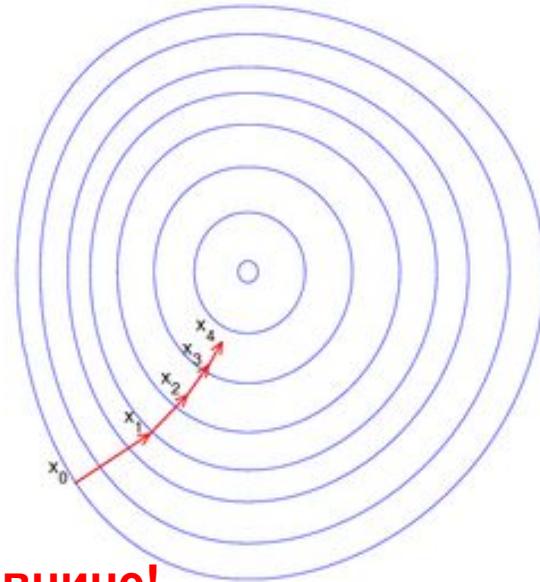
Есть функция $f(x)$ хотим найти точку экстремума, например $\underset{x}{\operatorname{argmin}} f(x)$

Есть первое приближение x_0 , давайте решим в какую новую точку стоит попробовать сходить?

$$x_{h+1} = x_h - \gamma \nabla f(x_h)$$

Сдвигаемся по направлению наисильнейшего
убывания $f(x)$

Локальные экстремумы! Медленно идем по равнине!



Поиск корня: Метод Ньютона

Сначала решим другую задачу: найдем корень функции

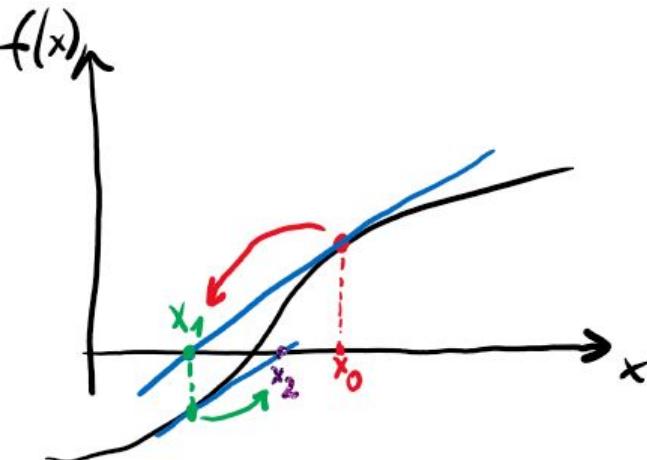
т.е. для $f(x)$ хотим решить уравнение $f(x) = 0$

Поиск корня: Метод Ньютона

Сначала решим другую задачу: найдем корень функции

т.е. для $f(x)$ хотим решить уравнение $f(x) = 0$

- 1) Считаем производную
- 2) Проводим касательную
- 3) Прыгаем в точку пересечения оси
- 4) Повторяем



Поиск корня: Метод Ньютона

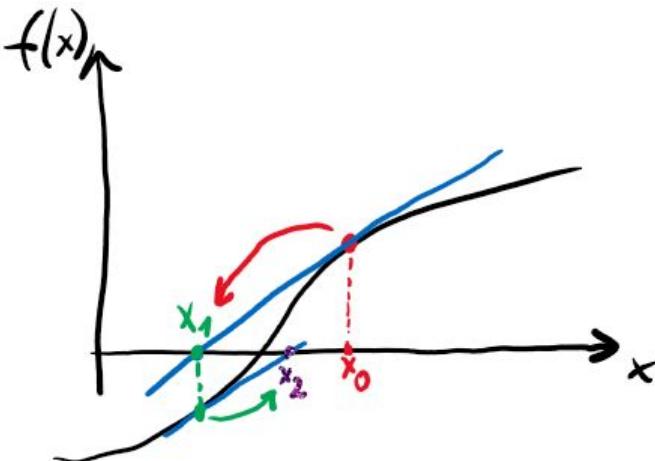
Сначала решим другую задачу: найдем корень функции

т.е. для $f(x)$ хотим решить уравнение $f(x) = 0$

Иначе говоря, через ряд Тейлора:

$$f(x_{n+1}) \approx f(x_n) + (x_{n+1} - x_n) f'(x_n)$$

текущая точка
следующая точка



Поиск корня: Метод Ньютона

Сначала решим другую задачу: найдем корень функции

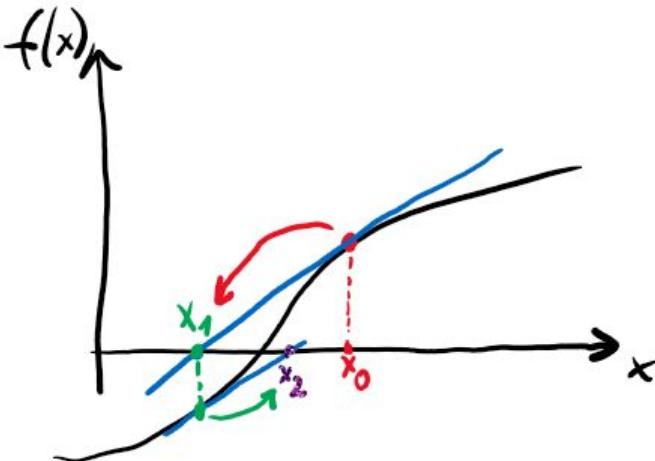
т.е. для $f(x)$ хотим решить уравнение $f(x) = 0$

Иначе говоря, через ряд Тейлора:

$$f(x_{n+1}) \approx f(x_n) + (x_{n+1} - x_n) f'(x_n)$$

↑
текущая точка
↑
следующая точка - как ее вычислить?

Что мы знаем про левую часть выражения?



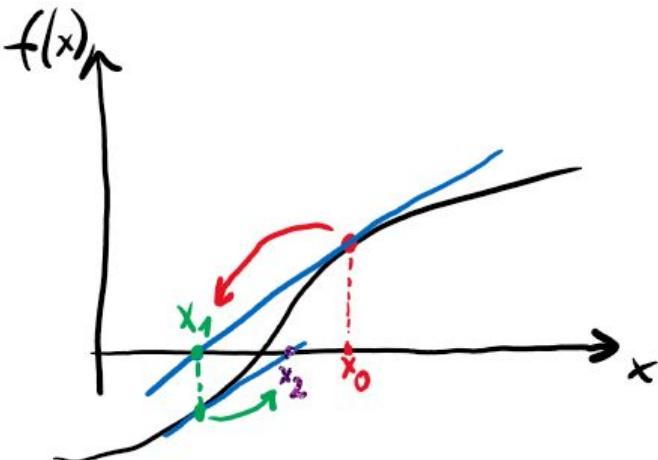
Поиск корня: Метод Ньютона

Сначала решим другую задачу: найдем корень функции

т.е. для $f(x)$ хотим решить уравнение $f(x) = 0$

Иначе говоря, через ряд Тейлора:

$$f(x_{n+1}) \approx f(x_n) + (x_{n+1} - x_n) f'(x_n)$$
$$\stackrel{||}{=} \Rightarrow \stackrel{||}{=} \Rightarrow x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$



Поиск корня: Метод Ньютона

Сначала решим другую задачу: найдем корень функции

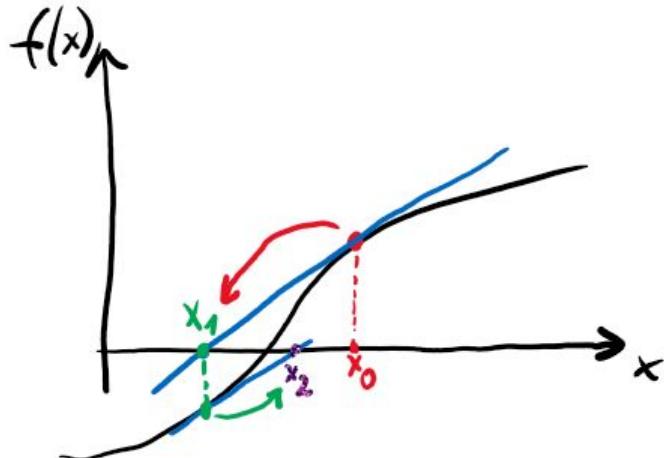
т.е. для $f(x)$ хотим решить уравнение $f(x) = 0$

Иначе говоря, через ряд Тейлора:

$$f(x_{n+1}) \approx f(x_n) + (x_{n+1} - x_n) f'(x_n)$$

$\parallel \quad \parallel \quad \Rightarrow \quad \parallel \quad \parallel$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$



Для каких функций мы найдем корень за один шаг?

Как искать локальный экстремум?

Поиск экстремума 2: алгоритм Ньютона

- 1) Есть функция $f(x)$ хотим найти точку экстремума, т.е. $f'(x) = 0$

Поиск экстремума 2: алгоритм Ньютона

- 1) Есть функция $f(x)$ хотим найти точку экстремума, т.е. $f'(x) = 0$
- 2) Переформулируем:
- 3) Есть функция $f'(x)$ хотим решить уравнение $f'(x) = 0$

Поиск экстремума 2: алгоритм Ньютона

- 1) Есть функция $f(x)$ хотим найти точку экстремума, т.е. $f'(x) = 0$
- 2) Переформулируем:
- 3) Есть функция $f'(x)$ хотим решить уравнение $f'(x) = 0$
- 4) Свели задачу к поиску корня алгоритмом Ньютона:

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

Поиск экстремума 2: алгоритм Ньютона

- 1) Есть функция $f(x)$ хотим найти точку экстремума, т.е. $f'(x) = 0$
- 2) Переформулируем:
- 3) Есть функция $f'(x)$ хотим решить уравнение $f'(x) = 0$
- 4) Свели задачу к поиску корня алгоритмом Ньютона:

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

- 5) $f'(x)$ - в каких случаях прыгнем в экстремум за один шаг?

Поиск экстремума 2: алгоритм Ньютона

- 1) Есть функция $f(x)$ хотим найти точку экстремума, т.е. $f'(x) = 0$
- 2) Переформулируем:
- 3) Есть функция $f'(x)$ хотим решить уравнение $f'(x) = 0$
- 4) Свели задачу к поиску корня алгоритмом Ньютона:

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

- 5) Если $f'(x)$ - похожа на линейную функцию в окрестности содержащей нас и экстремум - то прыгнем в экстремум **за один шаг!**
 $f'(x)$ - похожа на линейную $\Leftrightarrow f(x)$ - что это говорит про исходную?

Поиск экстремума 2: алгоритм Ньютона

- 1) Есть функция $f(x)$ хотим найти точку экстремума, т.е. $f'(x) = 0$
- 2) Переформулируем:
- 3) Есть функция $f'(x)$ хотим решить уравнение $f'(x) = 0$
- 4) Свели задачу к поиску корня алгоритмом Ньютона:

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

- 5) Если $f'(x)$ - похожа на линейную функцию в окрестности содержащей нас и экстремум - то прыгнем в экстремум **за один шаг!**
 $f'(x)$ - похожа на линейную $\Leftrightarrow f(x)$ - похожа на кривую второго порядка.

Поиск экстремума 2: алгоритм Ньютона

- 1) Есть функция $f(x)$ хотим найти точку экстремума, т.е. $f'(x) = 0$
- 2) Переформулируем:
- 3) Есть функция $f'(x)$ хотим решить уравнение $f'(x) = 0$
- 4) Свели задачу к поиску корня алгоритмом Ньютона:

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

для произвольной размерности:

$$x_{n+1} = x_n - [\underbrace{Hf(x_n)}_{\text{Гессиан}}]^{-1} \nabla f(x_n)$$

Тессиан

Поиск экстремума 3: алгоритм Гаусса-Ньютона

Пусть есть функция и наблюдения.

$$g(x) : \mathbb{R}^n \rightarrow \mathbb{R}$$

m наблюдений: y_i

Поиск экстремума 3: алгоритм Гаусса-Ньютона

Пусть есть функция и наблюдения.

$$g(x) : \mathbb{R}^n \rightarrow \mathbb{R}$$

m наблюдений: y_i

ищем такой $x \in \mathbb{R}^n$, что:

$$\forall i=1 \dots m: g(x) \approx y_i$$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

Пусть есть функция и наблюдения.

Функция может быть нелинейной, поэтому **Non-linear Least squares**.

$$g(x) : \mathbb{R}^n \rightarrow \mathbb{R}$$

m наблюдений: y_i

ищем такой $x \in \mathbb{R}^n$, что:

$$\forall i=1 \dots m: g(x) \approx y_i$$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

Пусть есть функция и наблюдения.

Функция может быть нелинейной, поэтому **Non-linear Least squares**.

$$g(x) : \mathbb{R}^n \rightarrow \mathbb{R}$$

m наблюдений: y_i

ищем такой $x \in \mathbb{R}^n$, что:

$$\forall i=1 \dots m: g(x) \approx y_i$$

Как переформулировать?
Что значит “примерно равно”?

Поиск экстремума 3: алгоритм Гаусса-Ньютона

Переформулируем:

$$\forall i=1 \dots m : g(x) \approx y_i$$
$$f(\underset{\substack{\uparrow \\ \text{(исковый)} \\ \text{параметр)}}{x}) = \frac{1}{2} \sum_{i=1 \dots m} (g(x) - y_i)^2 = \frac{1}{2} \sum r_i(x)^2 = r(x)^T r(x)$$
$$\begin{pmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_n(x) \end{pmatrix}$$

residual
(небязка)

Итого опять ищем $\arg\min_x f(x)$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

Переформулируем:

~~$\forall i=1 \dots m : g(x) \approx y_i$~~

Какие есть аналогии с маш. обучем?

$f(\underset{\substack{\uparrow \\ \text{исковый} \\ \text{параметр}}}{x}) = \frac{1}{2} \sum_{i=1 \dots m} (g(x) - y_i)^2 = \frac{1}{2} \sum r_i(x)^2 = r(x)^T r(x)$

$r(x) = \begin{pmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_n(x) \end{pmatrix}$

$r_i(x)$ residual (небязка)

Итого опять ищем $\arg\min_x f(x)$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

$$f(x) = r(x)^T \cdot r(x)$$

$$\nabla f(x) = \left(\frac{\partial r_1}{\partial x_1}, r_1(x) + \frac{\partial r_2}{\partial x_1} r_2(x) + \dots + \frac{\partial r_m}{\partial x_1} r_m(x) \right)$$

$$\begin{pmatrix} \frac{\partial r_1}{\partial x_1} \\ \frac{\partial r_1}{\partial x_2} \\ \vdots \\ \frac{\partial r_1}{\partial x_m} \end{pmatrix}$$

$$r(x) = \begin{pmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_m(x) \end{pmatrix}$$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

$$\nabla f(x) = \begin{pmatrix} \frac{\partial r_1}{\partial x_1} \\ \frac{\partial r_1}{\partial x_2} \\ \vdots \\ \frac{\partial r_1}{\partial x_m} \end{pmatrix} r_1(x) + \begin{pmatrix} \frac{\partial r_2}{\partial x_1} \\ \frac{\partial r_2}{\partial x_2} \\ \vdots \\ \frac{\partial r_2}{\partial x_m} \end{pmatrix} r_2(x) + \dots + \begin{pmatrix} \frac{\partial r_m}{\partial x_1} \\ \frac{\partial r_m}{\partial x_2} \\ \vdots \\ \frac{\partial r_m}{\partial x_m} \end{pmatrix} r_m(x)$$

$$\nabla f(x) = \begin{pmatrix} \frac{\partial r_1}{\partial x_1} & \dots & \frac{\partial r_m}{\partial x_1} \\ \vdots & & \vdots \\ \frac{\partial r_1}{\partial x_m} & \dots & \frac{\partial r_m}{\partial x_m} \end{pmatrix} \begin{pmatrix} r_1(x) \\ \vdots \\ r_m(x) \end{pmatrix} = J(x) r(x)$$

Якобиан
(матриця)

Поиск экстремума 3: алгоритм Гаусса-Ньютона

$$\nabla f(x) = J^T(x) r(x)$$

В методе Ньютона для:

$$x_{n+1} = x_n - \left[H(x_n) \right]^{-1} \frac{\nabla f(x_n)}{J^T(x_n) r(x_n)}$$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

$$\nabla f(x) = J^T(x) r(x)$$

В методе Ньютона буд:

$$x_{n+1} = x_n - \left[H(x_n) \right]^{-1} \nabla f(x_n)$$

В методе Гаусс-Ньютона *приближенный гессиан*:

(ради экономии времени вычислений)

$$H(x_n) \approx J^T(x_n) J(x_n)$$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

$$\nabla f(x) = J^T(x) r(x)$$

В методе Ньютона для:

$$x_{n+1} = x_n - \left[H(x_n) \right]^{-1} \nabla f(x_n)$$

В методе Гаусс-Ньютона *приближенной* гессиан:

$$H(x_n) \approx J^T(x_n) J(x_n)$$

м.о. метод Гаусс-Ньютона:

$$x_{n+1} = x_n - \left[J^T(x_n) J(x_n) \right]^{-1} J^T(x_n) r(x_n)$$

Сравнение методов оптимизации (поиска экстремума)

Метод Ньютона:

- **хорошо:** если окрестность похожа на кривую второго порядка, то прыгнем в экстремум за один шаг
- **плохо:** требуется считать вторые производные (гессиан гораздо плотнее якобиана)

Метод Гаусса-Ньютона:

- **хорошо:** если окрестность похожа на кривую второго порядка, то прыгнем в экстремум за один шаг
- **хорошо:** не нужно считать вторые производные (якобиан гораздо разреженнее гессиана - нам ведь обращать матрицу)
- **плохо:** подходит только для минимизации суммы квадратов

Сравнение методов оптимизации (поиска экстремума)

Метод Ньютона:

- **хорошо:** если окрестность похожа на кривую второго порядка, то прыгнем в экстремум за один шаг
- **плохо:** требуется считать вторые производные (гессиан гораздо плотнее якобиана)

Метод Гаусса-Ньютона:

- **хорошо:** если окрестность похожа на кривую второго порядка, то прыгнем в экстремум за один шаг
- **хорошо:** не нужно считать вторые производные (якобиан гораздо разреженнее гессиана - нам ведь обращать матрицу)
- **нас устраивает:** подходит только для минимизации суммы квадратов

Сравнение методов оптимизации (поиска экстремума)

Метод Градиентного спуска:

- плохо: (решаемо) не учитывает кривизну - медленно сходится по пологой поверхности, далеко прыгает при движении по резким обрывам, пример: протяженная впадина-овраг
- плохо: медленно сходится даже если экстремум близко
- хорошо: всегда движется в правильном направлении

Метод Гаусса-Ньютона:

- хорошо: если окрестность похожа на кривую второго порядка, то прыгнем в экстремум за один шаг
- плохо: может прыгнуть не туда

Сравнение методов оптимизации (поиска экстремума)

Метод Градиентного спуска:

- **плохо:** (решаемо) не учитывает кривизну - медленно сходится по пологой поверхности, далеко прыгает при движении по резким обрывам, пример: протяженная впадина-овраг
- **плохо:** медленно сходится даже если экстремум близко
- **хорошо:** всегда движется в правильном направлении

Метод Гаусса-Ньютона:

- **хорошо:** если окрестность похожа на кривую второго порядка, то прыгнем в экстремум за один шаг
- **плохо:** может прыгнуть не туда

Как взять лучшее от градиентного спуска и Гаусса-Ньютона?

Например в начале кто лучше работает?

А под конец близко с экстремумом кто?

Сравнение методов оптимизации (поиска экстремума)

Метод Градиентного спуска:

- плохо: (решаемо) не учитывает кривизну - медленно сходится по пологой поверхности, далеко прыгает при движении по резким обрывам, пример: протяженная впадина-овраг
- плохо: медленно сходится даже если экстремум близко
- хорошо: всегда движется в правильном направлении

Метод Гаусса-Ньютона:

- хорошо: если окрестность похожа на кривую второго порядка, то прыгнем в экстремум за один шаг
- плохо: может прыгнуть не туда

Как взять лучшее от градиентного спуска и Гаусса-Ньютона?

Например в начале все очень ненадежно
- лучше Градиентный спуск.
А под конец близко с экстремумом лучше
Гаусс-Ньютон.

Сравнение методов оптимизации (поиска экстремума)

Метод Градиентного спуска:

- **плохо:** (решаемо) не учитывает кривизну - медленно сходится по пологой поверхности, далеко прыгает при движении по резким обрывам, пример: протяженная впадина-овраг
- **плохо:** медленно сходится даже если экстремум близко
- **хорошо:** всегда движется в правильном направлении

Метод Гаусса-Ньютона:

- **хорошо:** если окрестность похожа на кривую второго порядка, то прыгнем в экстремум за один шаг
- **плохо:** может прыгнуть не туда

Как взять лучшее от градиентного спуска и Гаусса-Ньютона?

Как совместить их мнения о том куда шагать?

Поиск экстремума 4: алгоритм Левенберга — Марквардта

$$\text{Градиентный спуск: } \mathbf{x}_{n+1} = \mathbf{x}_n - \lambda \nabla f(\mathbf{x}_n) = \mathbf{x}_n - \lambda \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

Поиск экстремума 4: алгоритм Левенберга — Марквардта

$$\text{Градиентный спуск: } \mathbf{x}_{n+1} = \mathbf{x}_n - \lambda \nabla f(\mathbf{x}_n) = \mathbf{x}_n - \lambda \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$$\text{Гаусс-Ньютон: } \mathbf{x}_{n+1} = \mathbf{x}_n - \left[\underbrace{\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n)}_{\mathcal{H}(\mathbf{x}_n) = \nabla^2 f(\mathbf{x}_n)} \right]^{-1} \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

Поиск экстремума 4: алгоритм Левенберга — Марквардта

$$\text{Градиентный спуск: } \mathbf{x}_{n+1} = \mathbf{x}_n - \lambda \nabla f(\mathbf{x}_n) = \mathbf{x}_n - \lambda \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$$\text{Гаусс-Ньютон: } \mathbf{x}_{n+1} = \mathbf{x}_n - \left[\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) \right]^{-1} \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) = \mathbf{H}(\mathbf{x}_n) = \nabla^2 f(\mathbf{x}_n)$

Как можно совместить эти формулы?
Что в нем общего?

Поиск экстремума 4: алгоритм Левенберга — Марквардта

$$\text{Градиентный спуск: } \mathbf{x}_{n+1} = \mathbf{x}_n - \lambda \nabla f(\mathbf{x}_n) = \mathbf{x}_n - \lambda \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$$\text{Гаусс-Ньютон: } \mathbf{x}_{n+1} = \mathbf{x}_n - \left[\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) \right]^{-1} \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) = \nabla^2 f(\mathbf{x}_n)$

$$\text{Совместим: } \mathbf{x}_{n+1} = \mathbf{x}_n - \left[\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) + \lambda \mathbf{I} \right]^{-1} \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

единичная матр.

Поиск экстремума 4: алгоритм Левенберга — Марквардта

$$\text{Градиентный спуск: } \mathbf{x}_{n+1} = \mathbf{x}_n - \lambda \nabla f(\mathbf{x}_n) = \mathbf{x}_n - \lambda \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$$\text{Гаусс-Ньютон: } \mathbf{x}_{n+1} = \mathbf{x}_n - \left[\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) \right]^{-1} \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) = \nabla^2 f(\mathbf{x}_n)$

$$\text{Совместим: } \mathbf{x}_{n+1} = \mathbf{x}_n - \left[\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) + \lambda \mathbf{I} \right]^{-1} \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

единичная матр.

Если λ мало - ???, если велико - ???.

Поиск экстремума 4: алгоритм Левенберга — Марквардта

$$\text{Градиентный спуск: } \mathbf{x}_{n+1} = \mathbf{x}_n - \lambda \nabla f(\mathbf{x}_n) = \mathbf{x}_n - \lambda \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$$\text{Гаусс-Ньютон: } \mathbf{x}_{n+1} = \mathbf{x}_n - \left[\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) \right]^{-1} \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

$\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) = \nabla^2 f(\mathbf{x}_n)$

$$\text{Совместим: } \mathbf{x}_{n+1} = \mathbf{x}_n - \left[\mathbf{J}^T(\mathbf{x}_n) \mathbf{J}(\mathbf{x}_n) + \lambda \mathbf{I} \right]^{-1} \mathbf{J}^T(\mathbf{x}_n) \mathbf{r}(\mathbf{x}_n)$$

единичная матр.

Если λ мало - Гаусс-Ньютон, если велико - почти градиентный спуск.

Поиск экстремума 4: алгоритм Левенберга — Марквардта

1) При больших лямбдах мы движемся в направлении антиградиента.

$$x_{n+1} = x_n - \left[J^T(x_n) J(x_n) + \lambda I \right]^{-1} J^T(x_n) r(x_n)$$

если лямбда маленькая

$$\lambda \rightarrow +\infty : \left[J^T(x_n) J(x_n) + \lambda I \right]^{-1}$$

Поиск экстремума 4: алгоритм Левенберга — Марквардта

1) При больших лямбдах мы движемся в направлении антиградиента.

$$x_{n+1} = x_n - \left[J^T(x_n) J(x_n) + \lambda I \right]^{-1} J^T(x_n) r(x_n)$$

если маленькая лямбда

$$\lambda \rightarrow +\infty : \lambda \left[J^T(x_n) J(x_n) + \lambda I \right]^{-1} = \left(I - J^T(x_n) J(x_n) / \lambda \right)^+$$

п. Тейлора

Поиск экстремума 4: алгоритм Левенберга — Марквардта

1) При больших лямбдах мы движемся в направлении антиградиента.

$$x_{n+1} = x_n - \left[J^T(x_n) J(x_n) + \lambda I \right]^{-1} J^T(x_n) r(x_n)$$

с увеличивающейся лямбдой

$$\lambda \rightarrow +\infty : \lambda \left[J^T(x_n) J(x_n) + \lambda I \right]^{-1} = \left(I - J^T(x_n) J(x_n) / \lambda \right) \rightarrow I$$

Поиск экстремума 4: алгоритм Левенберга — Марквардта

- 1) При больших лямбдах мы движемся в направлении антиградиента.
- 2) Можем ускорить/замедлить темп там где градиент сильный/слабый - учтя кривизну, т.е. заменив $\lambda I_{n \times n}$ на $\text{diag}(J^T J)$ (тогда будем идти по основанию оврага).

$$x_{n+1} = x_n - \left[J^T(x_n) J(x_n) + \lambda \underbrace{I}_{\text{суммирует матр.}} \right]^{-1} J^T(x_n) r(x_n)$$

Поиск экстремума 4: алгоритм Левенберга — Марквардта

- 1) При больших лямбдах мы движемся в направлении антиградиента.
- 2) Можем ускорить/замедлить темп там где градиент сильный/слабый - учтя кривизну, т.е. заменив $I_{\lambda} \text{ на } \text{diag}(J^T J)$ (тогда будем идти по основанию оврага).
- 3) Как выбрать λ ?
 - line search: увеличиваем λ в 10 раз пока не улучшим целевую функцию

Поиск экстремума 4: алгоритм Левенберга — Марквардта

- 1) При больших лямбдах мы движемся в направлении антиградиента.
- 2) Можем ускорить/замедлить темп там где градиент сильный/слабый - учтя кривизну, т.е. заменив $\lambda \mathbf{I}$ на $\text{diag}(J^T J)$ (тогда будем идти по основанию оврага).
- 3) Как выбрать λ ?
 - line search: увеличиваем λ в 10 раз пока не улучшим целевую функцию
 - увеличение λ не только делает шаг безопаснее
(тяготеет к градиентному спуску),
но и как влияет на длину шага?

$$x_{n+1} = x_n - [J^T(x_n) J(x_n) + \lambda \mathbf{I}]^{-1} J^T(x_n) r(x_n)$$

увеличивает длину шага

Поиск экстремума 4: алгоритм Левенберга — Марквардта

- 1) При больших лямбдах мы движемся в направлении антиградиента.
- 2) Можем ускорить/замедлить темп там где градиент сильный/слабый - учтя кривизну, т.е. заменив $\lambda \mathbf{I}$ на $\text{diag}(J^T J)$ (тогда будем идти по основанию оврага).
- 3) Как выбрать λ ?
 - line search: увеличиваем λ в 10 раз пока не улучшим целевую функцию
 - увеличение λ не только делает шаг безопаснее
(тяготеет к градиентному спуску),
но и делает шаг меньше
(т.е. более безопасным)

$$x_{n+1} = x_n - [J^T(x_n) J(x_n) + \lambda \mathbf{I}]^{-1} J^T(x_n) r(x_n)$$

увеличивает шаг.

Какие уже есть этапы?

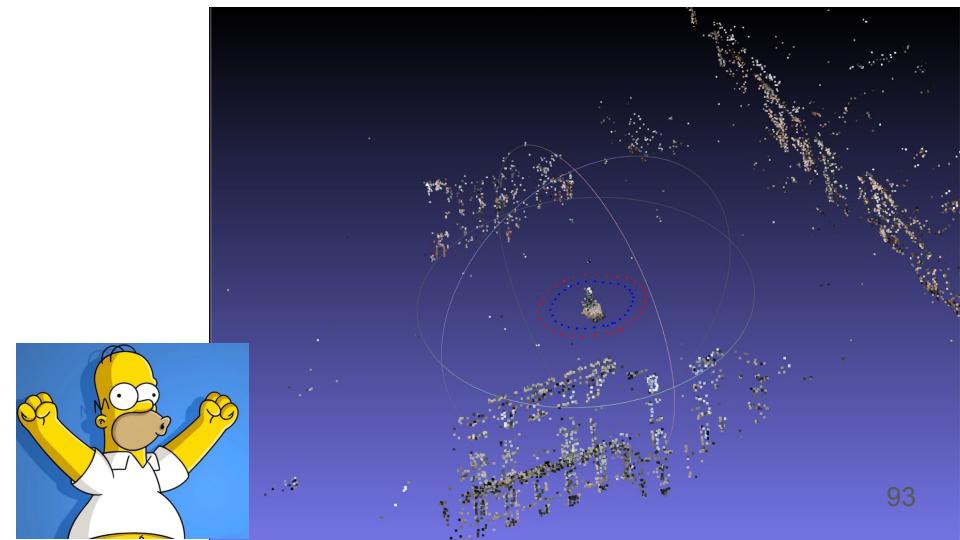
- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер

Какие уже есть этапы? **Что мы хотим сделать?**

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер

Какие уже есть этапы? Что мы хотим сделать?

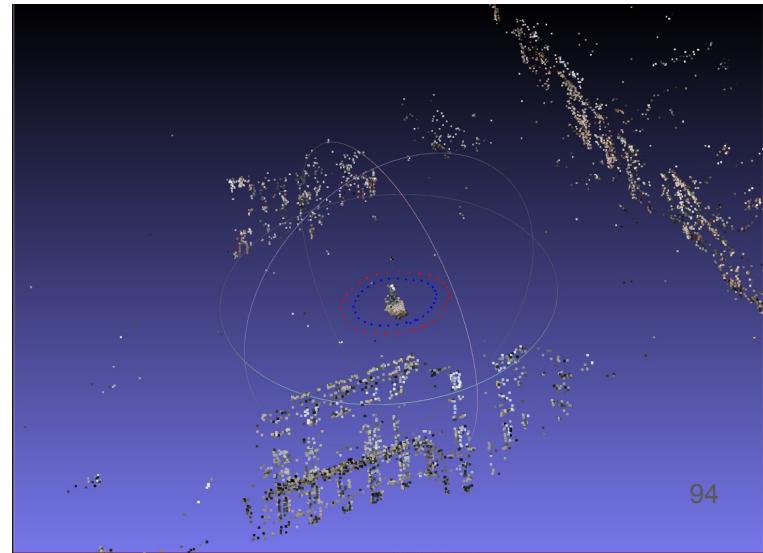
- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (**Bundle Adjustment**)



Какие уже есть этапы? Что мы хотим сделать?

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (Bundle Adjustment)

Как сформулировать какие **параметры** мы ищем
и **что** мы вообще оптимизируем?

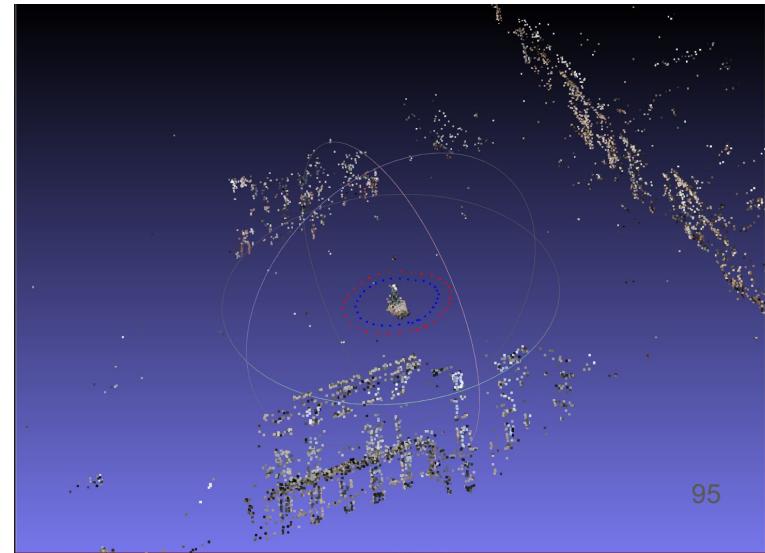


Какие уже есть этапы? Что мы хотим сделать?

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (Bundle Adjustment)

Чтобы оптимизировать эти **3D точки**
+ **параметры камер** нам нужна **функция невязки**.

Как ее определить?



Какие уже есть этапы? Что мы хотим сделать?

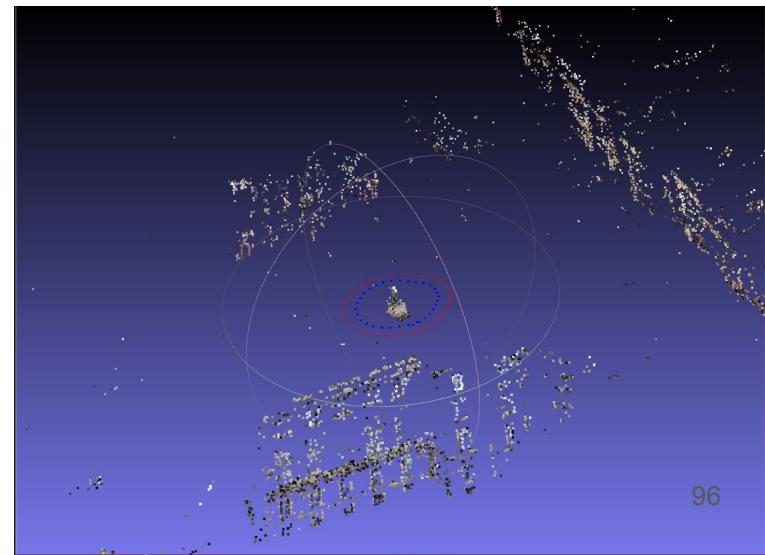
- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (Bundle Adjustment)

Чтобы оптимизировать эти **3D точки**

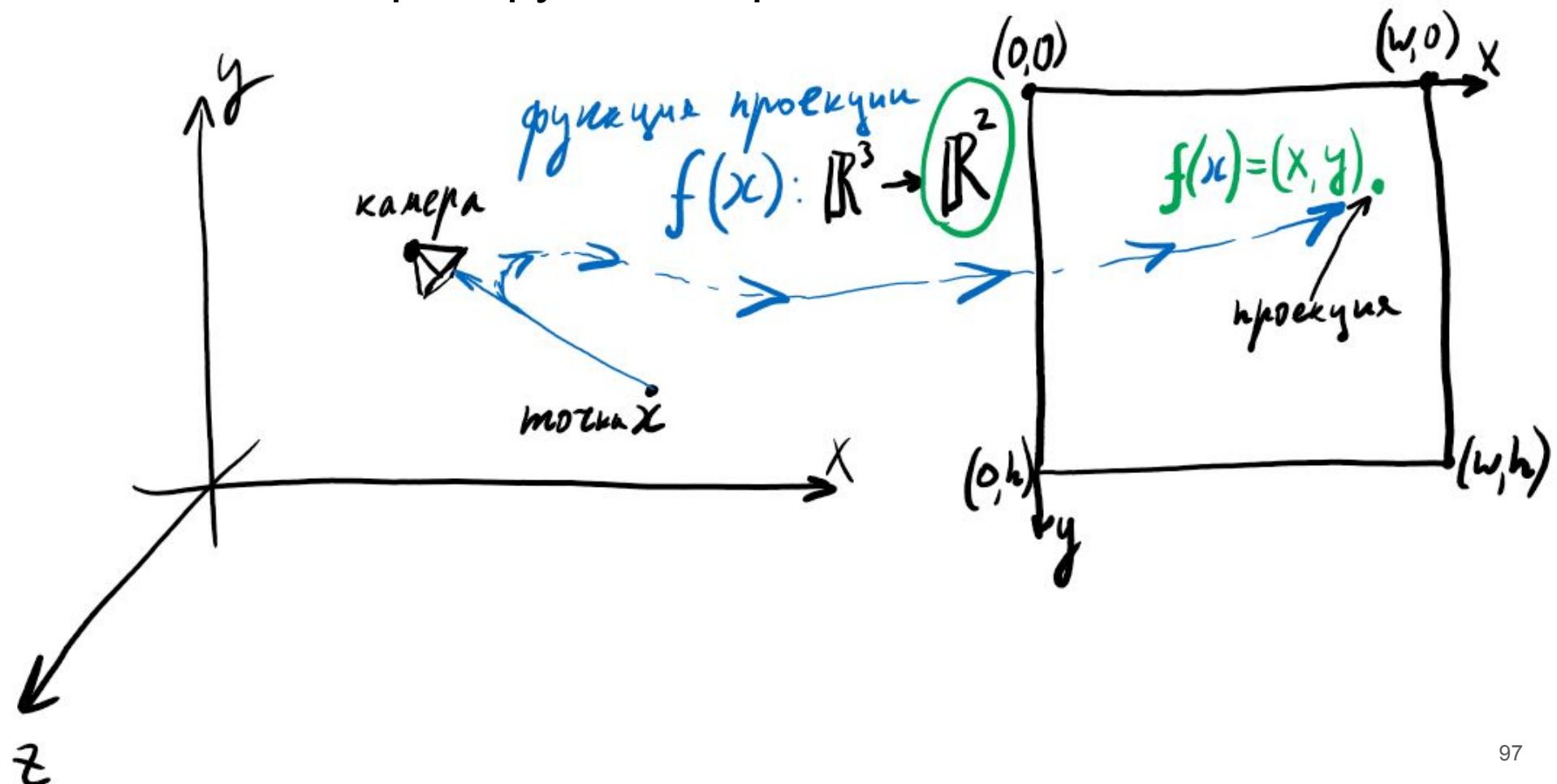
+ **параметры камер** нам нужна **функция невязки**.

Как ее определить?

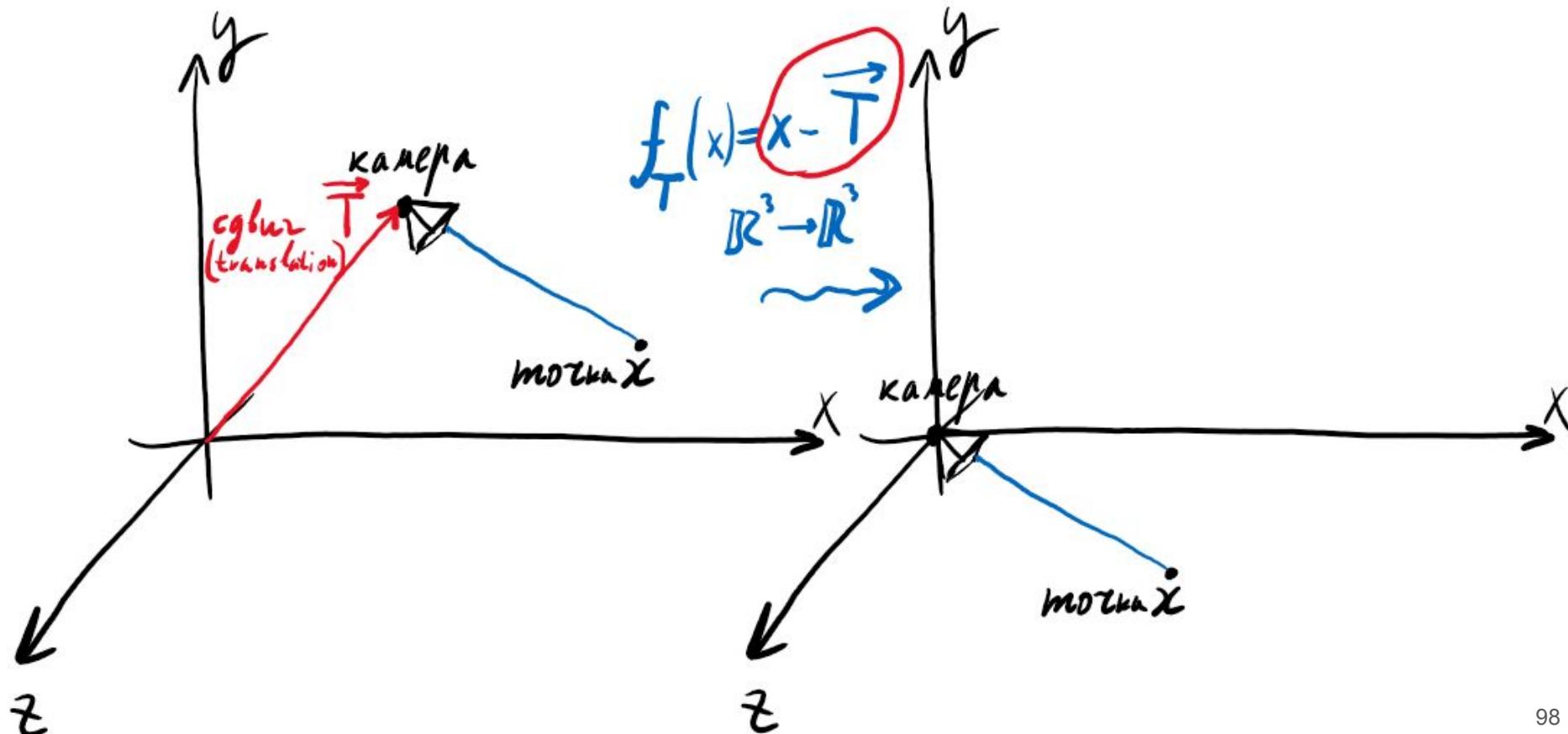
Введем функцию `project(3D точка) -> 2D пиксель`



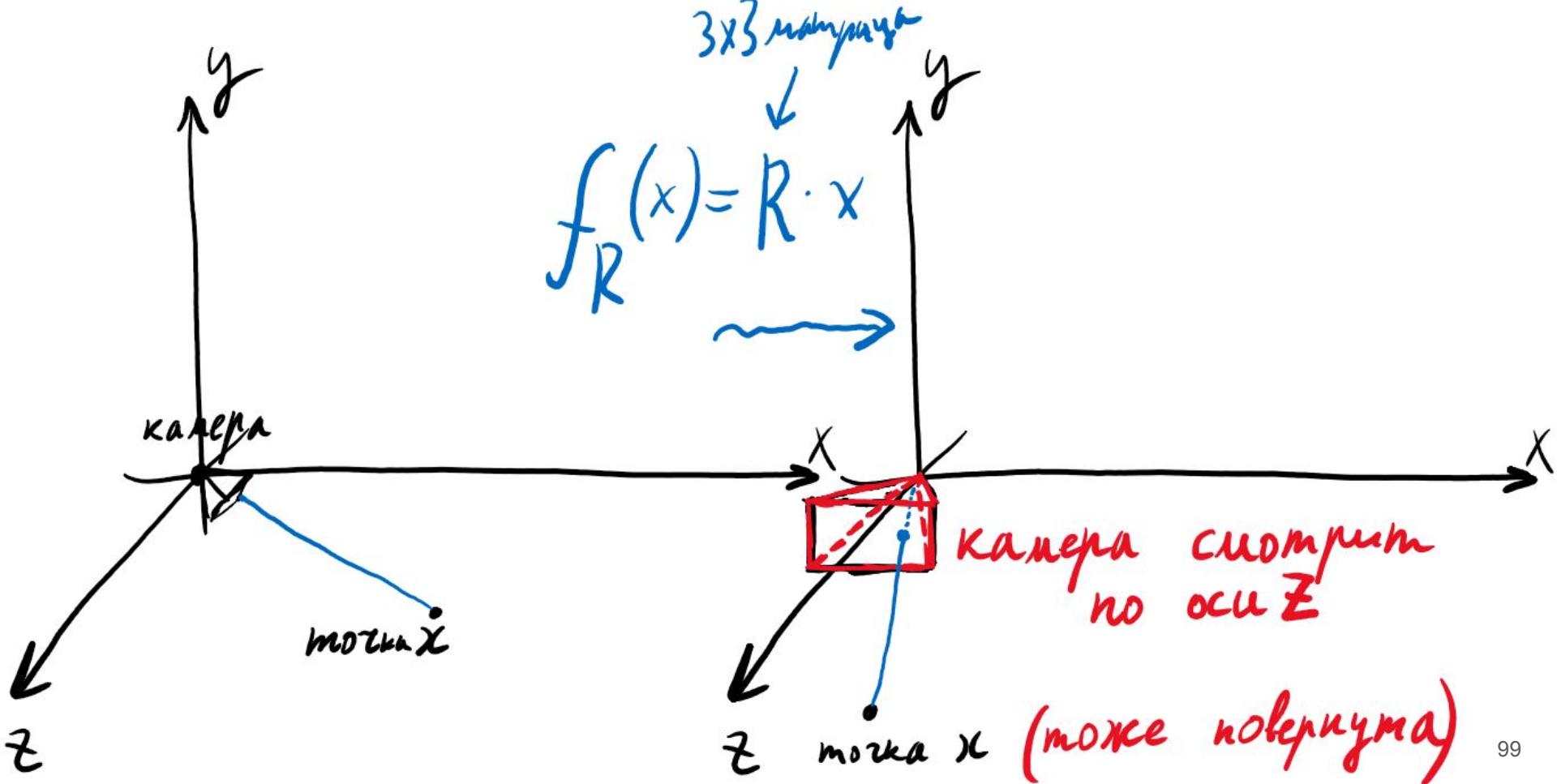
Модель камеры: функция проекции



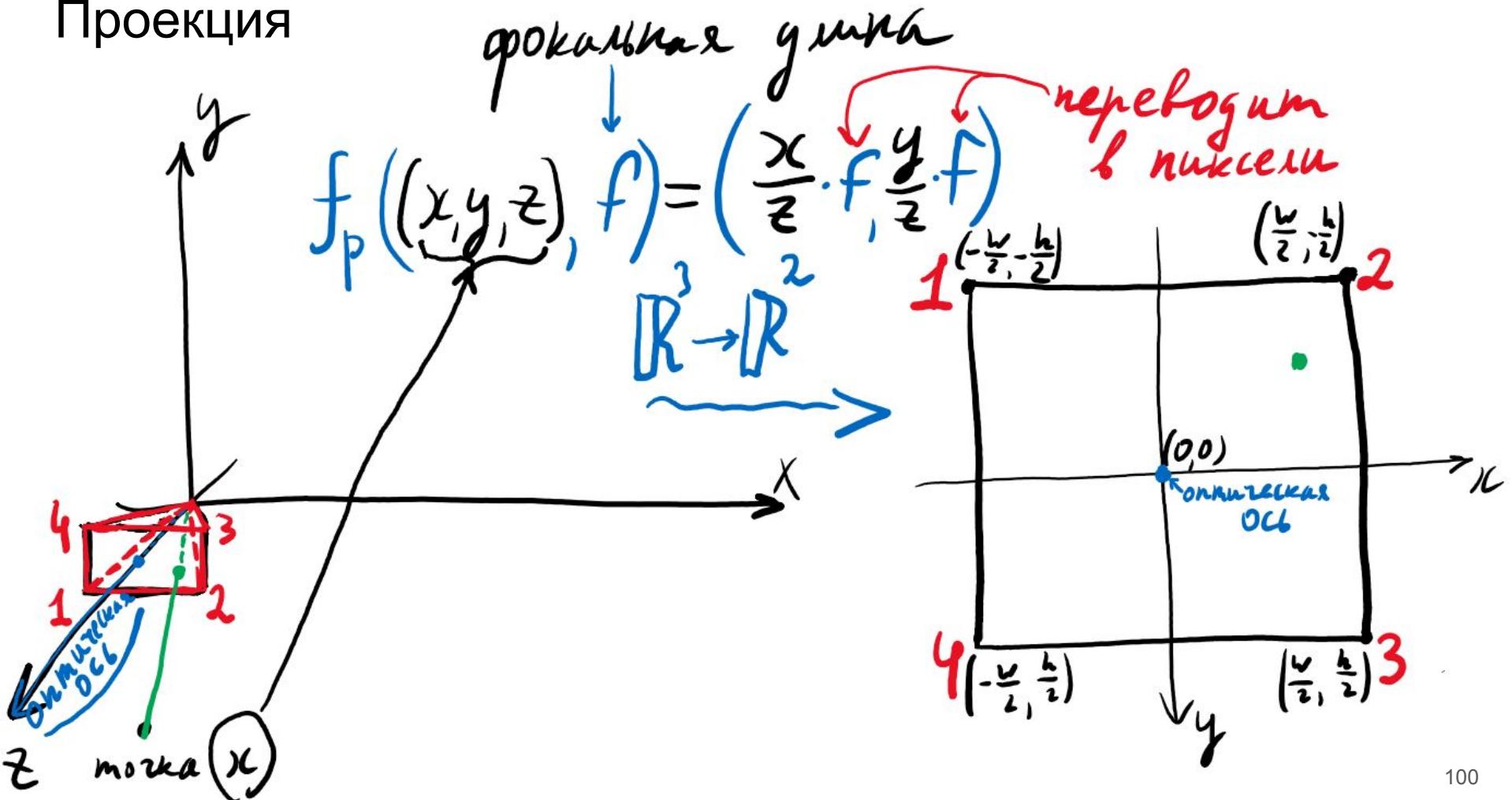
Модель камеры: функция проекции (сдвиг)



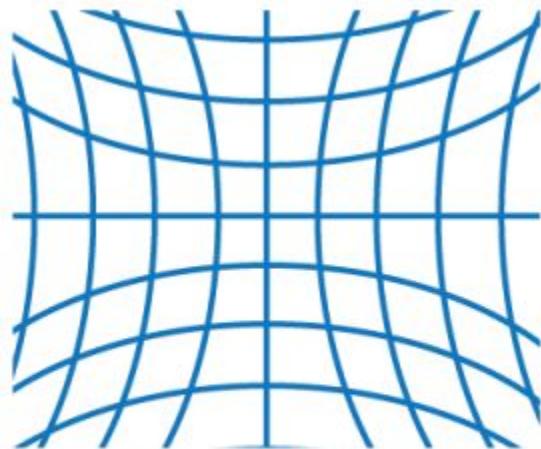
Модель камеры: функция проекции (поворот)



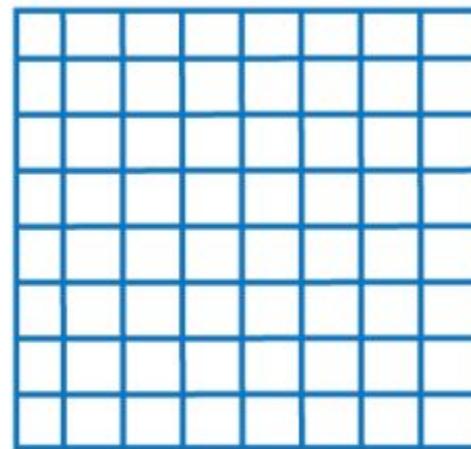
Проекция



Проекция: учет радиальных искажений



negative radial distortion
"pincushion"

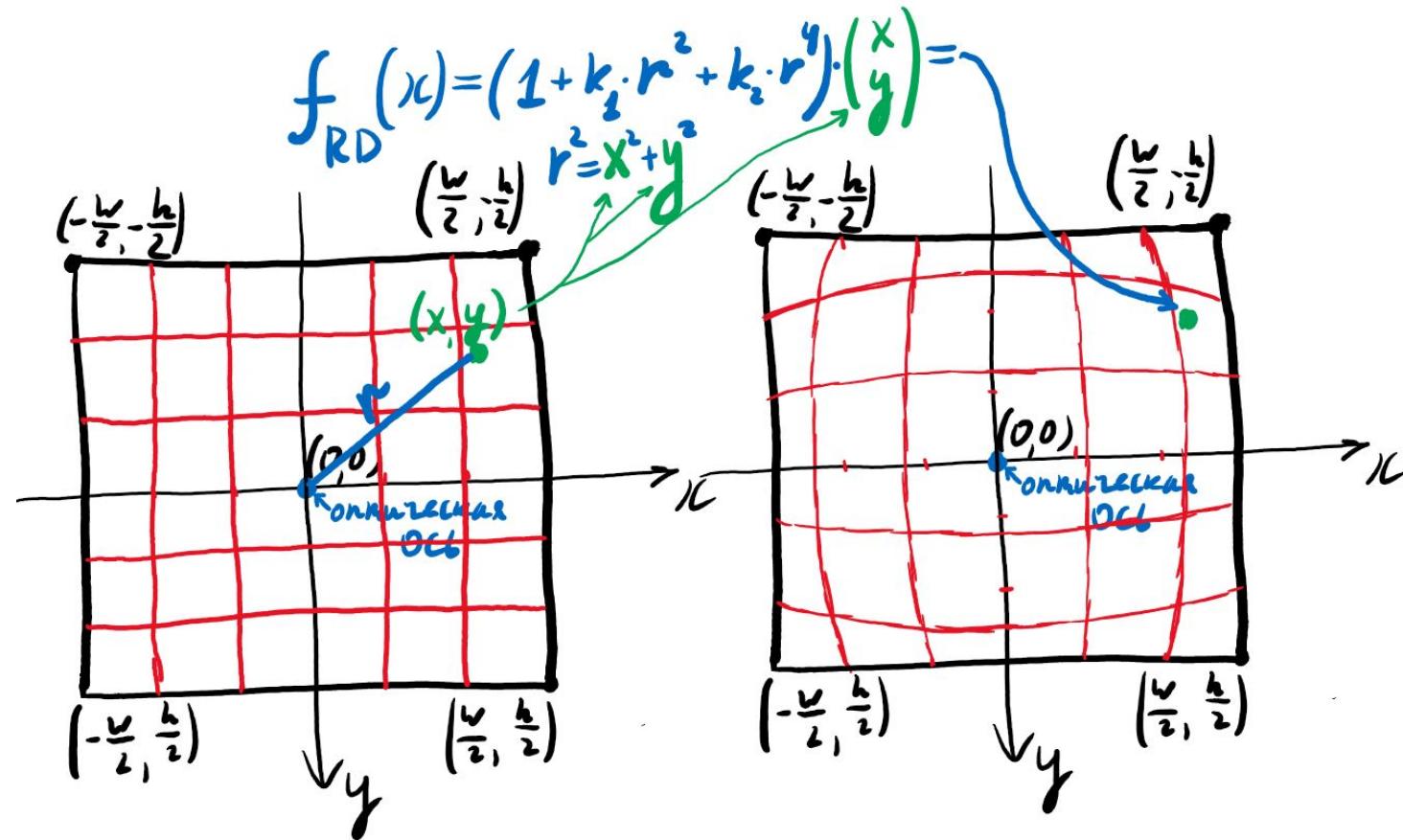


no distortion

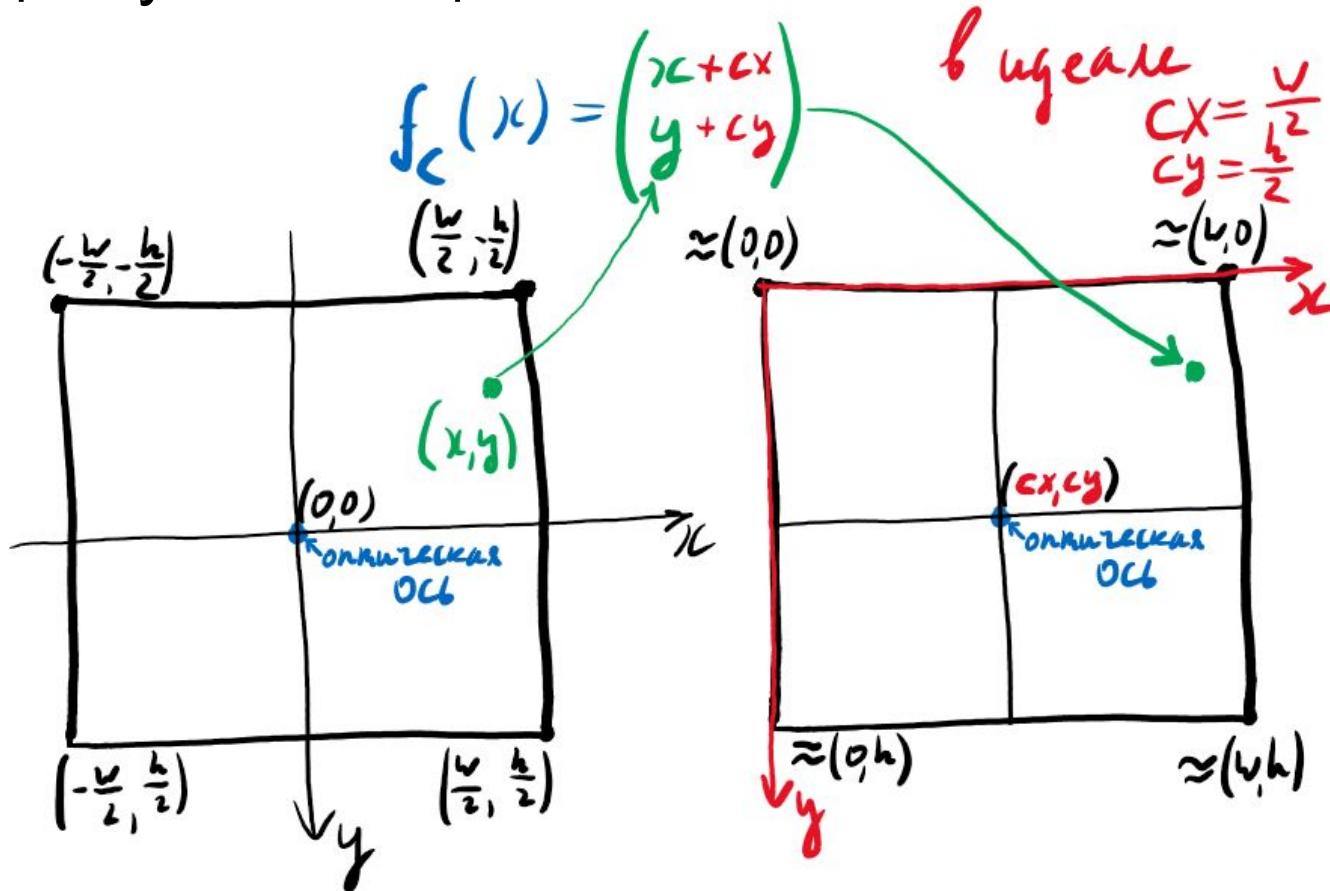


positive radial distortion
"barrel"

Проекция: учет радиальных искажений



Проекция: учет смещения оптической оси



Переход в локальную систему камеры: сдвиг

Сколько параметров?

Переход в локальную систему камеры: сдвиг

Переход в локальную систему камеры: поворот

Сколько параметров?

Переход в локальную систему камеры: сдвиг

Переход в локальную систему камеры: поворот

Проекция: учет фокальной длины

Сколько параметров?

Переход в локальную систему камеры: сдвиг

Переход в локальную систему камеры: поворот

Проекция: учет фокальной длины (f)

Переход в локальную систему камеры: сдвиг

Переход в локальную систему камеры: поворот

Проекция: учет фокальной длины (f)

Проекция: учет радиальных искажений

Сколько параметров?

Переход в локальную систему камеры: сдвиг

Переход в локальную систему камеры: поворот

Проекция: учет фокальной длины (f)

Проекция: учет радиальных искажений (k_1, k_2)

Переход в локальную систему камеры: сдвиг

Переход в локальную систему камеры: поворот

Проекция: учет фокальной длины (f)

Проекция: учет радиальных искажений (k_1, k_2)

Проекция: учет смещения оптической оси

Сколько параметров?

Переход в локальную систему камеры: сдвиг

Переход в локальную систему камеры: поворот

Проекция: учет фокальной длины (f)

Проекция: учет радиальных искажений (k_1, k_2)

Проекция: учет смещения оптической оси (cx, cy)

$$f_T(x) = x - \bar{T}$$

$$f_R(x) = \underline{R} \cdot x$$

$$f_p\left(\frac{x}{z}\right) = \left(\frac{\underline{f} \cdot \frac{x}{z}}{\underline{f} \cdot \frac{y}{z}}\right)$$

$$f_{RD}\left(\frac{x}{y}\right) = \left(1 + \underbrace{k_1 r^2}_{k_2 r^4}\right) \begin{pmatrix} x \\ y \end{pmatrix}, \quad \begin{matrix} r^2 = x^2 + y^2 \\ z = 1 \end{matrix}$$

$$f_C\left(\frac{x}{y}\right) = \begin{pmatrix} x + \underline{C_x} \\ y + \underline{C_y} \end{pmatrix}$$

напоминки:

\bar{T}

- координаты камеры
(3 единицы измер.)

\underline{R}

- 3 единицы измер.
 3×3 матрица изображения
(изображения)

f

- фокальная глубина
(бесконечн.)

k_1, k_2

- коэффициенты
радиальной
дисторсии

C_x, C_y

- смещение
области обзора
оку

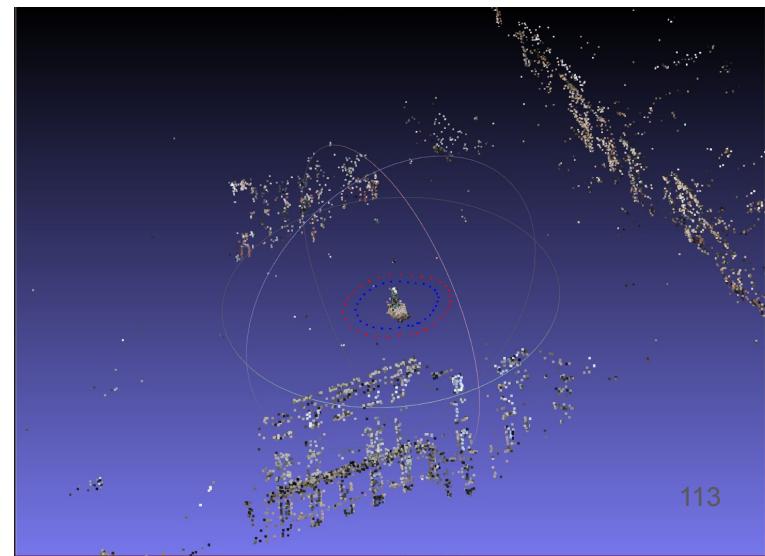
Bundle Adjustment

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (**Bundle Adjustment**)

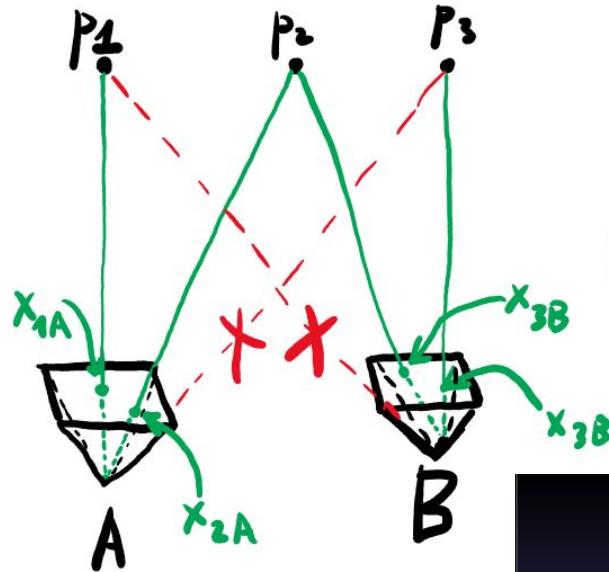
Чтобы оптимизировать эти **3D точки**
+ **параметры камер** нам нужна **функция невязки**.

Ввели функцию **project(3D точка) -> 2D пиксель**

Как определить функцию невязки?



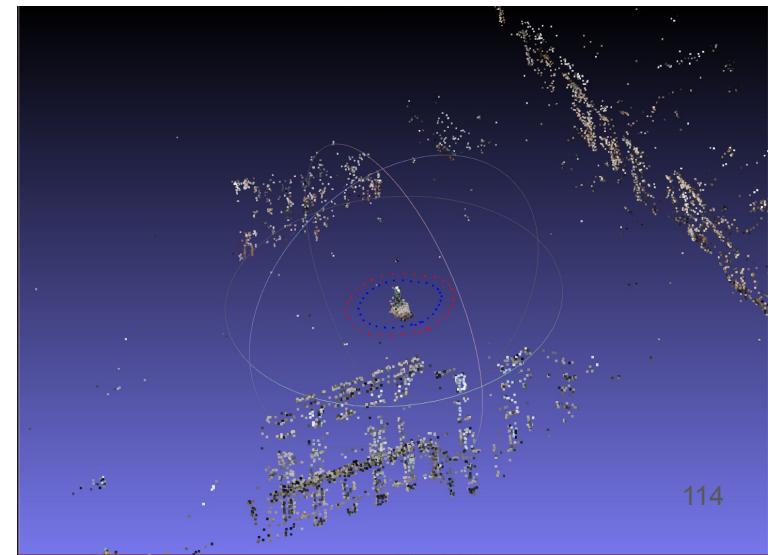
Bundle Adjustment



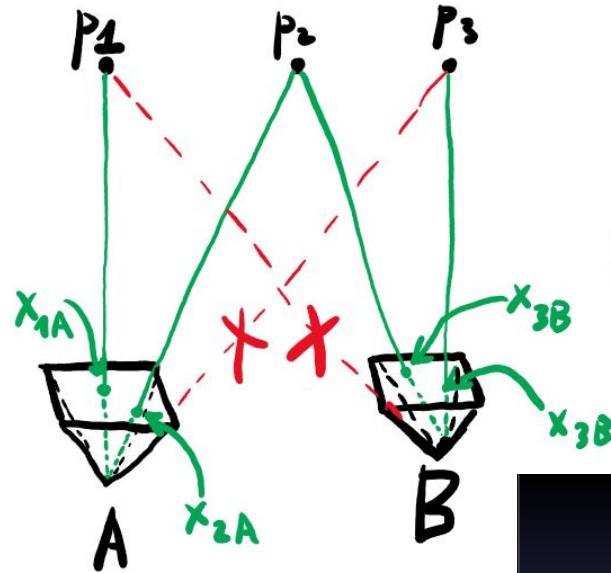
Чтобы оптимизировать эти **3D точки**
+ **параметры камер** нам нужна **функция невязки**.

Ввели функцию **project(3D точка) -> 2D пиксель**

Как определить функцию невязки?



Bundle Adjustment

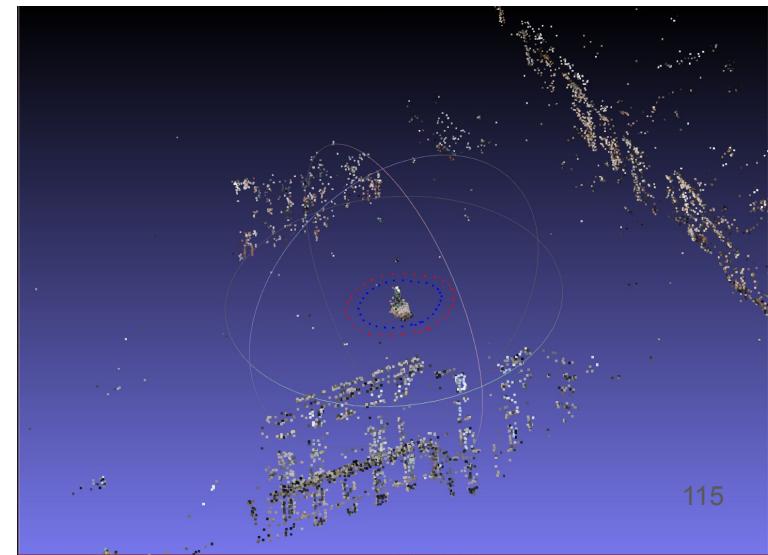


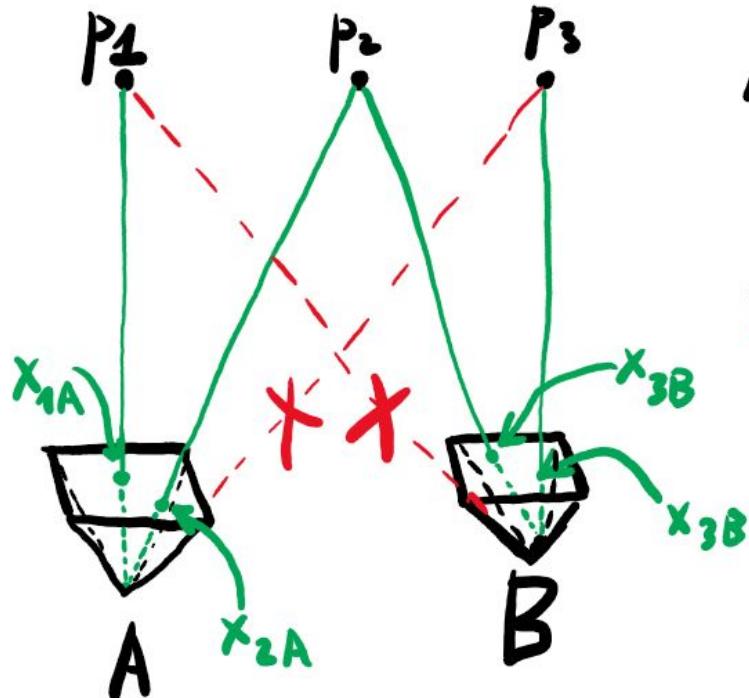
Чтобы оптимизировать эти **3D точки**
+ **параметры камер** нам нужна **функция невязки**.

Ввели функцию **project(3D точка) -> 2D пиксель**

Нужна ли нам функция **unproject(2D) -> 3D?**

Как определить функцию невязки?





проекция $g(x) : \mathbb{R}^3 \rightarrow \mathbb{R}^2$

небольшие (residuals):

$$r_{1A}(x) = g_A(P_1) - x_{1A}$$

$$r_{1B}(x) = g_B(P_1) - x_{1B}$$

$$r_{2A}(x) = g_A(P_2) - x_{2A}$$

$$r_{2B}(x) = g_B(P_2) - x_{2B}$$

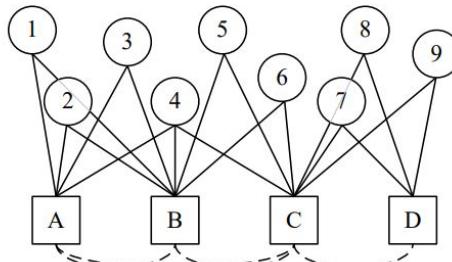
М наблюдений

минимизируя

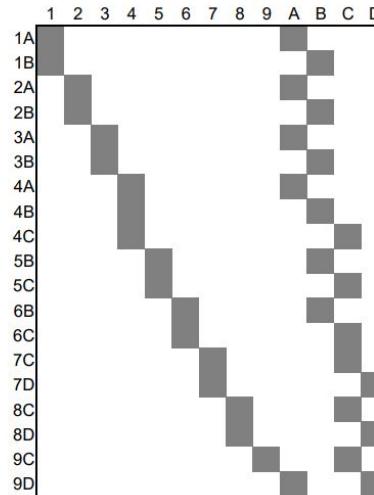
$$f(x) = \frac{1}{2} \sum_{i,j=1 \dots m} (g_j(P_i) - x_i)^2$$

Structure from motion: structure of the points, motion of the cameras.

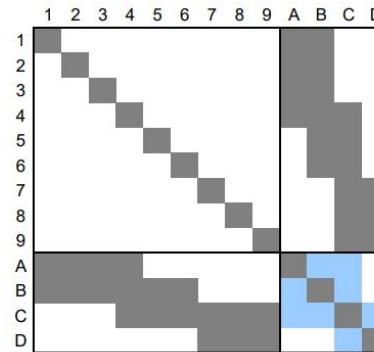
Bundle - связка пучков-лучей из центров камер к 3D точкам, эту связку итеративно улучшают (**Adjustment**) через минимизацию ошибки репроекции.



(a)



(b)



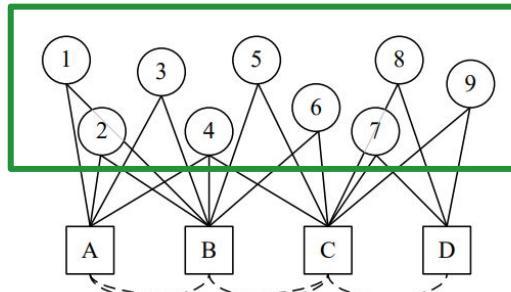
(c)

Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point) variables are eliminated.

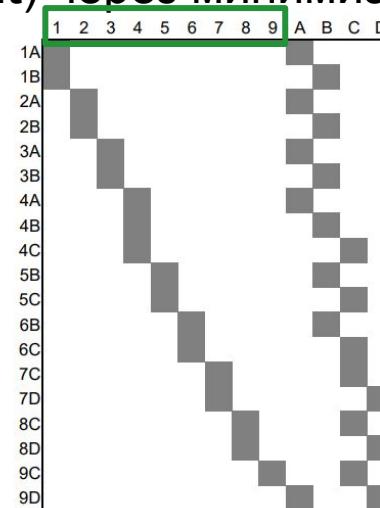
Structure from motion: structure of the points, motion of the cameras.

Bundle - связка пучков-лучей из центров камер к 3D точкам, эту связку итеративно улучшают (**Adjustment**) через минимизацию ошибки репроекции.

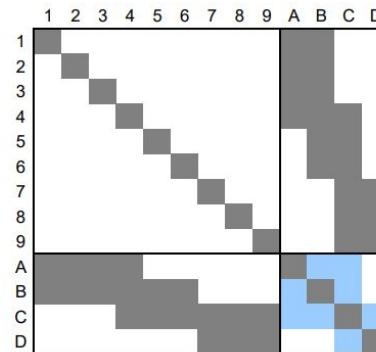
3D ключевые точки



(a)



(b)



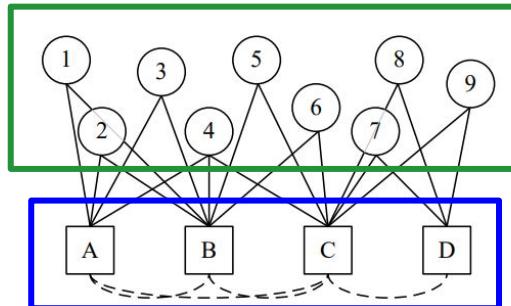
(c)

Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point) variables are eliminated.

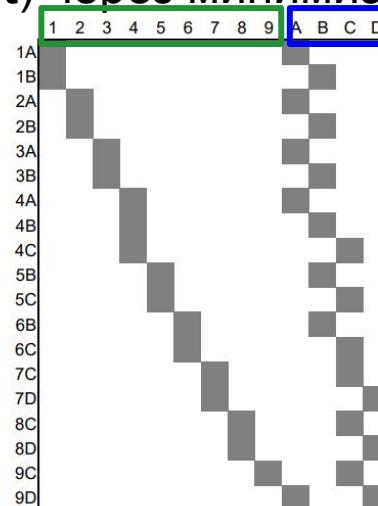
Structure from motion: structure of the points, motion of the cameras.

Bundle - связка пучков-лучей из центров камер к 3D точкам, эту связку итеративно улучшают (**Adjustment**) через минимизацию ошибки репроекции.

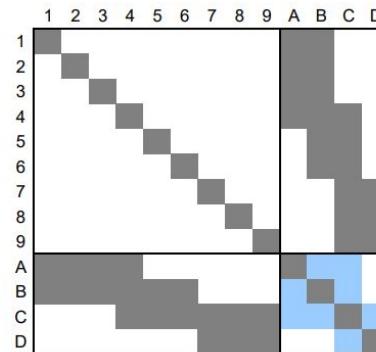
3D ключевые точки



Параметры камер



(b)



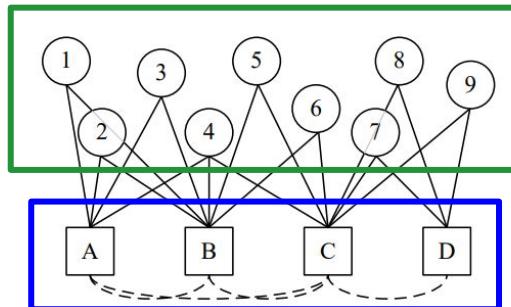
(c)

Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point) variables are eliminated.

Structure from motion: structure of the points, motion of the cameras.

Bundle - связка пучков-лучей из центров камер к 3D точкам, эту связку итеративно улучшают (**Adjustment**) через минимизацию ошибки репроекции.

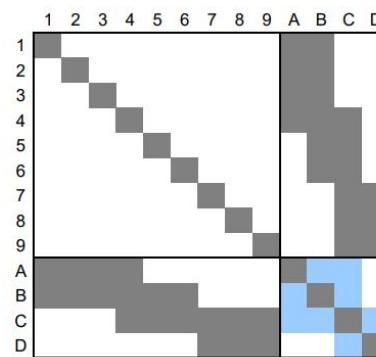
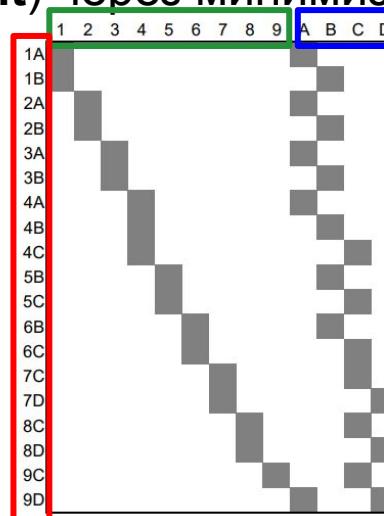
3D ключевые точки



Параметры камер

???

(b)



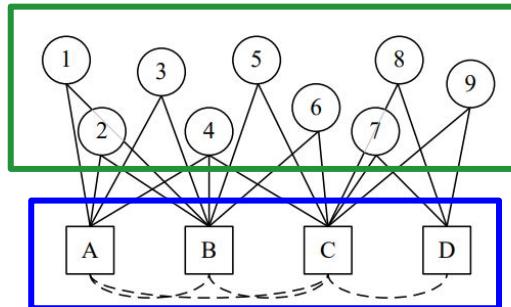
(c)

Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point) variables are eliminated.

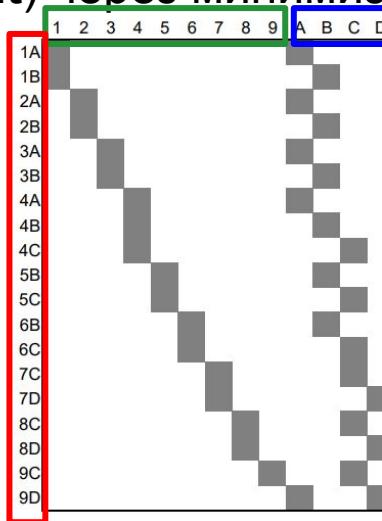
Structure from motion: structure of the points, motion of the cameras.

Bundle - связка пучков-лучей из центров камер к 3D точкам, эту связку итеративно улучшают (**Adjustment**) через минимизацию ошибки репроекции.

3D ключевые точки

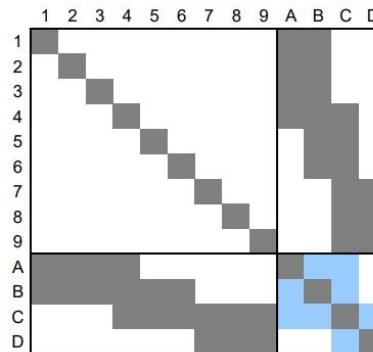


Параметры камер



(b)

Ожидаемые проекции (наблюдения)



(c)

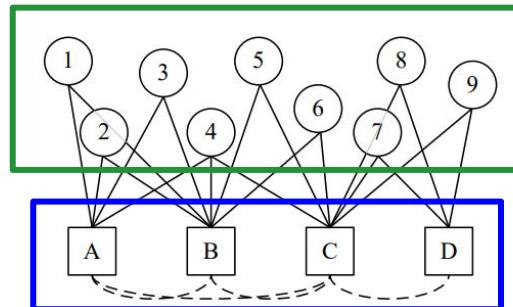
Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point) variables are eliminated.

Structure from motion: structure of the points, motion of the cameras.

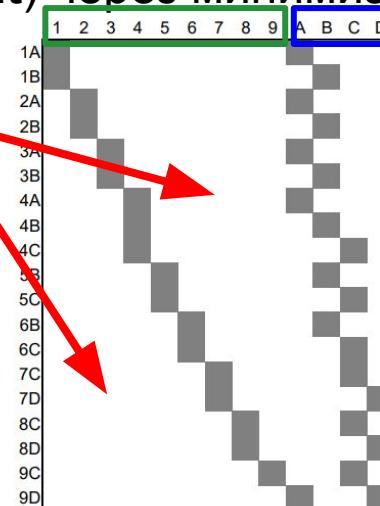
Bundle - связка пучков-лучей из центров камер к 3D точкам, эту связку итеративно улучшают (**Adjustment**) через минимизацию ошибки репроекции.

Почему Якобиан почти всюду из нулей?

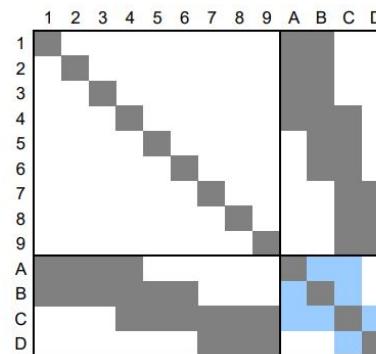
3D ключевые точки



Параметры камер



(b)



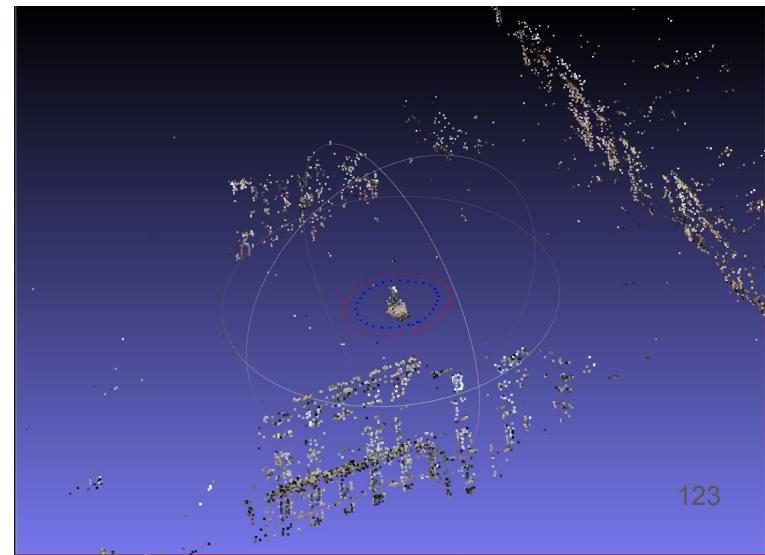
(c)

Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point) variables are eliminated.

Какие уже есть этапы? Что мы хотим сделать?

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (Bundle Adjustment)

Итак какие **параметры** мы ищем
и **что** мы вообще оптимизируем?

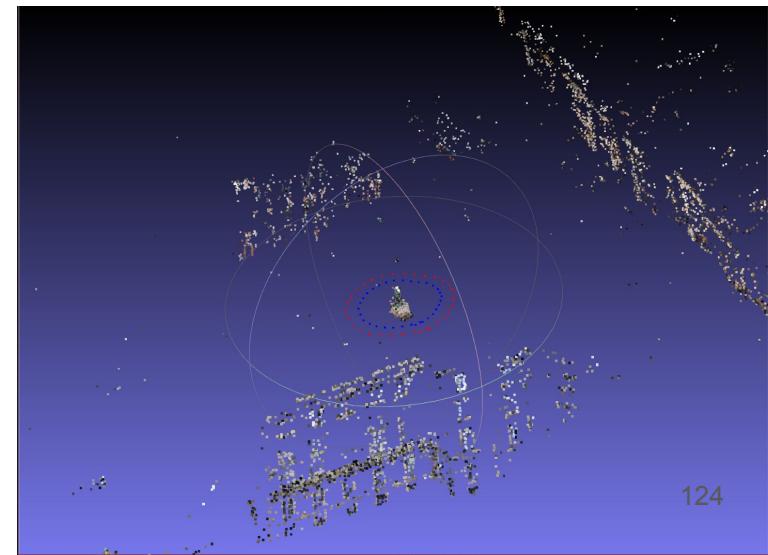


Какие уже есть этапы? Что мы хотим сделать?

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (Bundle Adjustment)

Итак какие **параметры** мы ищем
и **что** мы вообще оптимизируем?

Есть функция **project(3D точка) -> 2D пиксель**
зачем она нам была нужна?

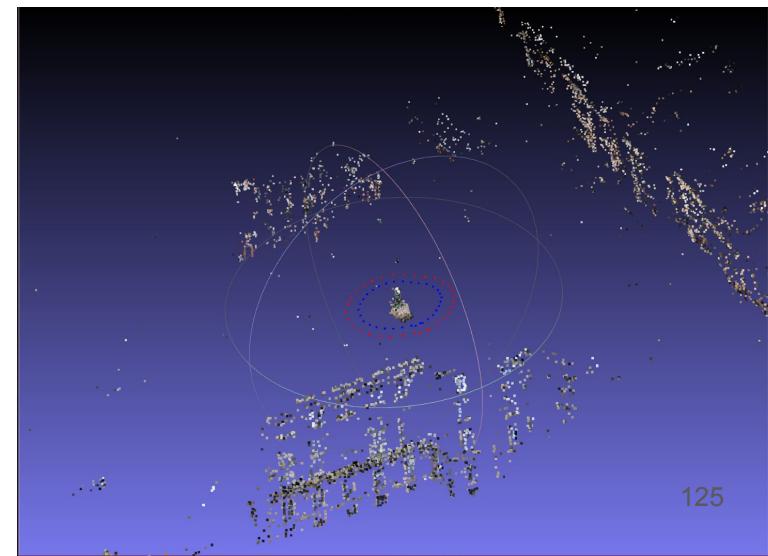


Какие уже есть этапы? Что мы хотим сделать?

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (Bundle Adjustment)

Итак какие **параметры** мы ищем
и **что** мы вообще оптимизируем?

Есть функция **project(3D точка) -> 2D пиксель**
она позволяет сформулировать невязку



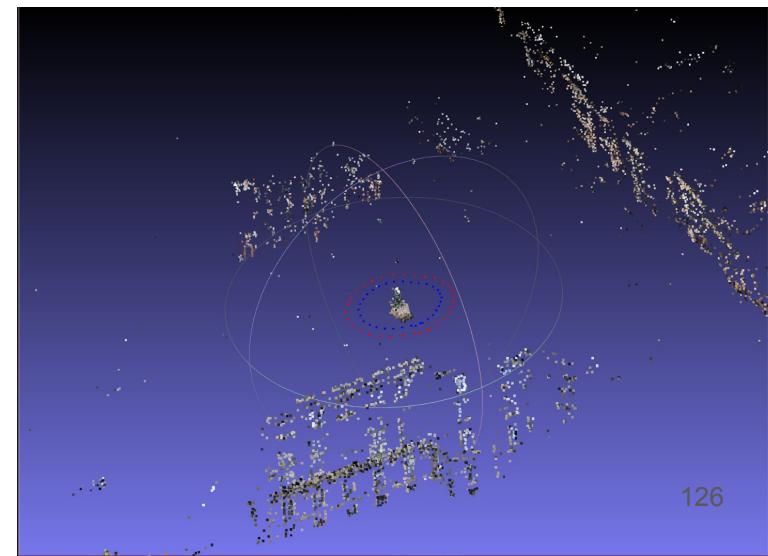
Какие уже есть этапы? Что мы хотим сделать?

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (Bundle Adjustment)

Итак какие **параметры** мы ищем
и **что** мы вообще оптимизируем?

Есть функция **project(3D точка) -> 2D пиксель**
она позволяет сформулировать невязку

А что еще нужно для Bundle Adjustment?



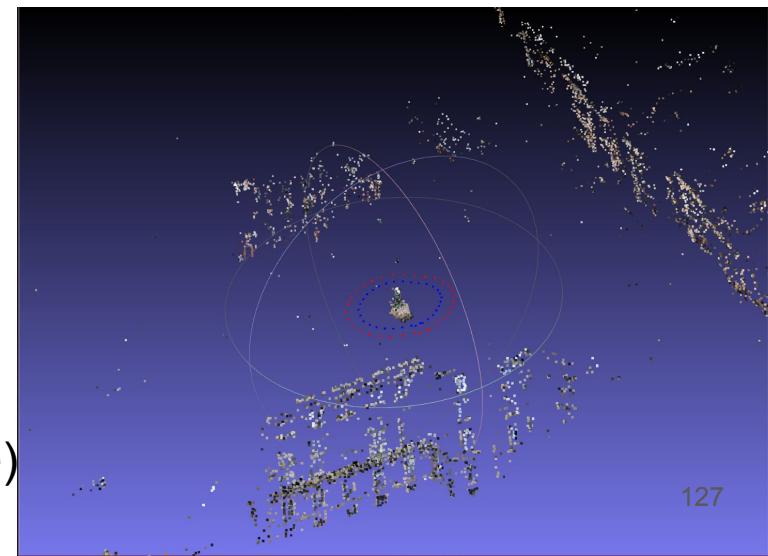
Какие уже есть этапы? Что мы хотим сделать?

- 1) Ключевые точки (SIFT)
- 2) Сопоставление ключевых точек (RANSAC, K-ratio test, left-right, cluster filtering)
- 3) Определение взаимного расположения пар камер
- 4) Уточнение расположения камер и их калибровок (Bundle Adjustment)

Итак какие **параметры** мы ищем
и **что** мы вообще оптимизируем?

Есть функция **project(3D точка) -> 2D пиксель**
она позволяет сформулировать невязку

Осталось лишь вычислить Якобиан! (производные)



Ссылки

Две ключевые книги-библии:

- Multiple View Geometry in Computer Vision, Richard Hartley & Andrew Zisserman (здесь особенно много деталей, в т.ч. про L-M)
- Computer Vision: Algorithms and Applications, Richard Szeliski

Статьи про детали ВА:

- Bundle adjustment—a modern synthesis, Triggs et al.
- Bundle Adjustment in the Large, Agarwal et. al.

Лекции:

- youtube/Behnam Asadi: про методы оптимизации и ВА
- youtube/Cyrill Stachniss: ВА 1, ВА 2
- coursera/Robotics: Perception (BA)
- лекция 2023 прочтения (есть про **Дополнение Шура - Schur Complement**)

Вопросы?



Полярный Николай
polarnick239@gmail.com¹²⁹