

Введение в фотограмметрию

Уточнение положения камер и автокалибровка

Фотограмметрия. Лекция 8



- Методы оптимизации
- Bundle Adjustment
- Schur complement
- Фильтрация связующих точек

Симиутин Борис
simiyutin.boris@yandex.ru

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

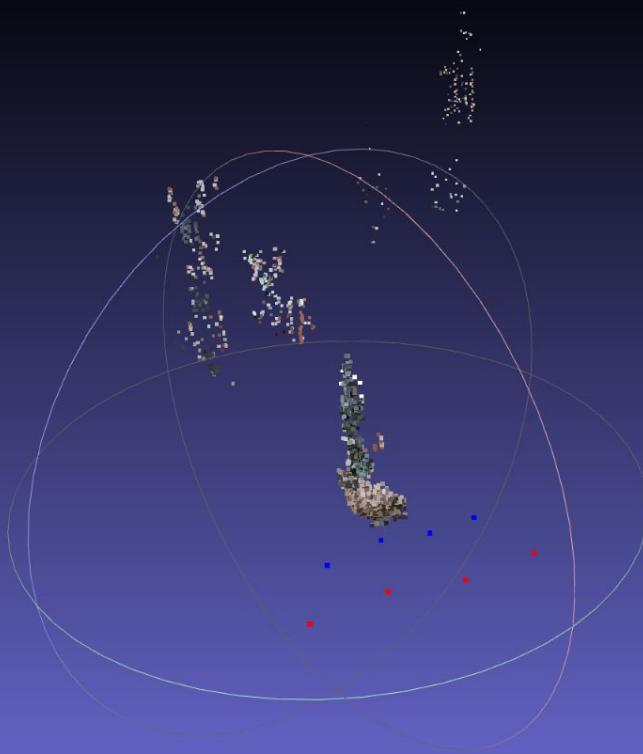
- 1) Научились с помощью решения линейных систем находить облака точек и положения камер
- 2) Из-за накопления ошибки и незнания калибровочных параметров камеры сцена очень быстро “разваливается” (не можем выполнить резекцию для новой камеры и триангулировать новые связующие точки)
- 3) Хотим сбрасывать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



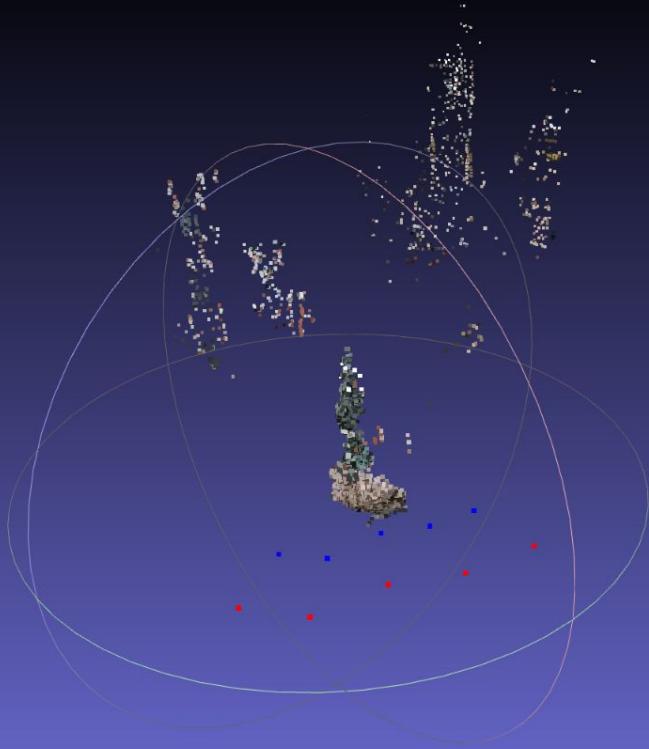
Триангуляция + резекция



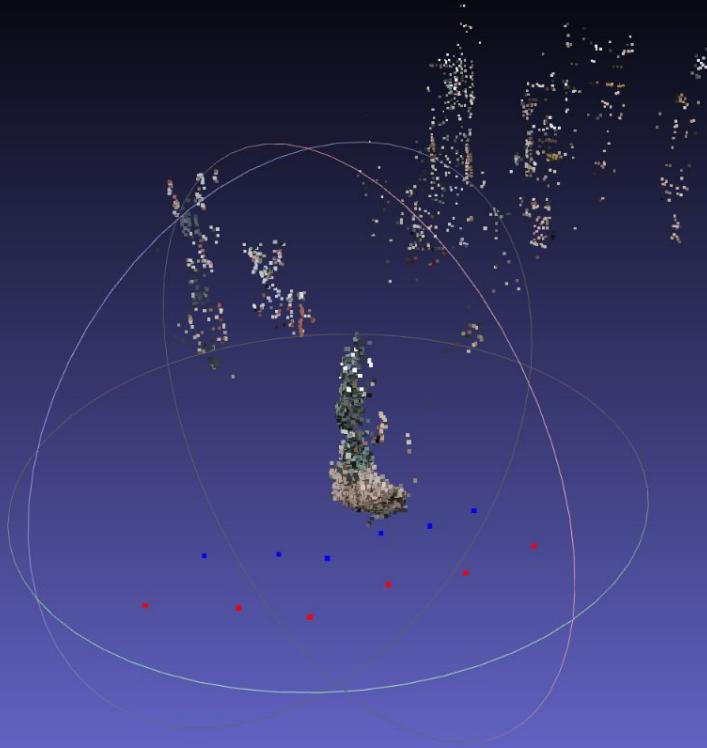
Триангуляция + резекция



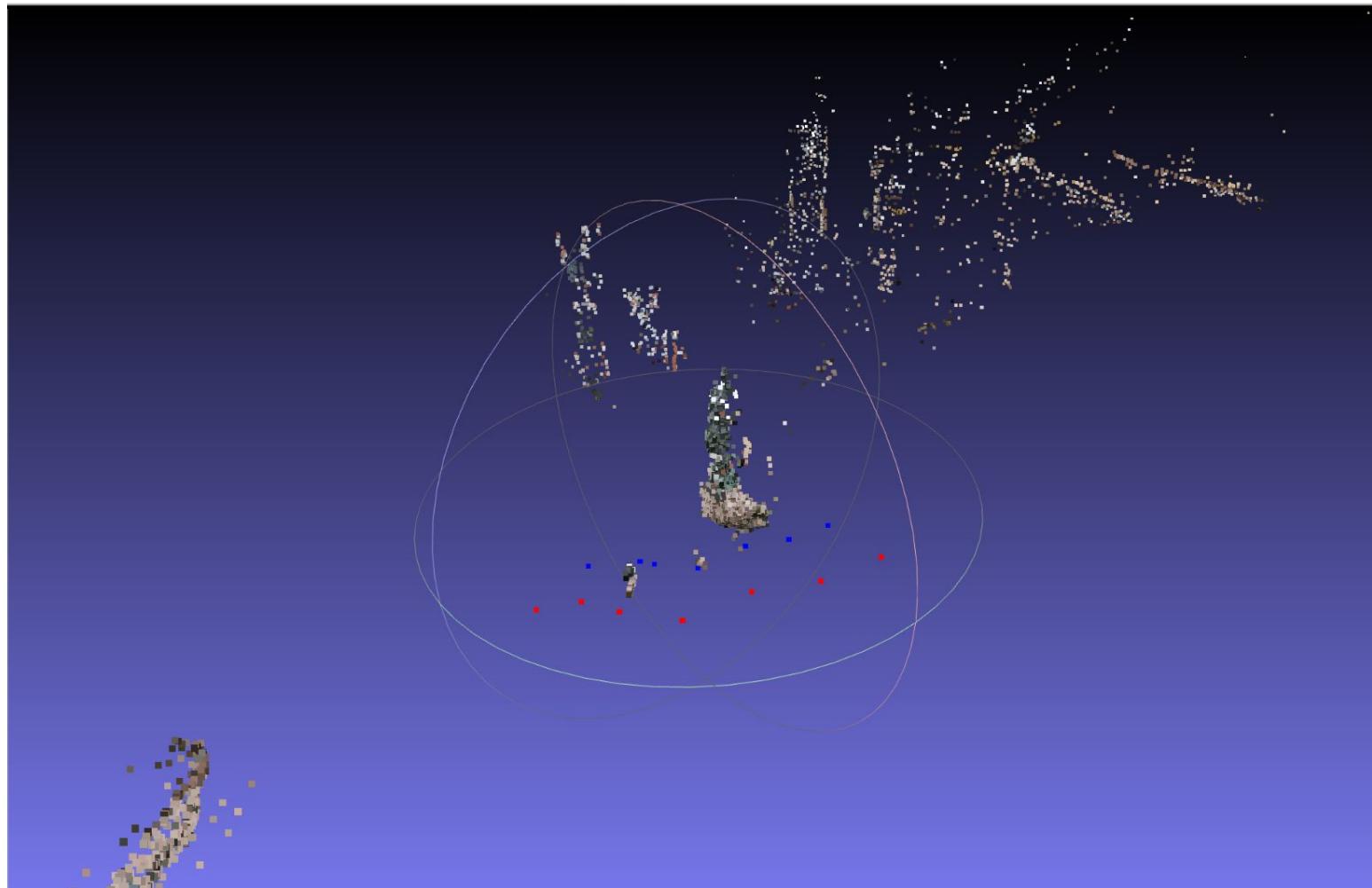
Триангуляция + резекция



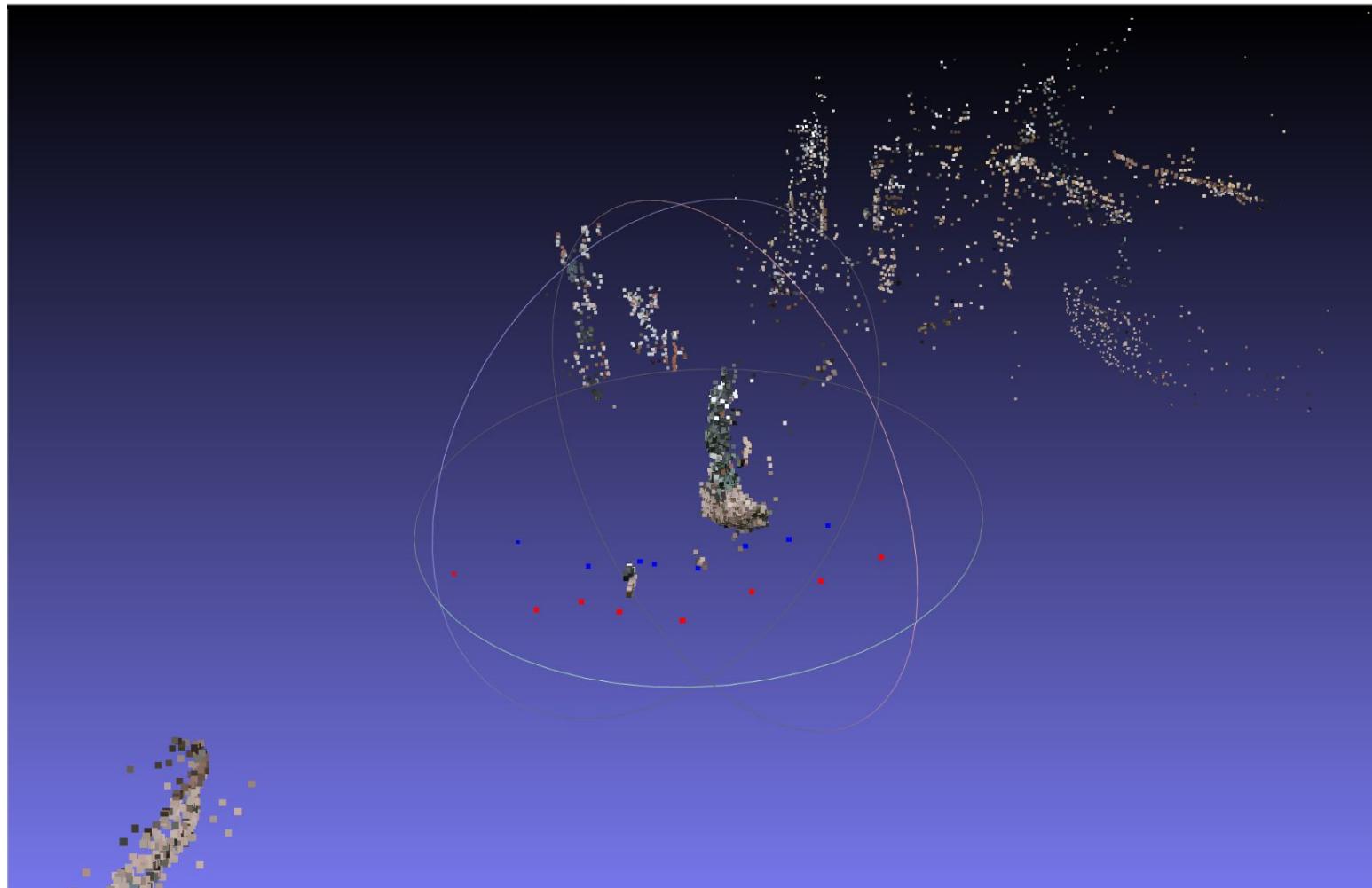
Триангуляция + резекция



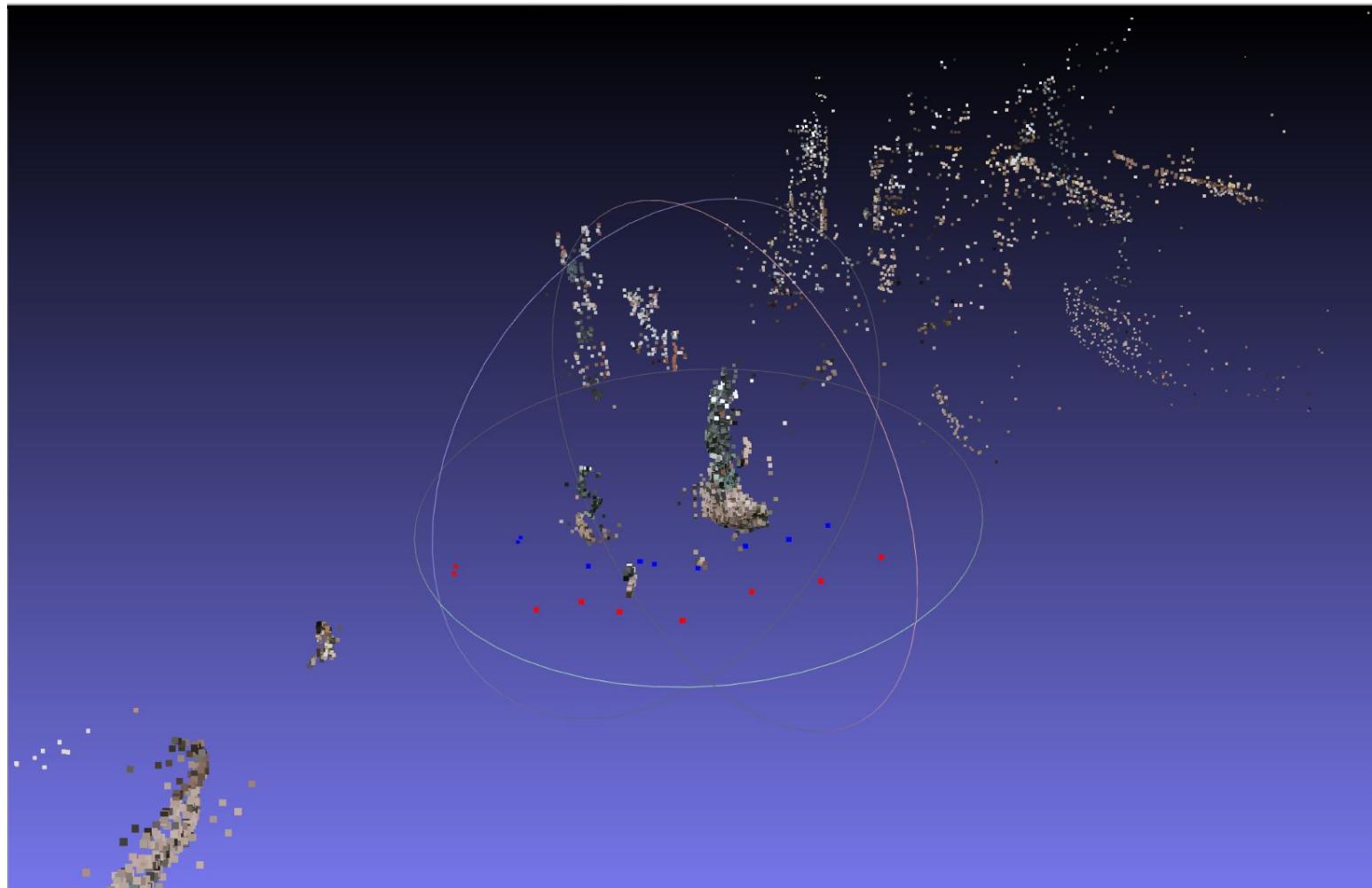
Триангуляция + резекция



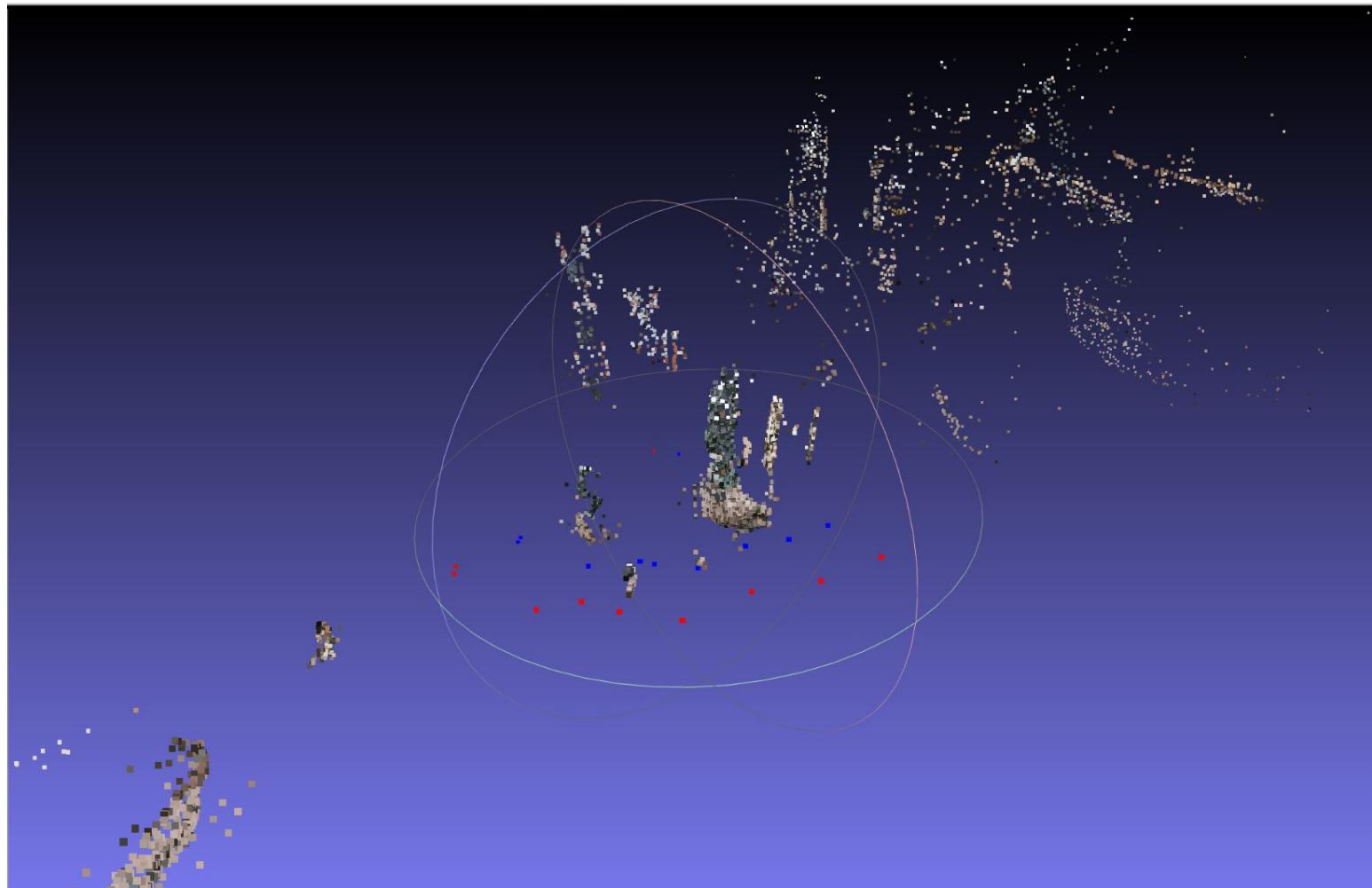
Триангуляция + резекция



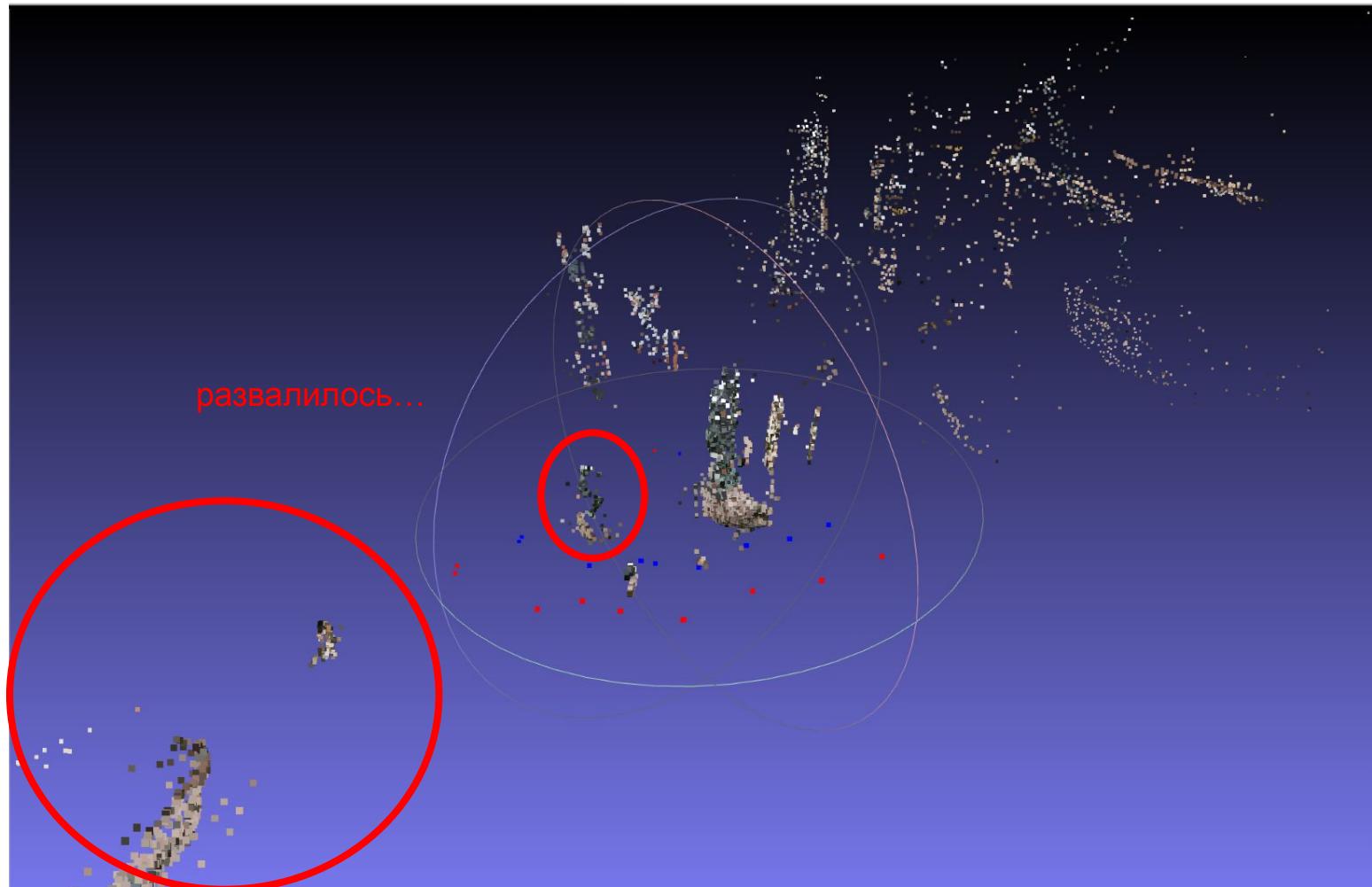
Триангуляция + резекция



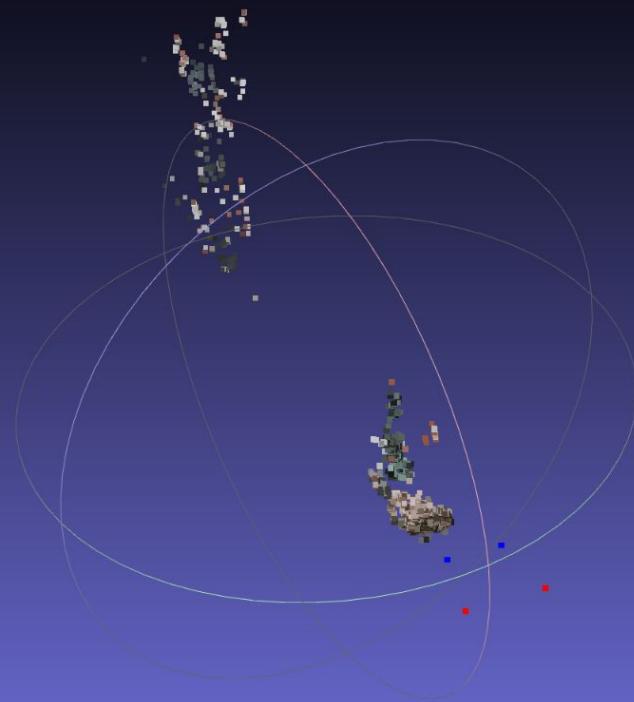
Триангуляция + резекция



Триангуляция + резекция

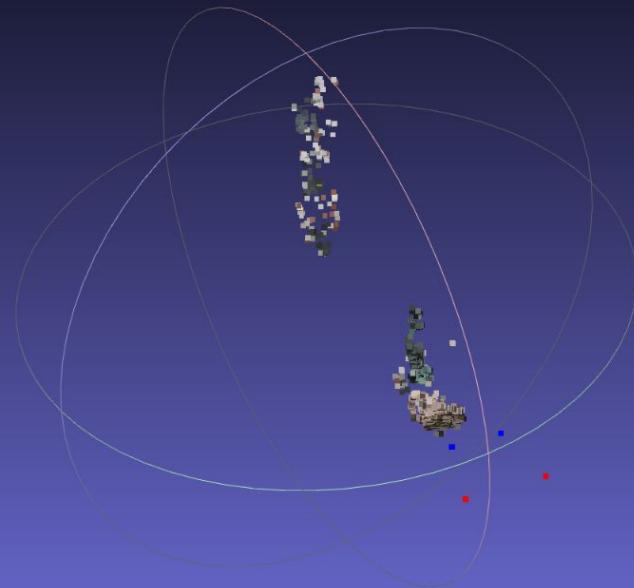


Триангуляция + резекция

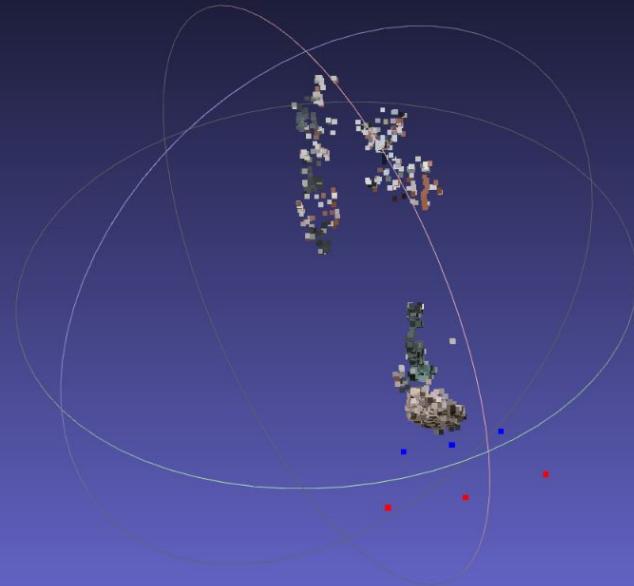


Триангуляция, резекция + уточнение положения камер и точек

BA

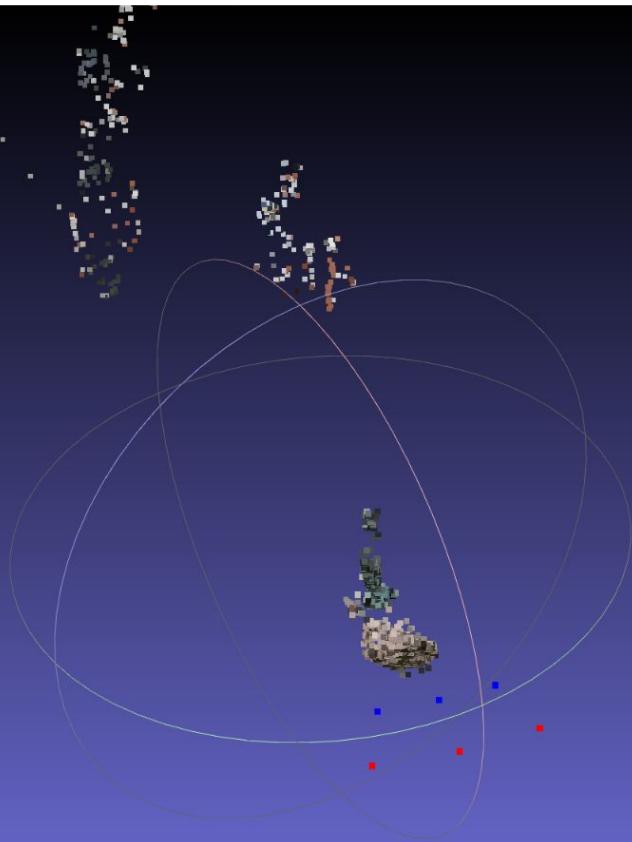


Триангуляция, резекция + уточнение положения камер и точек

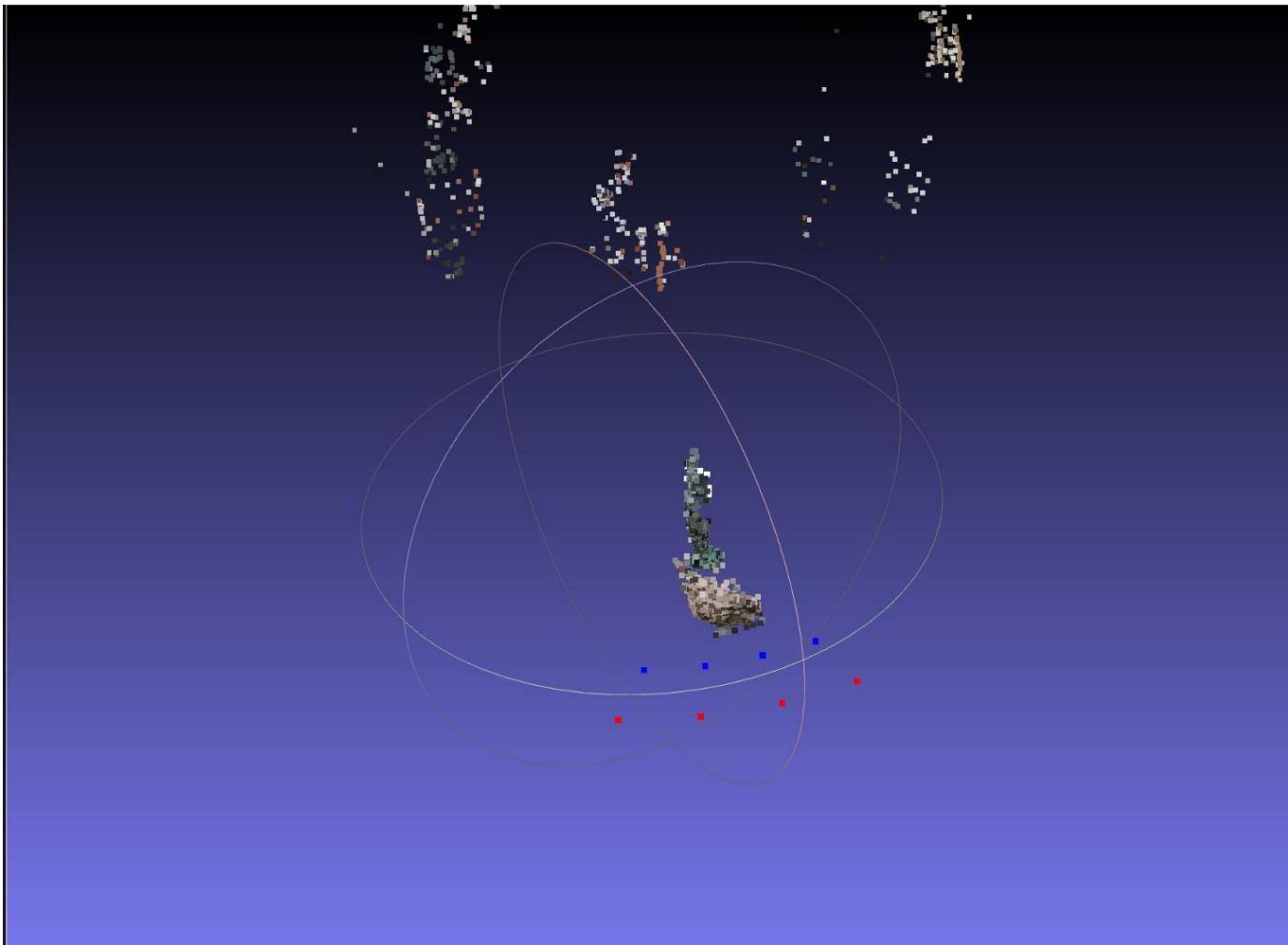


Триангуляция, резекция + уточнение положения камер и точек

BA

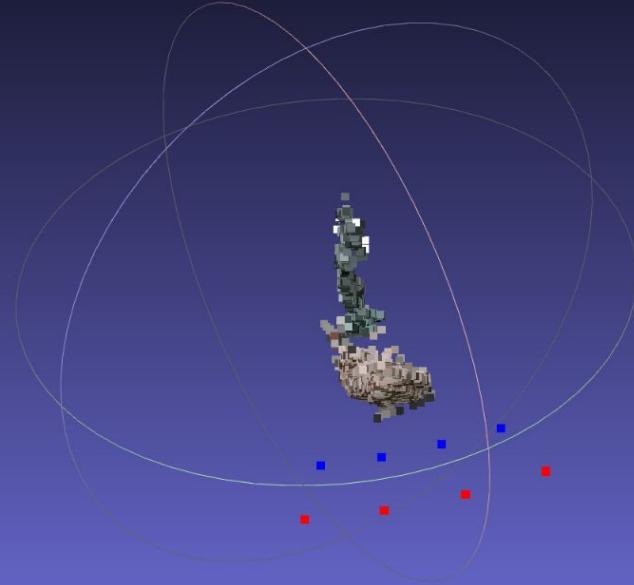


Триангуляция, резекция + уточнение положения камер и точек

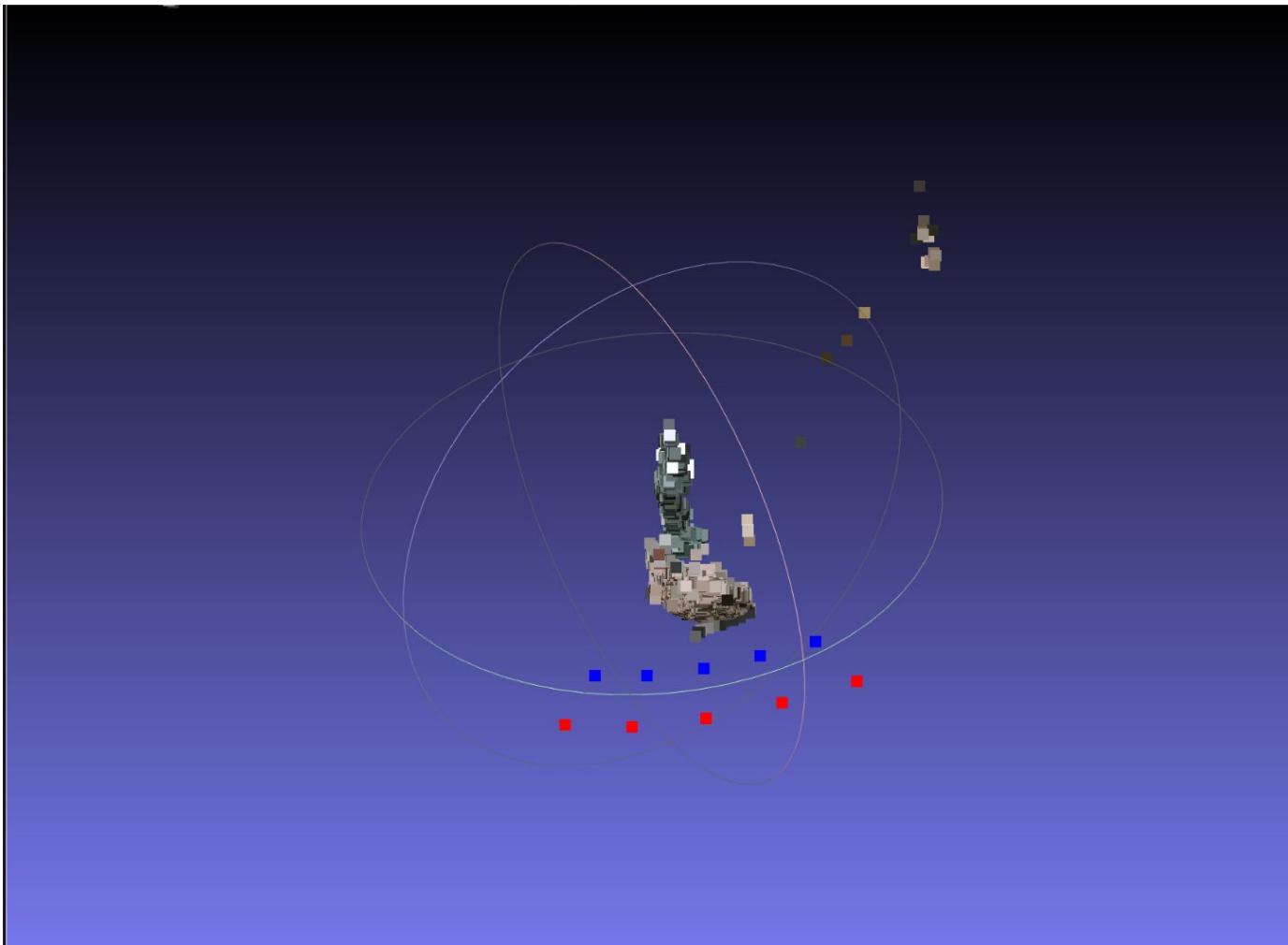


Триангуляция, резекция + уточнение положения камер и точек

BA

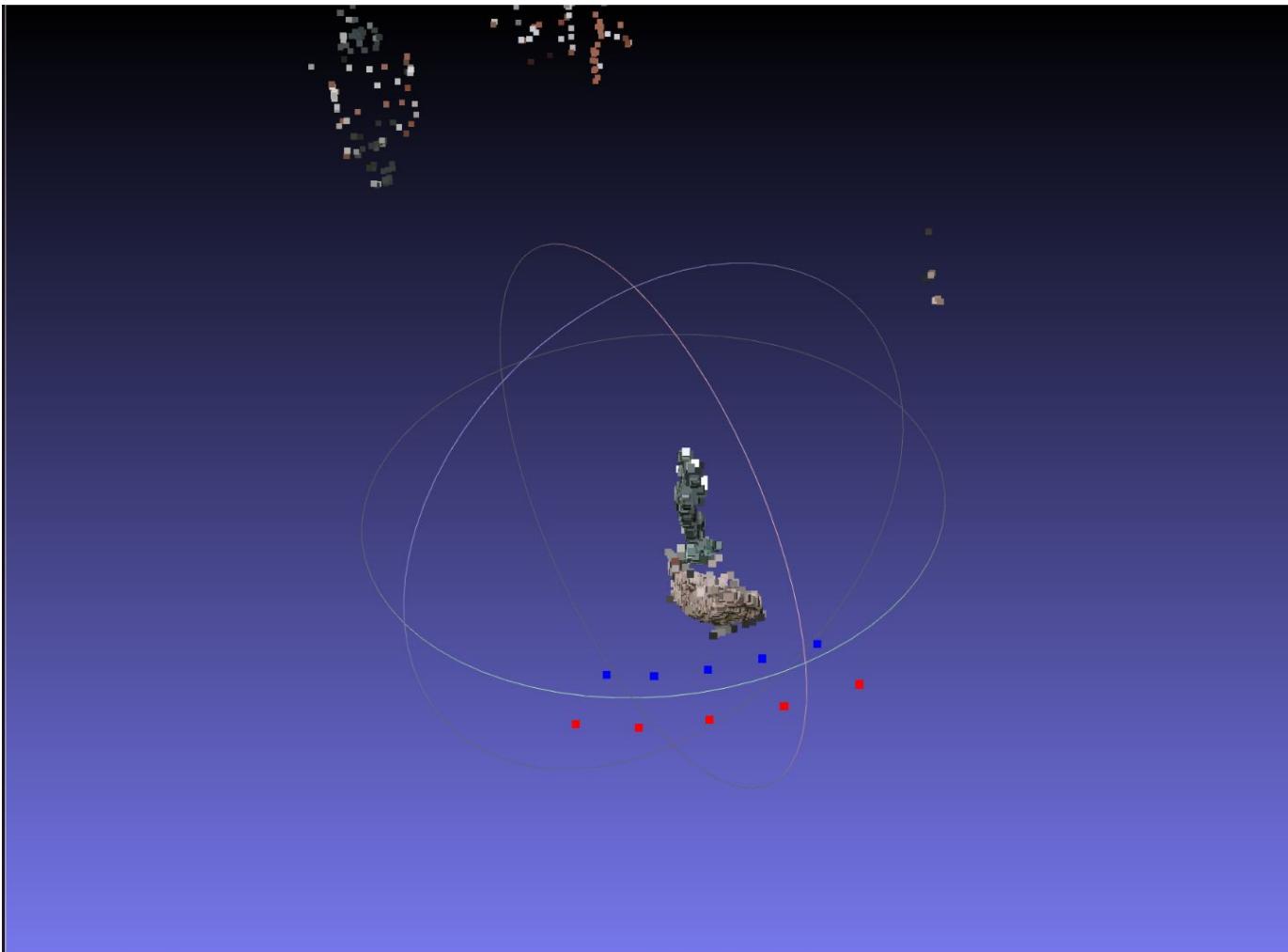


Триангуляция, резекция + уточнение положения камер и точек

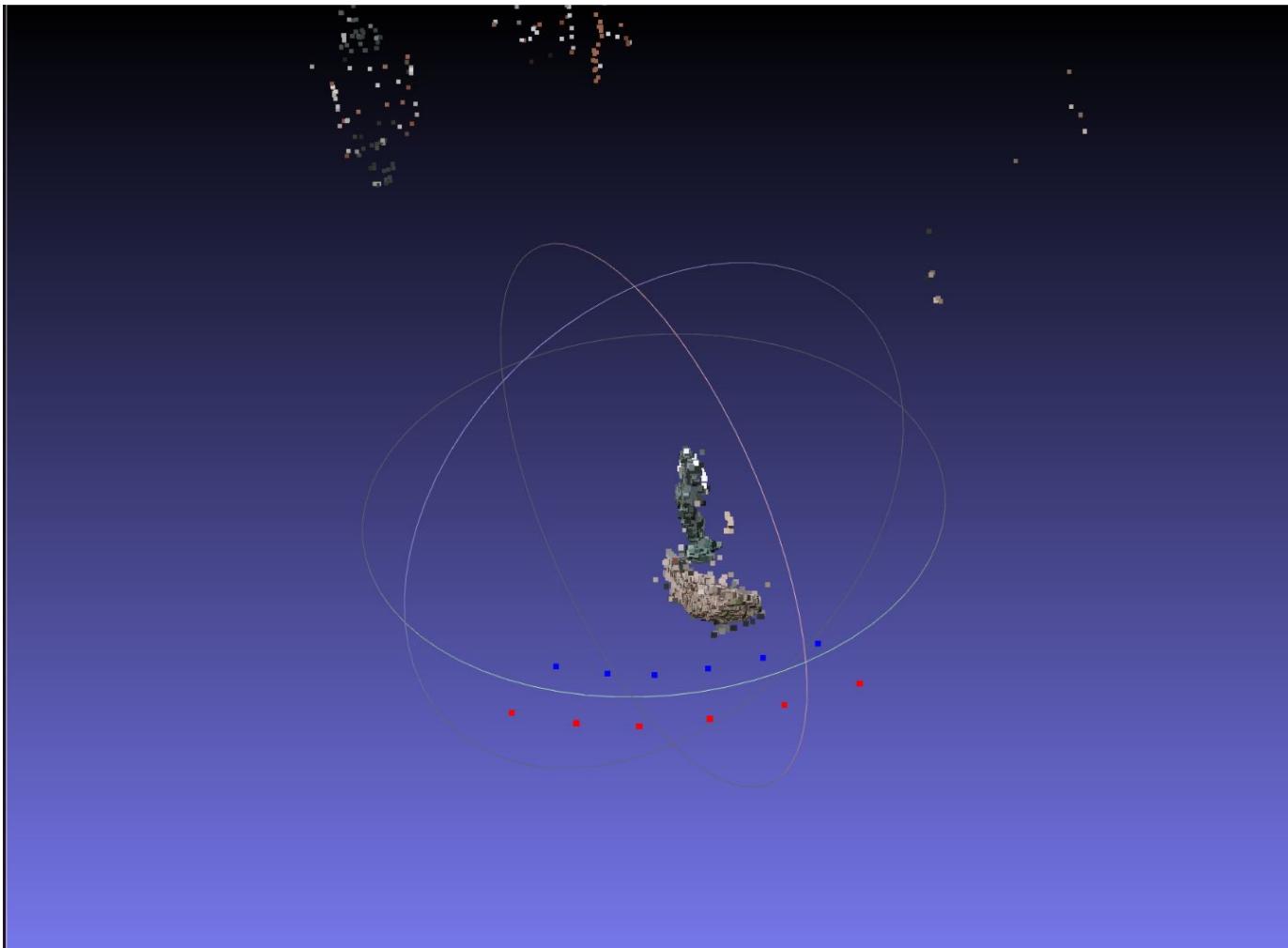


Триангуляция, резекция + уточнение положения камер и точек

BA

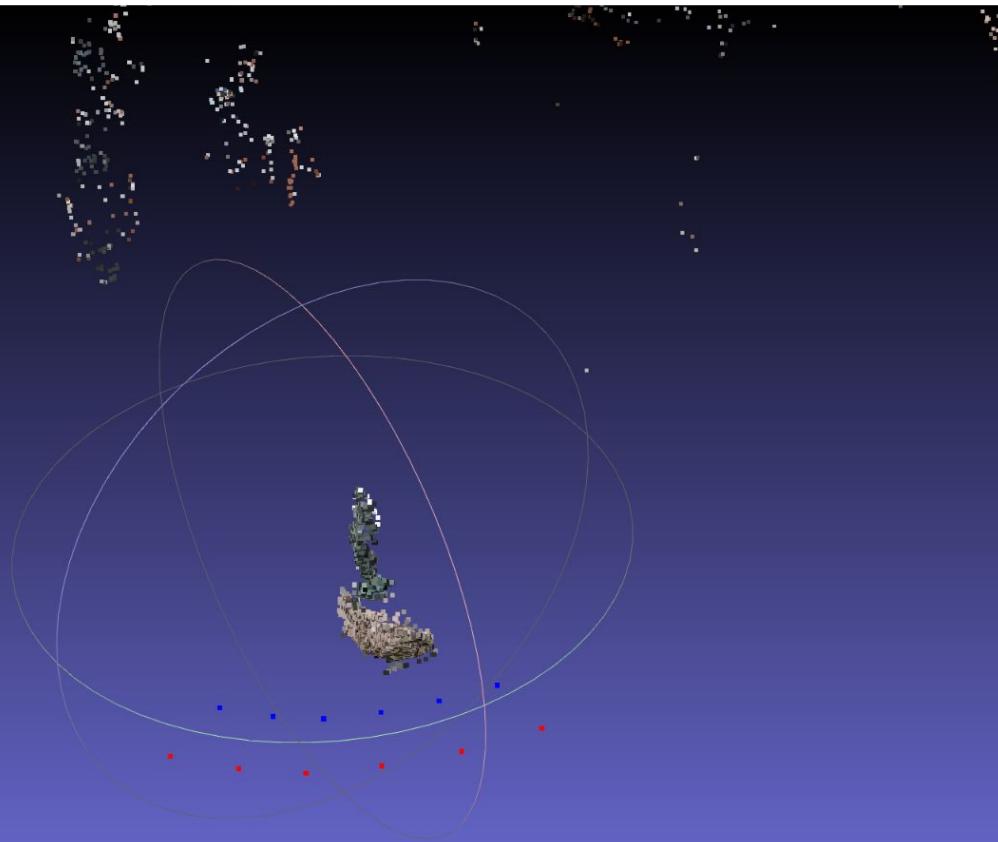


Триангуляция, резекция + уточнение положения камер и точек

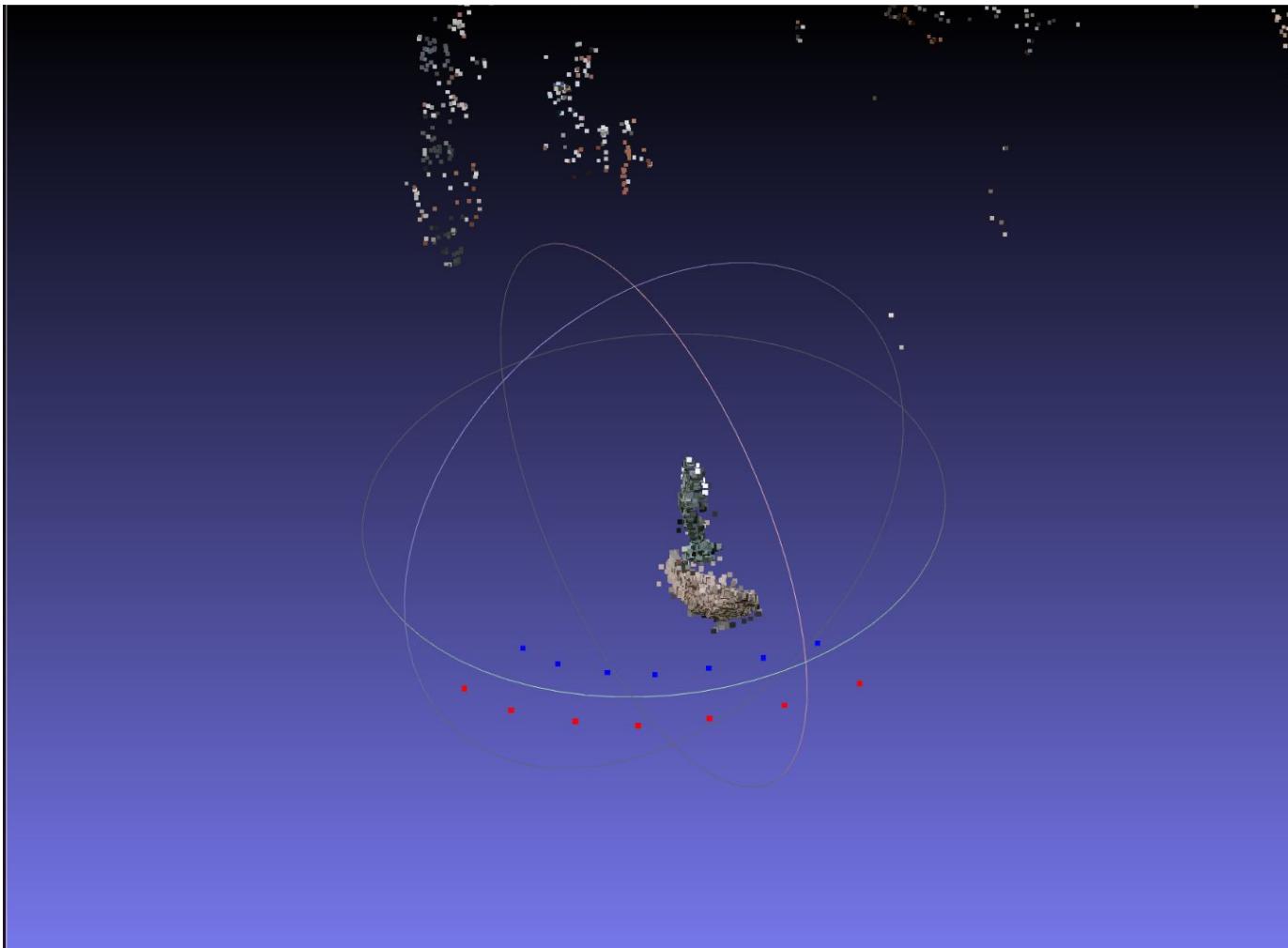


Триангуляция, резекция + уточнение положения камер и точек

BA

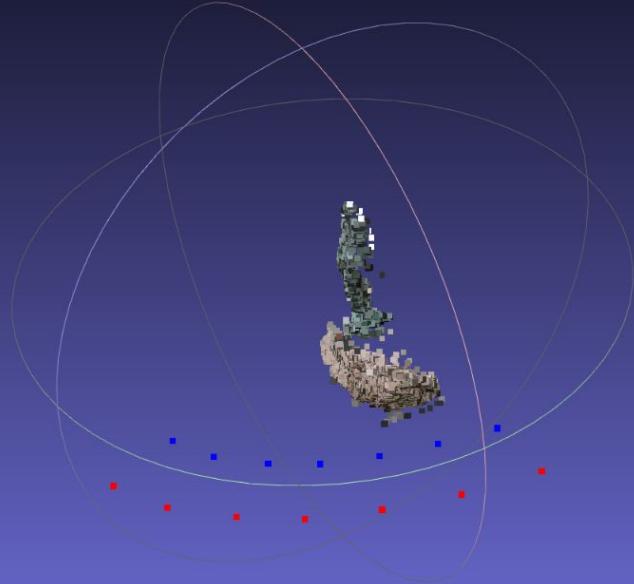


Триангуляция, резекция + уточнение положения камер и точек

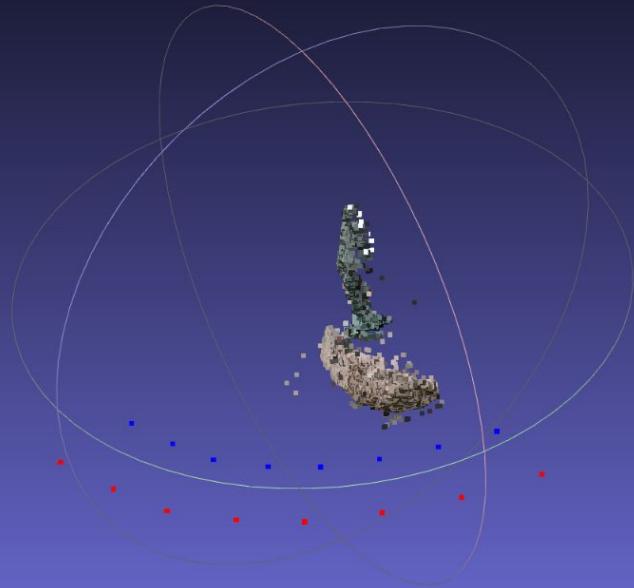


Триангуляция, резекция + уточнение положения камер и точек

BA

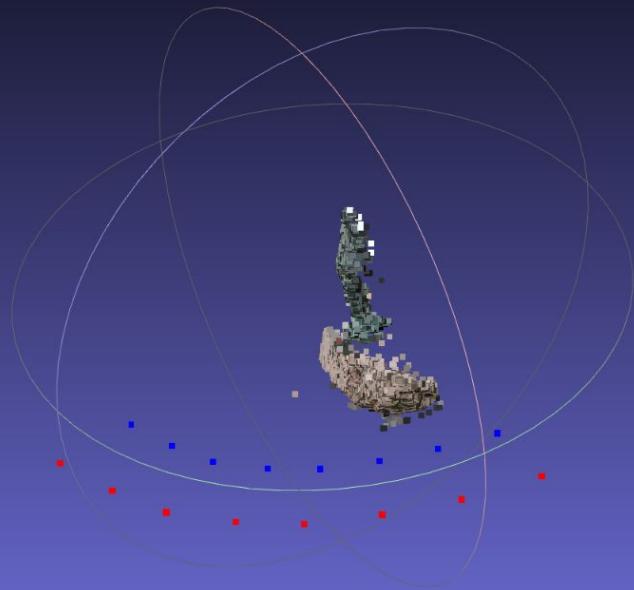


Триангуляция, резекция + уточнение положения камер и точек

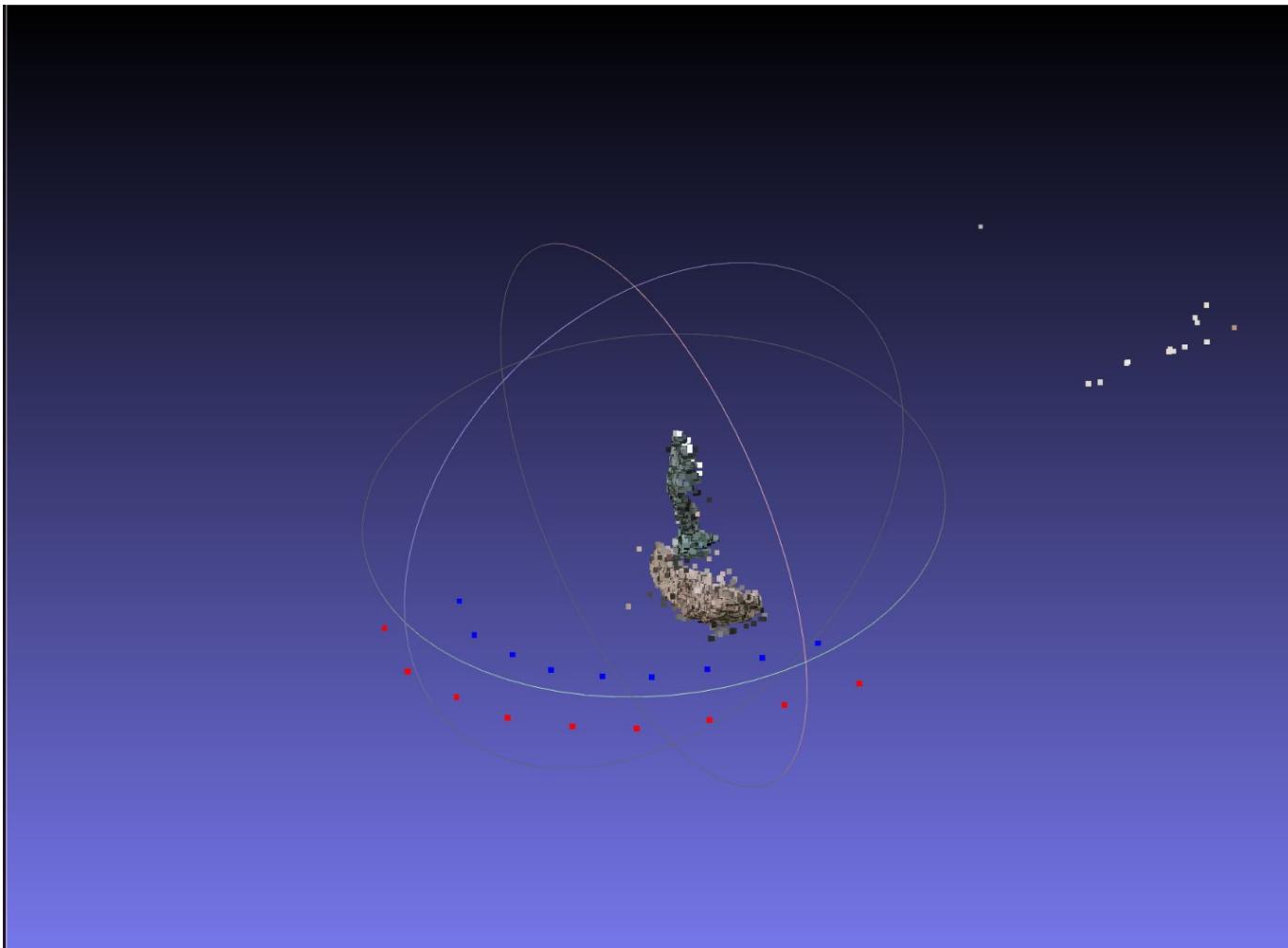


Триангуляция, резекция + уточнение положения камер и точек

BA

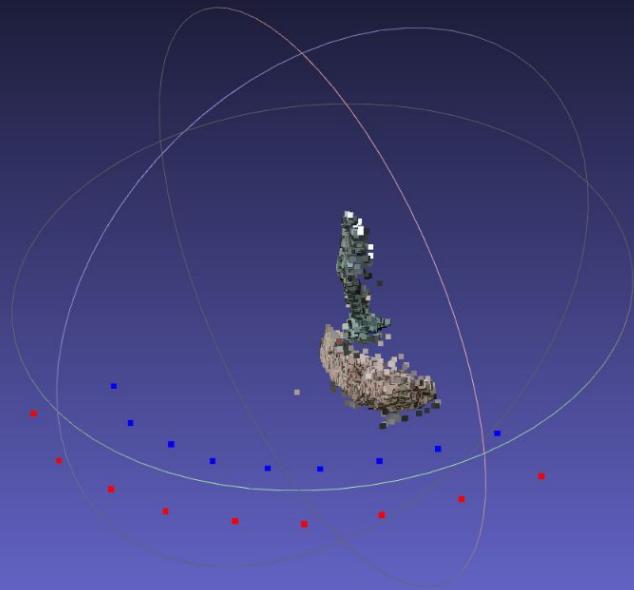


Триангуляция, резекция + уточнение положения камер и точек

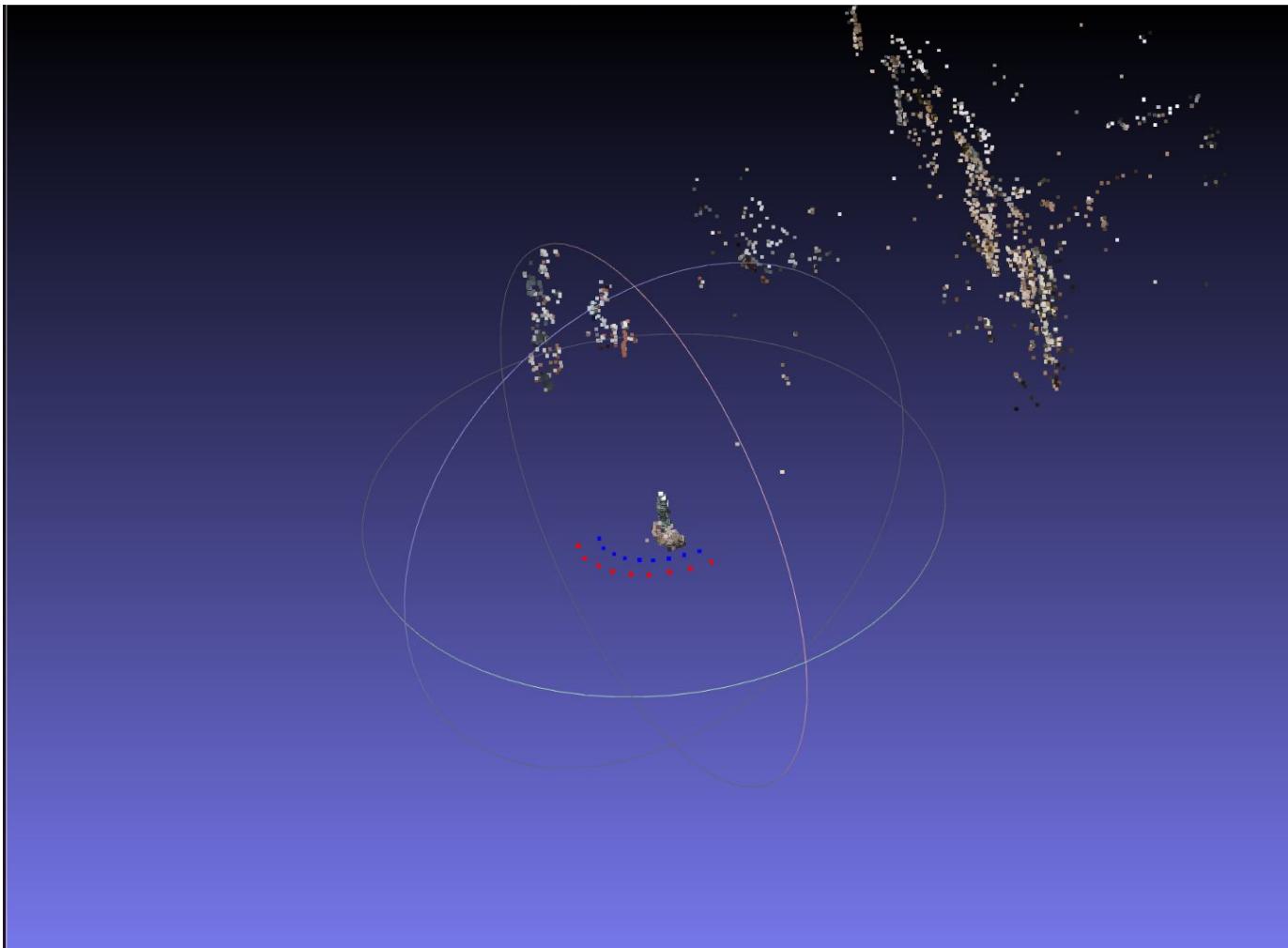


Триангуляция, резекция + уточнение положения камер и точек

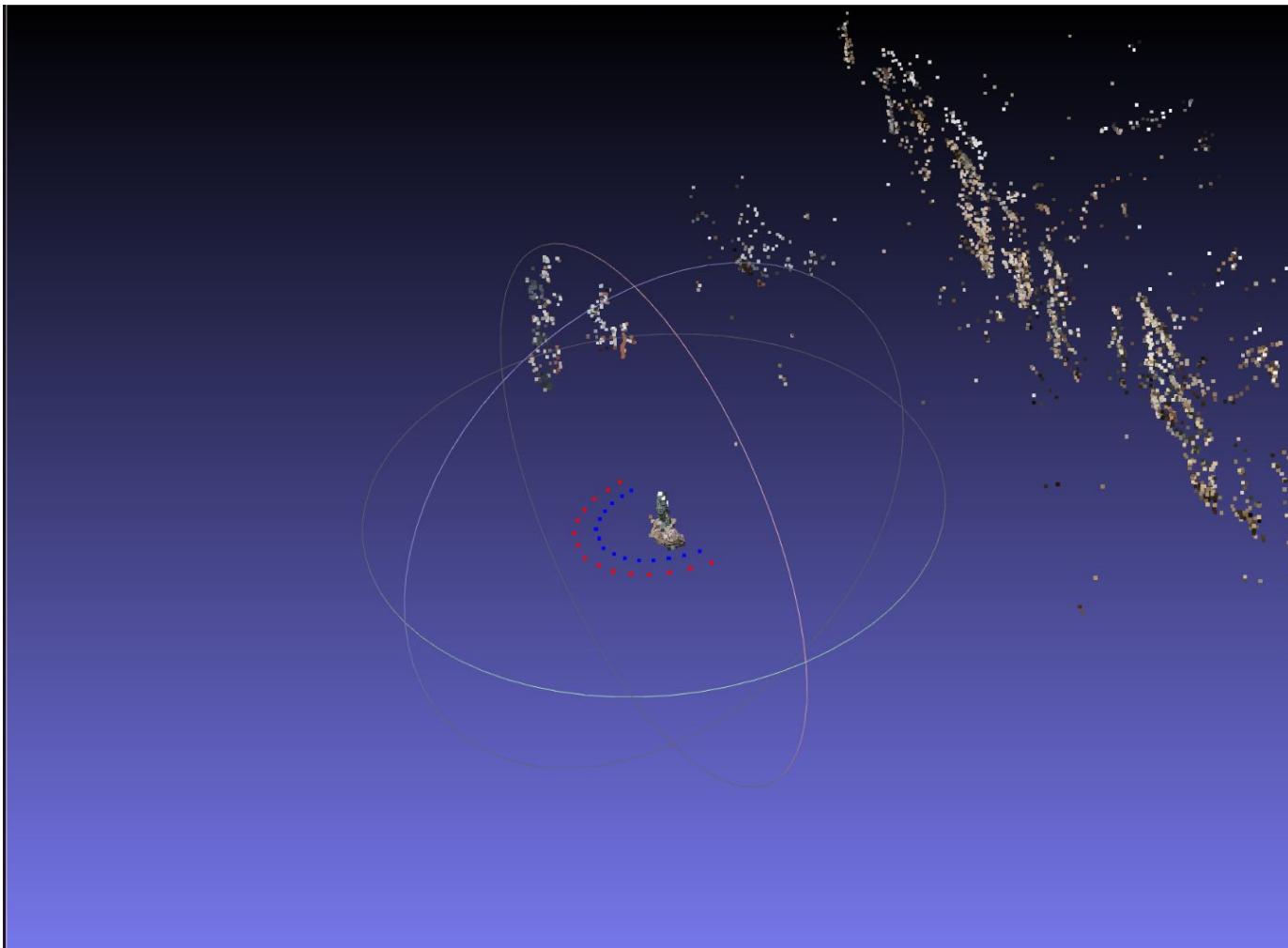
BA



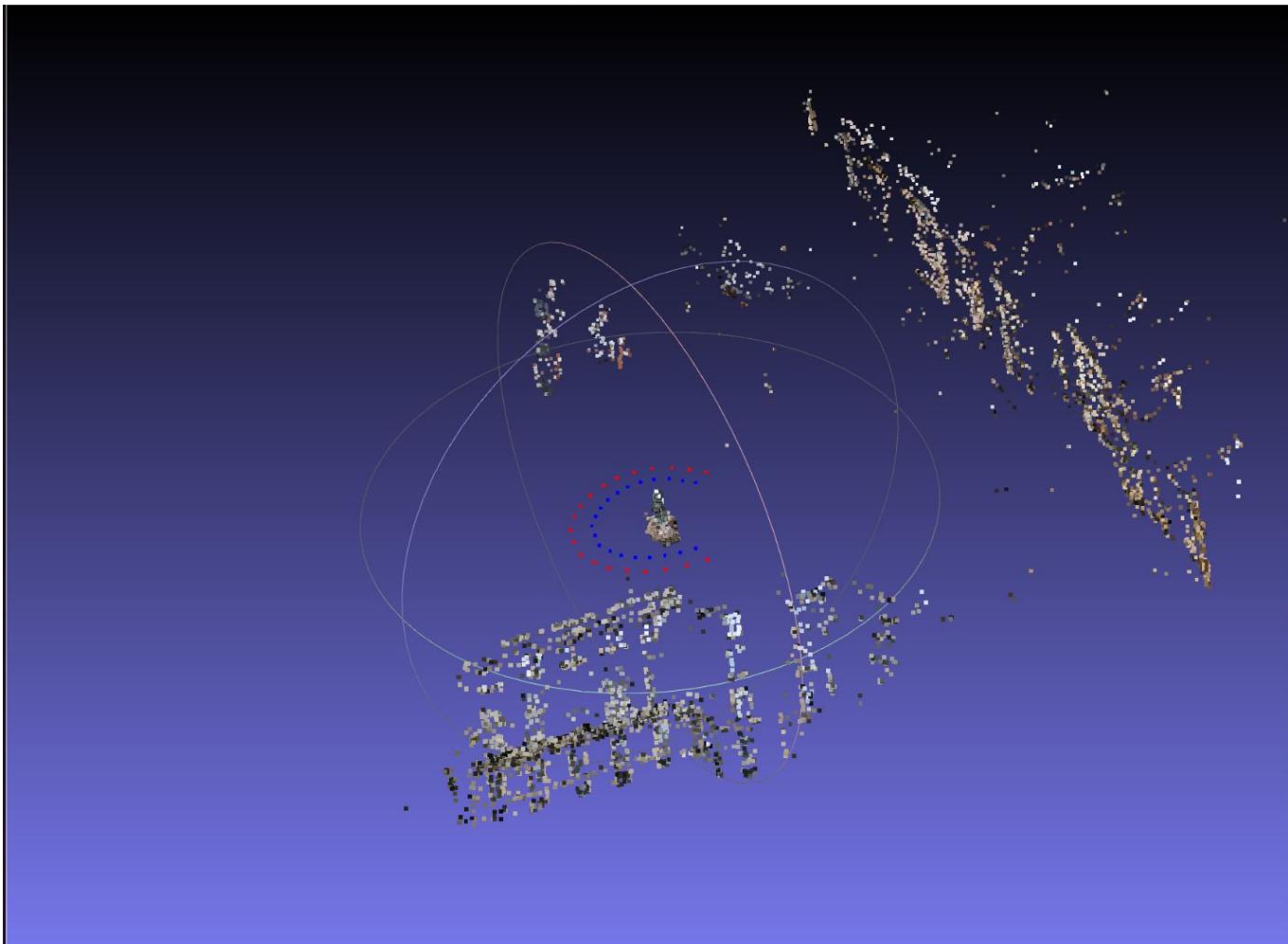
Триангуляция, резекция + уточнение положения камер и точек



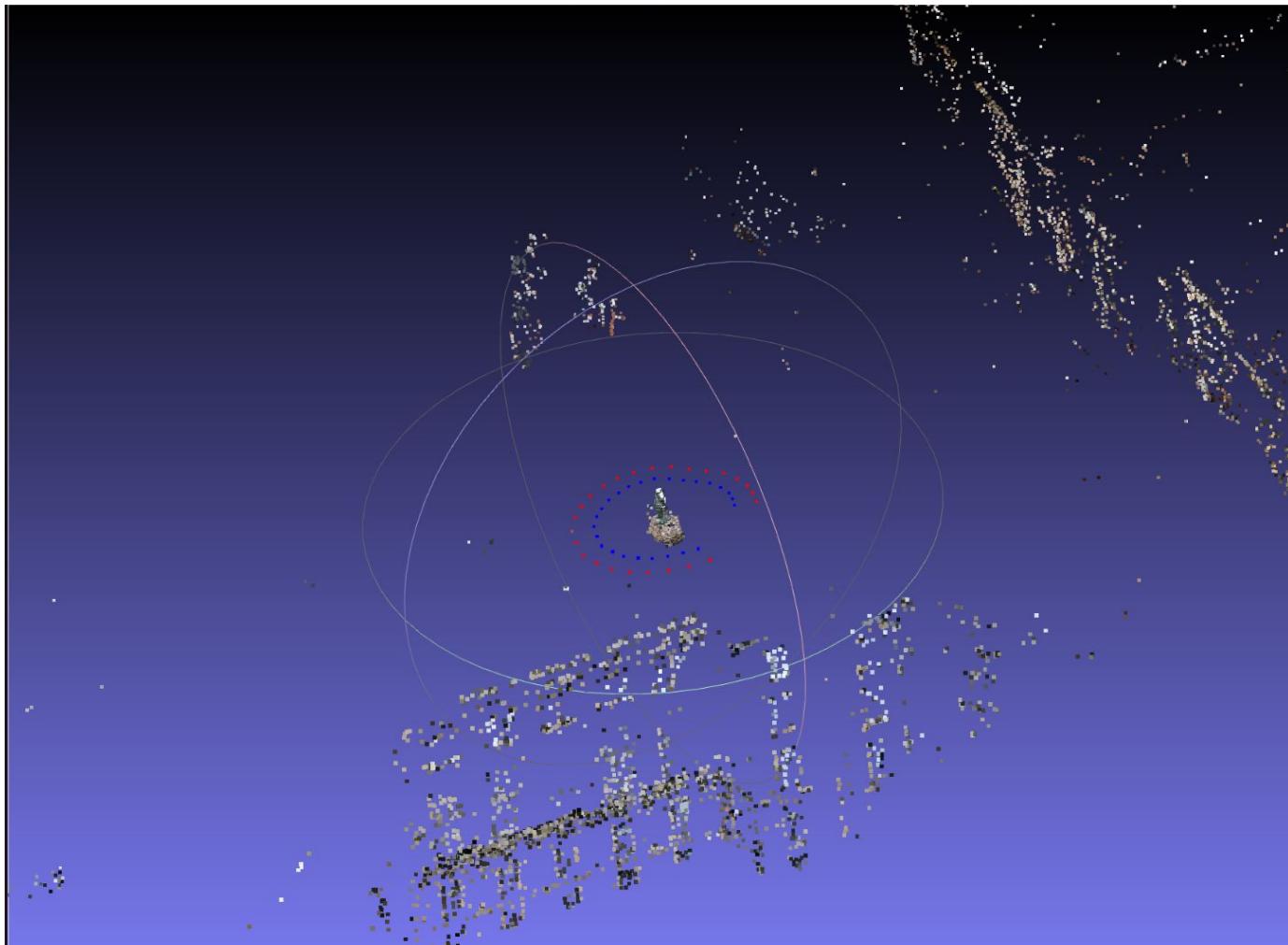
Триангуляция, резекция + уточнение положения камер и точек



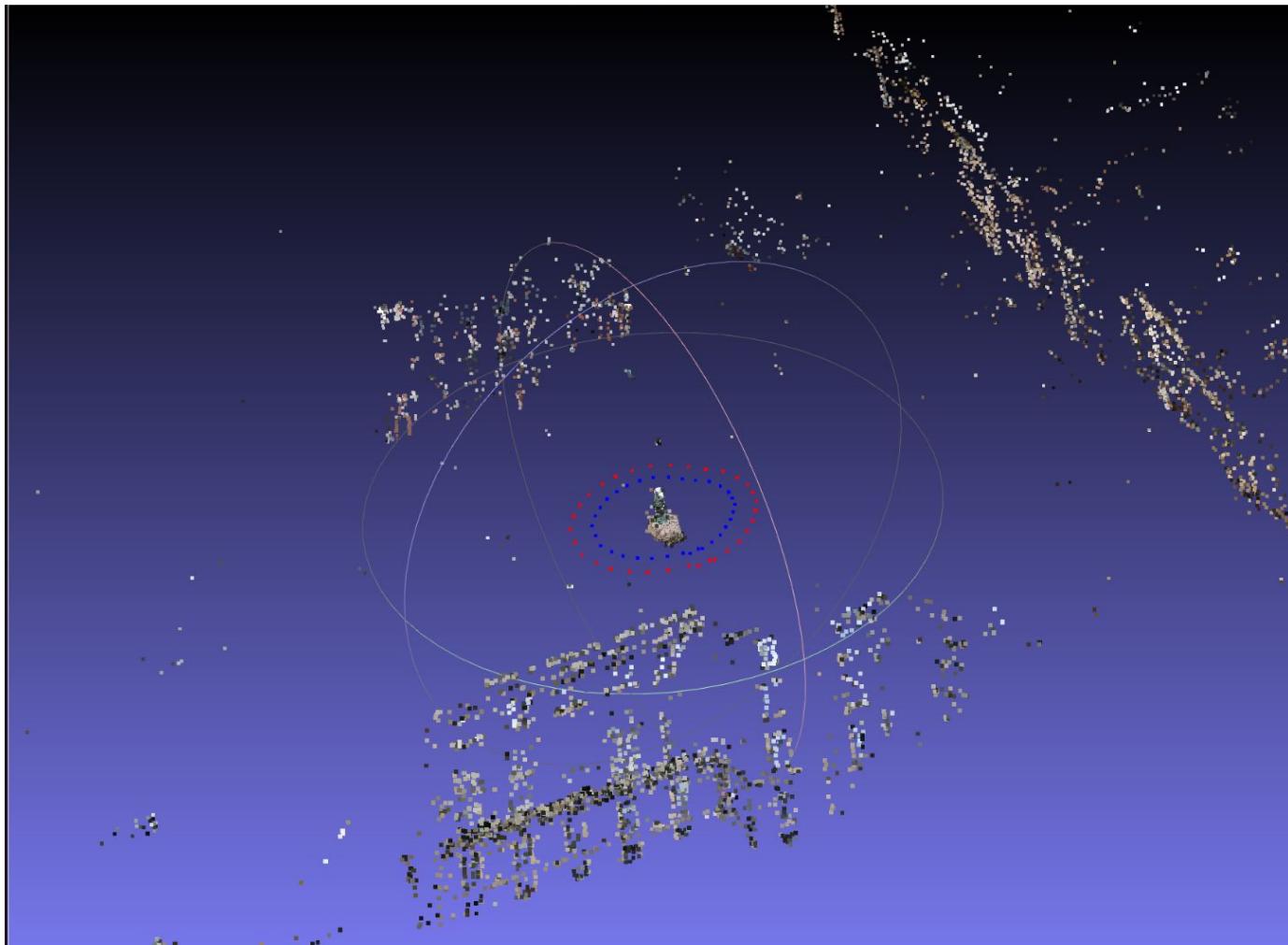
Триангуляция, резекция + уточнение положения камер и точек



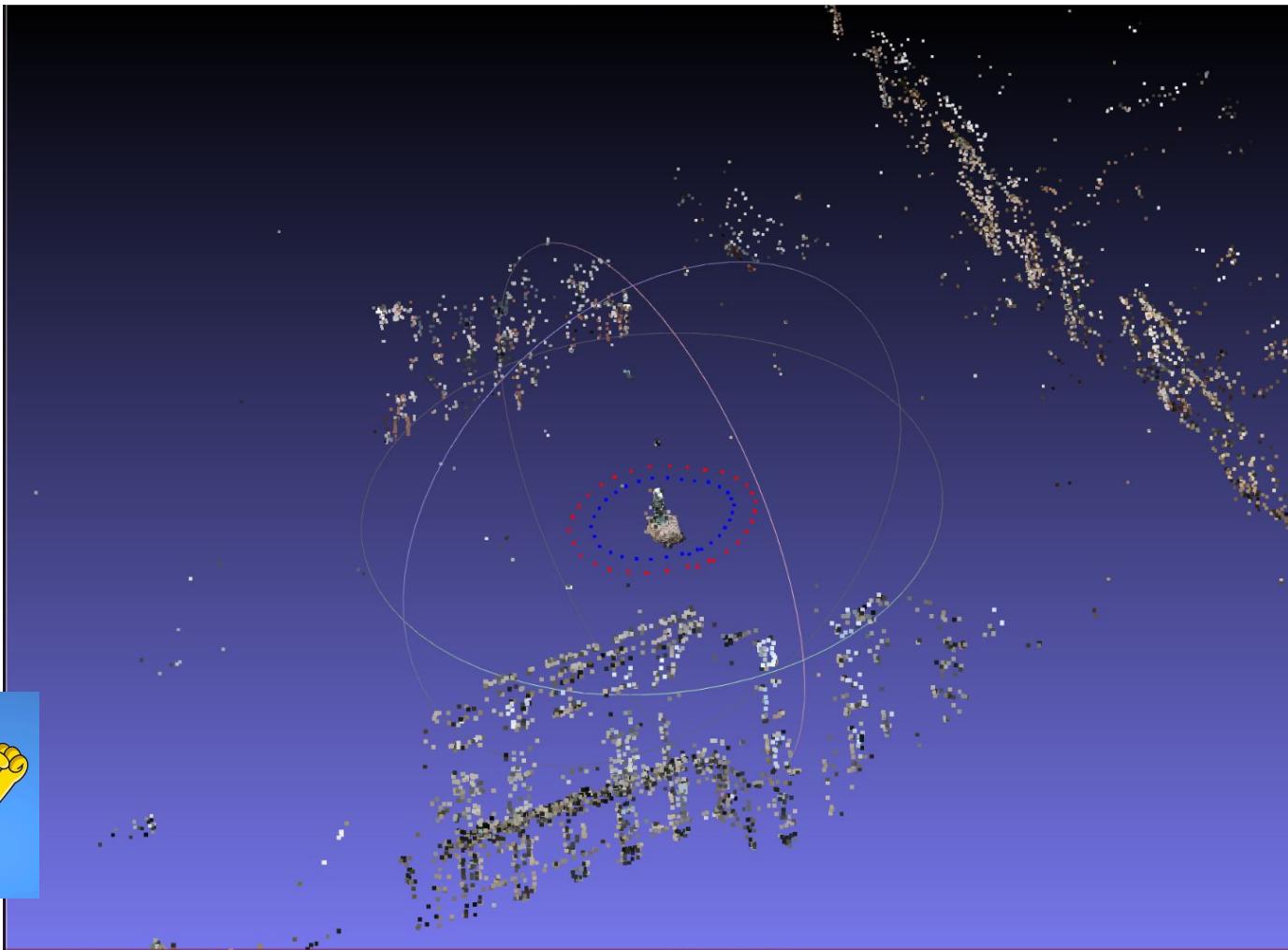
Триангуляция, резекция + уточнение положения камер и точек



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) градиентный спуск
- 2) метод Ньютона
 - a) для поиска корней
 - b) для поиска экстремума
- 3) метод Ньютона
- 4) метод Гаусса-Ньютона
- 5) метод Левенберга-Марквадта

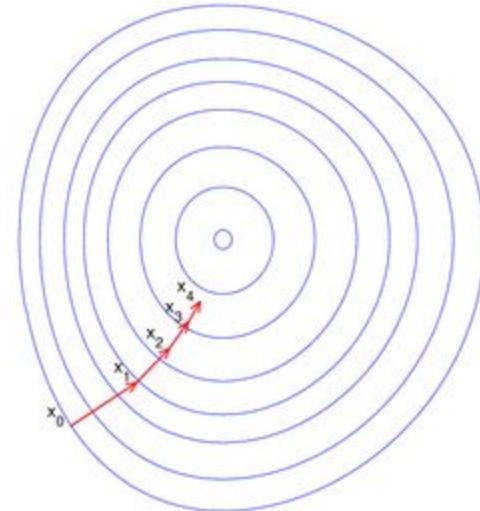
Поиск экстремума 1: градиентный спуск

- 1) Есть функция $f(x)$ хотим найти точку экстремума, например $\arg \min_x f(x)$
- 2) 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 наисклоннейшего
убывания $f(x)$

А как найти максимум? (две формулировки)

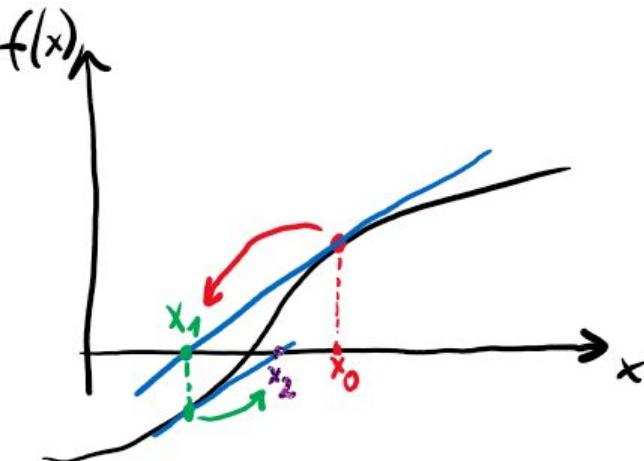


Поиск корня: Метод Ньютона

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



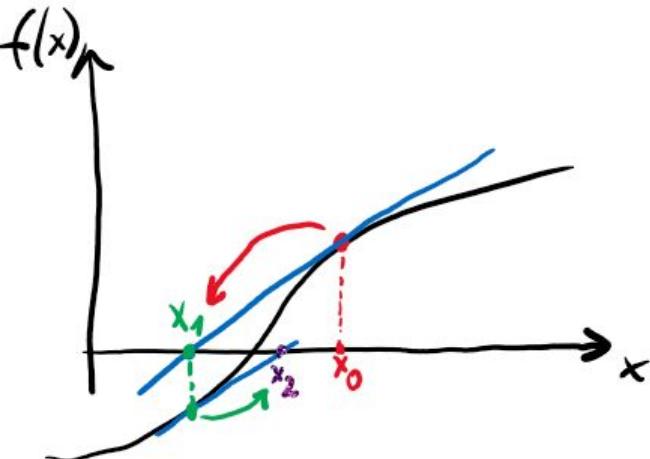
Поиск корня: Метод Ньютона

Сначала решим другую задачу: найдем корень функции

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

Иначе говоря, через ряд Тейлора:

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



Поиск корня: Метод Ньютона

Сначала решим другую задачу: найдем корень функции

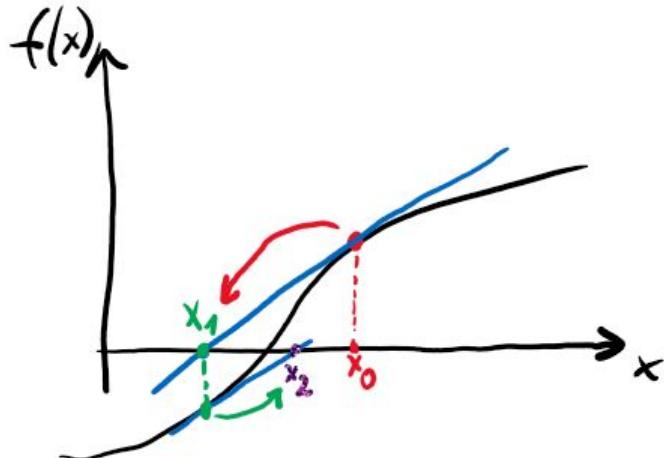
т.е. для $f(x)$ хотим решить уравнение $f(x) = 0$

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) 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:

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

Минимизация квадратичной ошибки

1) Переформулируем нашу задачу:

$$\begin{aligned} \vec{y} - \text{набл.} : \vec{y} \in \mathbb{R}^n \\ \vec{x} - \text{параметры модели} : \vec{x} \in \mathbb{R}^m \\ g(\vec{x}) = \vec{y}, \text{хотим } \vec{y} \approx \vec{y}, g: \mathbb{R}^m \rightarrow \mathbb{R}^n \\ \Rightarrow \frac{1}{2} \|g(\vec{x}) - \vec{y}\| \rightarrow \min \Leftrightarrow \\ \Leftrightarrow f(x) = \frac{1}{2} r^T(x) r(x) \rightarrow \min \\ r - \text{небольш., } r_i(x) = g(\vec{x})_i - \vec{y}_i \\ (\text{residual}) \end{aligned}$$

Минимизация квадратичной ошибки

- 1) Переформулируем нашу задачу:
- 2) Линейные задачи: линейно зависят от параметров
 - a) Существует замкнутое решение в виде линейной системы (SVD, QR decomposition, ...)
- 3) Нелинейные задачи: остальные
 - a) Приходится решать итерационно

\vec{y} -набл : $\vec{y} \in \mathbb{R}^n$
 \vec{x} -параметры модели : $\vec{x} \in \mathbb{R}^m$
 $g(\vec{x}) = \vec{y}$, тогда $\vec{y} \approx \vec{y}$, $g : \mathbb{R}^m \rightarrow \mathbb{R}^n$
 $\Rightarrow \frac{1}{2} \| g(\vec{x}) - \vec{y} \| \rightarrow \min \Leftrightarrow$
 $\Leftrightarrow f(\vec{x}) = \frac{1}{2} r^T(\vec{x}) r(\vec{x}) \rightarrow \min$
 r -небялка, $r_i(\vec{x}) = g(\vec{x})_i - \vec{y}_i$
(residual)

Минимизация квадратичной ошибки

1) Модель плоскости, линейная?

$$g(x, y, a, b, c) = ax + by + c$$

2) А модель параболоида?

$$g(x, y, a, b, c) = ax^2 + by^2 + c$$

3) А так?

$$g(x, x^2, y, y^2, a, b, c) = ax^2 + by^2 + cx$$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

1) Если минимизируем

квадратичную ошибку, то можем

ускориться по сравнению с $\nabla f(x_n) = \nabla(r^T r) = \nabla(r_1^2 + r_2^2 + \dots + r_n^2) =$

методом Ньютона (ускорить
подсчет гессиана)

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

$$\begin{aligned} &= \begin{pmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{pmatrix} (r_1^2 + \dots + r_n^2) = 2 \begin{pmatrix} \frac{\partial r_1}{\partial x_1} r_1 + \frac{\partial r_2}{\partial x_1} r_2 + \dots + \frac{\partial r_n}{\partial x_1} r_n \\ \vdots \\ \frac{\partial r_1}{\partial x_n} r_1 + \dots + \frac{\partial r_n}{\partial x_n} r_n \end{pmatrix} \end{aligned}$$

$$= 2 \begin{pmatrix} \frac{\partial r_1}{\partial x_1} & \frac{\partial r_2}{\partial x_1} & \dots & \frac{\partial r_n}{\partial x_1} \\ \vdots & & & \vdots \\ \frac{\partial r_1}{\partial x_n} & \dots & & \frac{\partial r_n}{\partial x_n} \end{pmatrix} \vec{r} = 2 \underline{[\vec{J} r] \cdot r}$$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

- 1) Если минимизируем квадратичную ошибку, то можем ускориться по сравнению с методом Ньютона (ускорить подсчет гессиана)

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

$$\begin{aligned} H f(x) &= \frac{\partial^2 f(x)}{\partial x^2} = \frac{\partial}{\partial x} \frac{\partial f(x)}{\partial x} = \underbrace{y}_{2r^T y} r = \\ &= 2(y^T y_r + \sum_{i=0}^n r_i H r_i) \approx \\ &\quad \text{мало} \quad \text{const} \end{aligned}$$

$$\approx \underline{2y^T y_r}$$

Поиск экстремума 3: алгоритм Гаусса-Ньютона

- 1) Если минимизируем квадратичную ошибку, то можем ускориться по сравнению с методом Ньютона (ускорить подсчет гессиана)

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

$$\downarrow$$
$$x_{n+1} = x_n - [\cancel{2} \tilde{y}_n^T \tilde{y}_n]^{-1} \cancel{[b \tilde{y}_n]} r$$

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)
- **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)
- **нас устраивает:** 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) Как взять лучшее от градиентного спуска и Гаусса-Ньютона?

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)
- **нас устраивает:** подходит только для минимизации суммы квадратов

Поиск экстремума 4: алгоритм Левенберга-Марквадта

градиентный спуск:

$$x_{n+1} = x_n - \lambda \nabla f(x_n) = x_n - \lambda [y^T] r$$

Гаусс-Ньютон:

$$x_{n+1} = x_n - [y^T_n y_n]^{-1} [y^T_n] r$$

Поиск экстремума 4: алгоритм Левенберга-Марквадта

градиентный спуск:

$$x_{n+1} = x_n - \lambda \nabla f(x_n) = x_n - \lambda [y^T] r$$

Гаусс-Ньютон:

$$x_{n+1} = x_n - [y^T_n y_n]^{-1} [y^T_n] r$$

совместим:

Левенберг-Марквадт:

$$x_{n+1} = x_n - [y^T_n y_n + \lambda I]^{-1} [y^T_n] r$$

==

Поиск экстремума 4: алгоритм Левенберга-Марквадта

градиентный спуск:

$$x_{n+1} = x_n - \lambda \nabla f(x_n) = x_n - \lambda [y^T] r$$

Гаусс-Ньютон:

$$x_{n+1} = x_n - [y_n^T y_n]^{-1} [y_n^T r] r$$

совместим:

Левенберг-Марквадт:

$$x_{n+1} = x_n - [y_n^T y_n + \lambda I]^{-1} [y_n^T r] r$$

λ - ползунок между ГС и ГН

Поиск экстремума 4: алгоритм Левенберга-Марквадта

- При увеличении λ шаг становится в направлении антиградиента

$$\begin{aligned} [\mathbf{J}^T \mathbf{J} + \lambda \mathbf{I}]^{-1} &= \frac{1}{\lambda} \left[\mathbf{I} + \frac{\mathbf{J}^T \mathbf{J}}{\lambda} \right]^{-1} = \\ &= \frac{1}{\lambda} \left[\mathbf{I} - \frac{\mathbf{J}^T \mathbf{J}}{\lambda} + \left(\frac{\mathbf{J}^T \mathbf{J}}{\lambda} \right)^2 + \dots \right] \\ &\quad \text{след. порядок по } \lambda \end{aligned}$$

Поиск экстремума 4: алгоритм Левенберга-Марквадта

- 1) При увеличении λ шаг становится в направлении антиградиента
- 2) Для ускорения сходимости в узком каньоне можно заменить

$$I \rightarrow \text{diag}(\gamma^\top \gamma)$$

$$\begin{aligned} [\gamma^\top \gamma + \lambda I]^{-1} &= \frac{1}{\lambda} [I + \frac{\gamma^\top \gamma}{\lambda}]^{-1} = \\ &= \frac{1}{\lambda} \left[I - \frac{\gamma^\top \gamma}{\lambda} + \left(\frac{\gamma^\top \gamma}{\lambda} \right)^2 + \dots \right] \end{aligned}$$

↓
0

след. порядок
по λ

Поиск экстремума 4: алгоритм Левенберга-Марквадта

- 1) При увеличении λ шаг становится в направлении антиградиента
- 2) Для ускорения сходимости в узком каньоне можно заменить

$$I \rightarrow \text{diag}(\gamma^\top \gamma)$$

- 3) Как выбирать λ ?
 - a) line search: увеличиваем λ в 10 раз пока не улучшим целевую функцию
 - b) вместе с направлением λ еще и замедляет шаг, точно улучшим

$$\begin{aligned} [\gamma^\top \gamma + \lambda I]^{-1} &= \frac{1}{\lambda} [I + \frac{\gamma^\top \gamma}{\lambda}]^{-1} = \\ &= \frac{1}{\lambda} \left[I - \frac{\gamma^\top \gamma}{\lambda} + \left(\frac{\gamma^\top \gamma}{\lambda} \right)^2 + \dots \right] \\ &\quad \downarrow 0 \\ &\quad \text{след. переходы по } \lambda \end{aligned}$$

Поиск экстремума 4: алгоритм Левенберга-Марквардта

Как менять параметр Марквардта λ ?

Уменьшать если значение функции уменьшается (хотим лучше угадывать шаги методом Гаусса-Ньютона чтобы быстрее сойтись).

В результате когда мы будем рядом с минимумом - мы прыгнем в него Гаусс-Ньютоном за мало шагов, т.к. окрестность у экстремума обычно похожа на поверхность второго порядка.

Увеличивать если что-то пошло не так и функция увеличилась (паникуем и переходим на простой и надежный но медленный - градиентный спуск).

Bundle Adjustment

- 1) Хотим научиться улучшать положения камер, их калибровок и связующих точек

Bundle Adjustment

- 1) Хотим научиться улучшать положения камер, их калибровок и связующих точек
- 2) Функция проекции:

$$f_T(x) = \underline{x} - \bar{T}$$
$$f_R(x) = \underline{R} \cdot \underline{x}$$
$$f_p\left(\frac{\underline{x}}{\underline{z}}\right) = \begin{pmatrix} \underline{f} \cdot \frac{\underline{x}}{\underline{z}} \\ \underline{f} \cdot \frac{\underline{y}}{\underline{z}} \end{pmatrix}$$
$$f_{RD}\left(\frac{\underline{x}}{\underline{z}}\right) = \left(1 + \frac{k_1}{r^2} + \frac{k_2}{r^4}\right) \begin{pmatrix} \underline{x} \\ \underline{y} \end{pmatrix} \text{ где } r^2 = \underline{x}^2 + \underline{y}^2$$
$$f_c\left(\frac{\underline{x}}{\underline{y}}\right) = \begin{pmatrix} \underline{x} + c_x \\ \underline{y} + c_y \end{pmatrix}$$

параметры:

- \bar{T} - координаты камеры (3 степени свободы)
- \underline{R} - 3x3 матрица квадрата (матрица камера)
- f - фокальная длина (в пикселях)
- k_1, k_2 - коэффициенты радиальной дисторсии
- c_x, c_y - смещение оптической оси

Bundle Adjustment

- 1) Хотим научиться улучшать положения камер, их калибровок и связующих точек
- 2) Функция проекции:
- 3) У нас есть наблюдения: задетектированные и сматченные ключевые точки

$$f_T(x) = x - \bar{T}$$
$$f_R(x) = R \cdot x$$
$$f_p\left(\frac{x}{z}\right) = \begin{pmatrix} f \cdot \frac{x}{z} \\ f \cdot \frac{y}{z} \end{pmatrix}$$
$$f_{RD}(x) = (1 + k_1 r^2 + k_2 r^4) \begin{pmatrix} x \\ y \end{pmatrix} \text{ где } r^2 = x^2 + y^2$$
$$f_c\left(\begin{matrix} x \\ y \end{matrix}\right) = \begin{pmatrix} x + c_x \\ y + c_y \end{pmatrix}$$

параметры:

- \bar{T} - координаты камеры (3 степени свободы)
- R - 3x3 матрица квадрата (мировая камера)
- f - фокальная длина (в пикселях)
- k_1, k_2 - коэффициенты радиальной дисторсии
- c_x, c_y - смещение оптической оси

Bundle Adjustment

- 1) Хотим научиться улучшать положения камер, их калибровок и связующих точек
- 2) Функция проекции:
- 3) У нас есть наблюдения: задетектированные и сматченные ключевые точки
- 4) Составим оптимизационную задачу

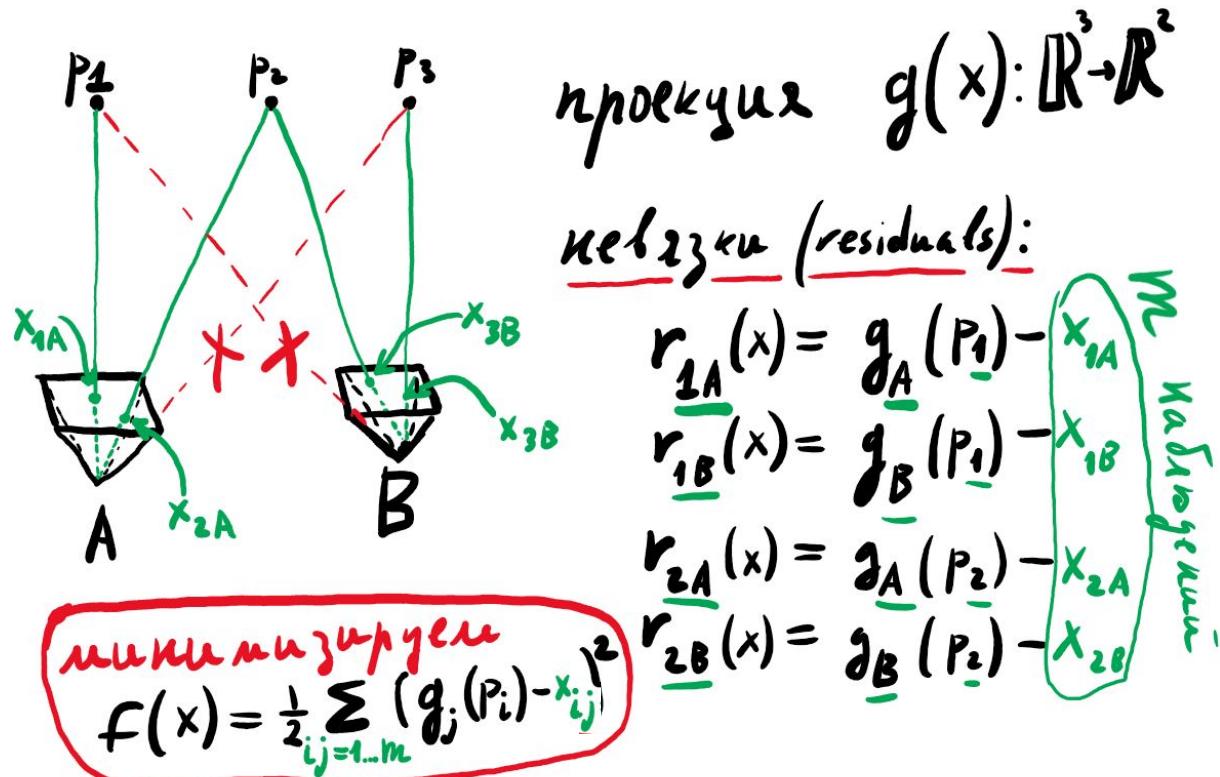
$$f_T(x) = x - \bar{T}$$
$$f_R(x) = R \cdot x$$
$$f_p\left(\frac{x}{z}\right) = \begin{pmatrix} f \cdot \frac{x}{z} \\ f \cdot \frac{y}{z} \end{pmatrix}$$
$$f_{RD}(x) = (1 + k_1 r^2 + k_2 r^4) \begin{pmatrix} x \\ y \end{pmatrix} \text{ где } r^2 = x^2 + y^2$$
$$f_c\left(\begin{matrix} x \\ y \end{matrix}\right) = \begin{pmatrix} x + c_x \\ y + c_y \end{pmatrix}$$

параметры:

- \bar{T} - координаты камеры (3 степени свободы)
- R - 3x3 матрица калюбра (матрица камеры)
- f - фокальная длина (в пикселях)
- k_1, k_2 - коэффициенты радиальной дисторсии
- c_x, c_y - смещение оптической оси

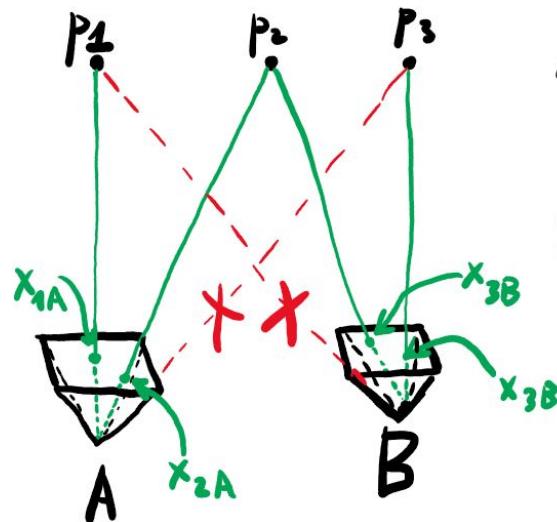
Bundle Adjustment

- 1) Составим оптимизационную задачу



Bundle Adjustment

- 1) Составим оптимизационную задачу
- 2) Решим с помощью алгоритма Левенберга-Марквадта



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

небольшие (residuals):

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

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

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

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

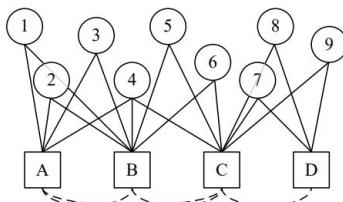
минимизируем

$$f(x) = \frac{1}{2} \sum_{ij=1 \dots m} (g_j(P_i) - x_{ij})^2$$

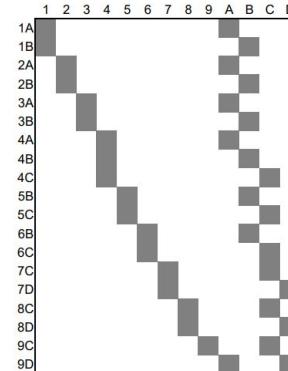
наблюдения

Bundle Adjustment. Schur complement

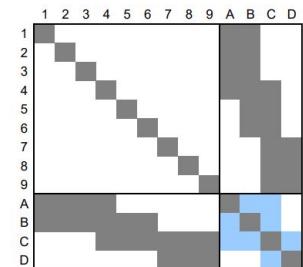
1) Посмотрим на структуру характерных матриц:



(a)



(b)

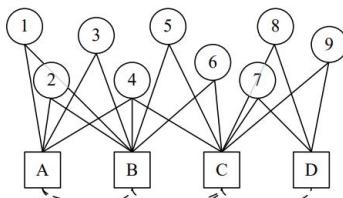


(c)

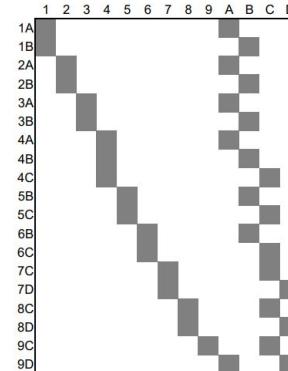
Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point)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

Bundle Adjustment. Schur complement

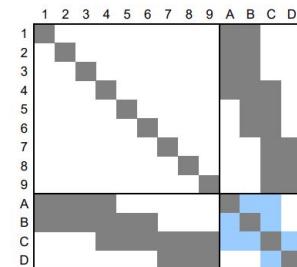
- 1) Посмотрим на структуру характерных матриц:
- 2) Размер матриц $O(N_{cams} + N_{points})$
- 3) N_{points} может быть очень большим!
А N_{cams} обычно не очень



(a)



(b)

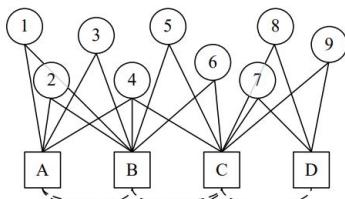


(c)

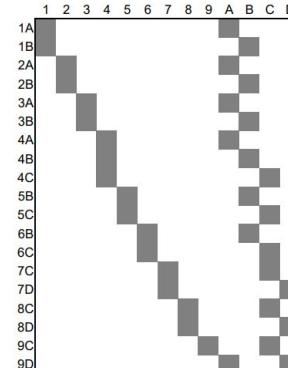
Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point)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

Bundle Adjustment. Schur complement

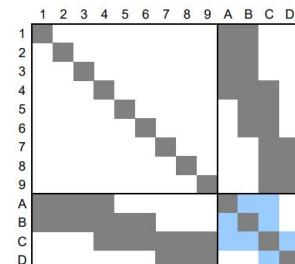
- 1) Посмотрим на структуру характерных матриц:
- 2) Размер матриц $O(N_{\text{cams}} + N_{\text{points}})$
- 3) N_{points} может быть очень большим!
А N_{cams} обычно не очень
- 4) Разреженные матрицы,
QR, column reorderings...



(a)



(b)

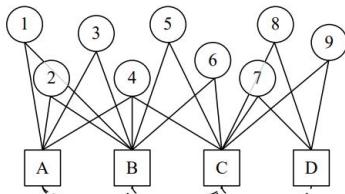


(c)

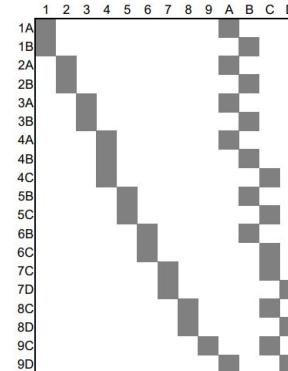
Figure 11.16 (a) Bipartite graph for a toy structure from motion problem and (b) its associated Jacobian \mathbf{J} and (c) Hessian \mathbf{A} . Numbers indicate 3D points and letters indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point)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

Bundle Adjustment. Schur complement

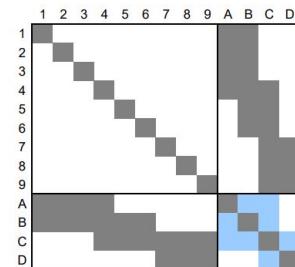
- 1) Посмотрим на структуру характерных матриц:
- 2) Размер матриц $O(N_{cams} + N_{points})$
- 3) N_{points} может быть очень большим!
А N_{cams} обычно не очень
- 4) Разреженные матрицы,
QR, column reorderings...
- 5) Все равно очень тяжело
решать такие большие
системы



(a)



(b)

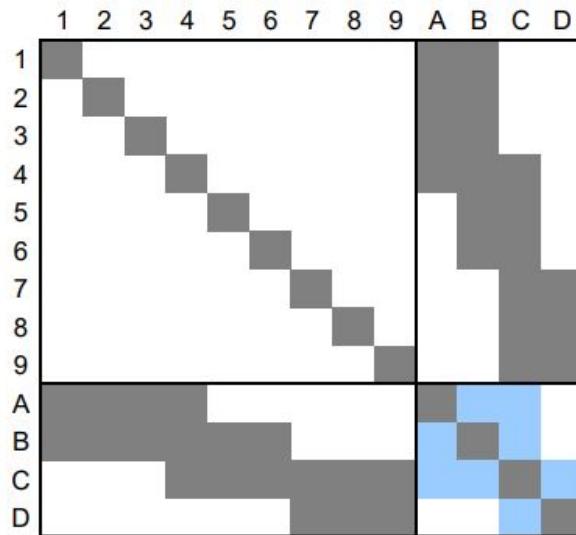


(c)

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.

Bundle Adjustment. Schur complement

- 1) Посмотрим на структуру характерных матриц:
 - 2) Размер матриц $O(N_{cams} + N_{points})$
 - 3) N_{points} может быть очень большим!
А N_{cams} обычно не очень
 - 4) Разреженные матрицы,
QR, column reorderings...
 - 5) Все равно очень тяжело
решать такие большие
системы

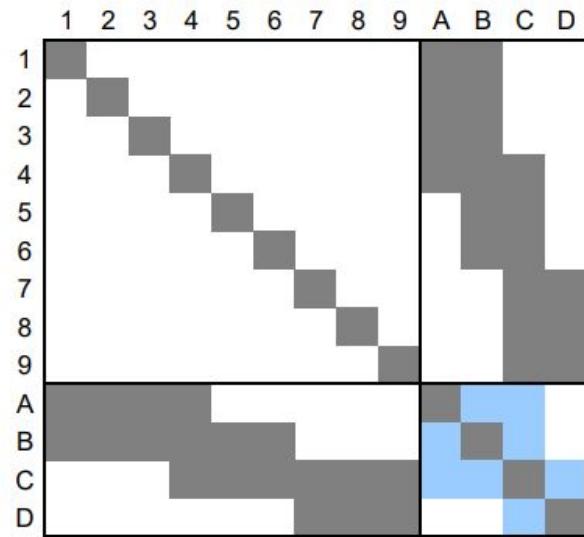


Bundle Adjustment. Schur complement

$$x_{n+1} = x_n - [H_r]^{-1} [y_r]_r$$

$$H_r(\Delta x) = y_r \quad r \equiv g$$

$$H_r(\Delta x) = g$$

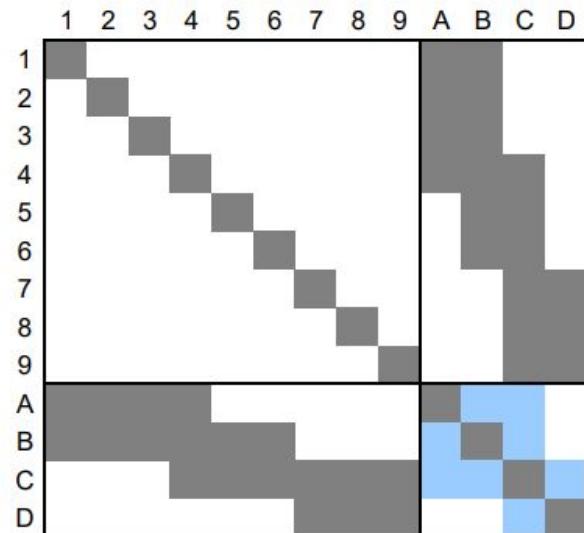


Bundle Adjustment. Schur complement

$$x_{n+1} = x_n - [H_r]^{-1} [y_r]_r$$

$$H_r(\Delta x) = y_r \quad r \equiv g$$

$$H_r(\Delta x) = g$$



для удобства переставим блоки

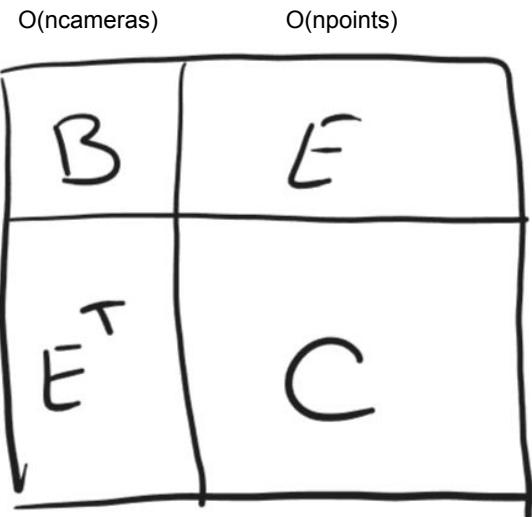


Bundle Adjustment. Schur complement

$$x_{n+1} = x_n - [H_r]^{-1} [y_r]_r$$

$$H_r(\Delta x) = y_r \quad r \equiv g$$

$$H_r(\Delta x) = g$$



Bundle Adjustment. Schur complement

$$x_{n+1} = x_n - [H_r]^{-1} [y_r]_r$$

$$H_r(\Delta x) = y_r \quad r \equiv q$$

$$H_r(\Delta x) = g$$

$$x = \begin{bmatrix} y_1 & \dots & y_p \\ \text{ext+int} & & \text{c param.} \end{bmatrix}, \begin{bmatrix} z_1 & \dots & z_q \\ xy, z & & s \text{ param.} \end{bmatrix}$$

p камер
q транс.
ext+int
c param.

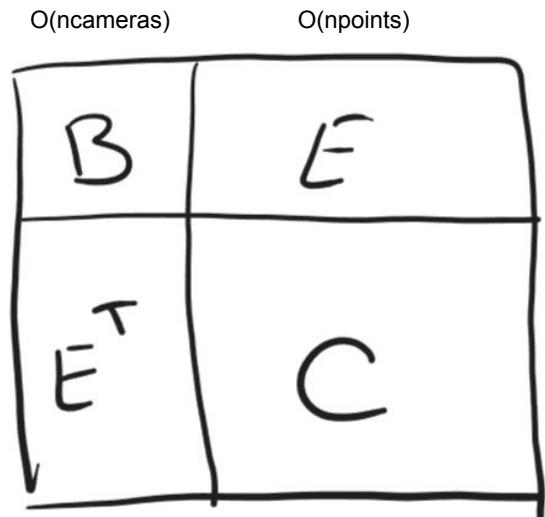
$$H_r = \begin{array}{|c|c|} \hline B & E \\ \hline E^T & C \\ \hline \end{array}$$

O(ncameras) O(npoints)

Bundle Adjustment. Schur complement

$$\begin{bmatrix} B & E \\ E^T & C \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}$$

$$H_v =$$

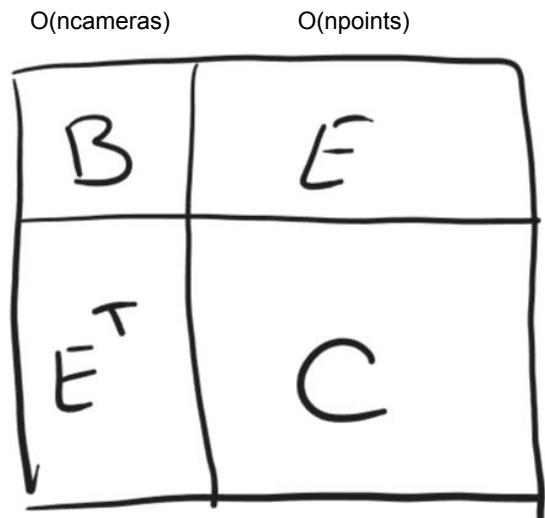


Bundle Adjustment. Schur complement

$$\begin{bmatrix} B & E \\ E^T & C \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}$$

C неявно обратим \Rightarrow находим C^{-1}

$$H_v =$$



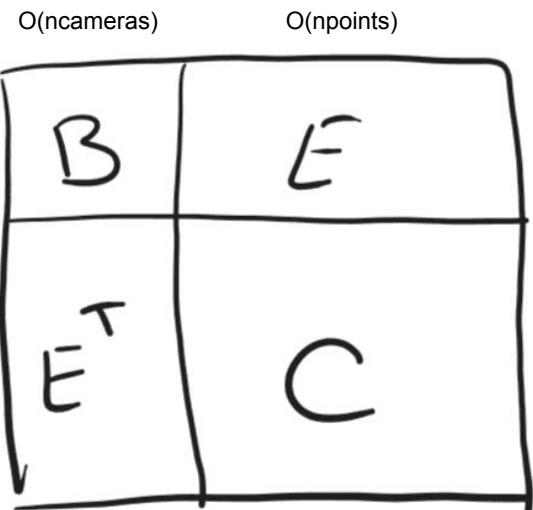
Bundle Adjustment. Schur complement

$$\begin{bmatrix} B & E \\ E^T & C \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}$$

C неявно обратим \Rightarrow имеем C^{-1}

$$\boxed{\Delta z = C^{-1}(w - E^T \Delta y)}$$

$$\begin{pmatrix} B \Delta y + E \Delta z \\ E^T \Delta y + C \Delta z \end{pmatrix} \underset{\Delta z = \star}{=} \begin{pmatrix} B \Delta y + E \star \\ \cancel{E^T \Delta y} + C \cancel{C^{-1}(w - E^T \Delta y)} \end{pmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}$$



Bundle Adjustment. Schur complement

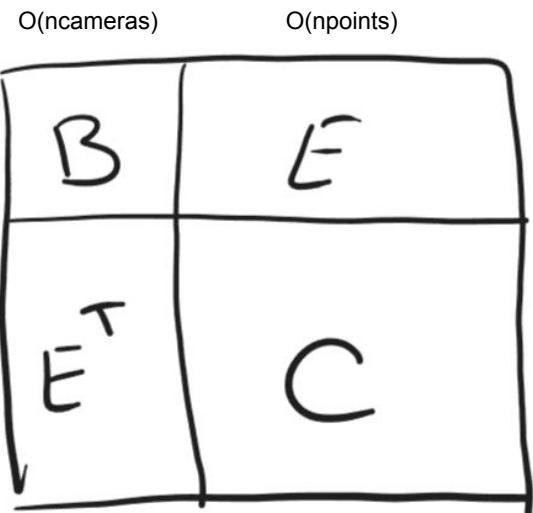
$$\begin{bmatrix} B & E \\ E^T & C \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}$$

C неявно оберните \Rightarrow находим C^{-1}

$$\boxed{\Delta z = C^{-1}(w - E^T \Delta y)}$$

$$\begin{pmatrix} B \Delta y + E \Delta z \\ E^T \Delta y + C \Delta z \end{pmatrix} \Big|_{\Delta z = \star} = \begin{pmatrix} B \Delta y + E \star \\ E^T \Delta y + C \star (w - E^T \Delta y) \end{pmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}$$

$$\boxed{\begin{pmatrix} B \Delta y + E \star \\ w \end{pmatrix} = \begin{pmatrix} v \\ w \end{pmatrix}} \quad \leftarrow \text{система размера } O(\text{количество камер}) !!$$



Фильтрация точек

- 1) После триангуляции точек и
после ВА могут появиться
выбросы

Фильтрация точек

- 1) После триангуляции точек и
после ВА могут появиться
выбросы
- 2) 3-сигма фильтрация

Фильтрация точек

- 1) После триангуляции точек и
после ВА могут появиться
выбросы
- 2) 3-сигма фильтрация
- 3) colinear points filtering

Фильтрация точек

- 1) После триангуляции точек и
после ВА могут появиться
выбросы
- 2) 3-сигма фильтрация
- 3) colinear points filtering
- 4) negative-z filtering

Забегая вперед, другие функции потерь

- 1) Наименьшие квадраты
неустойчивы к выбросам
- 2) Хотим функцию потерь
которая не будет
перетягивать одеяло на
единственного забежавшего
аутлаера
- 3) absolute loss, Huber loss, etc.

Ссылки

- 1) [Schur complement](#)
- 2) [Multiple View Geometry in Computer Vision, Richard Hartley & Andrew Zisserman](#) (здесь особенно много деталей, в т.ч. про L-M)
- 3) [Computer Vision: Algorithms and Applications, Richard Szeliski](#)

Вопросы?



Симиутин Борис
simiyutin.boris@yandex.ru⁸⁸