Лабораторная работа
Восстановление траектории и параметров движения куба
с использованием метода Левенберга-Марквардта

Студенты: Брызгалов Никита, Дубинин Андрей
Группа: 3383

13 декабря 2025 г.

Аннотация

В данной работе рассматривается задача восстановления полной траектории движения твердого тела (куба) по наблюдениям за координатами его вершин. Разработан алгоритм, позволяющий определить траекторию центра масс, параметры движения, геометрический размер куба и полное движение во все моменты времени. Основное внимание уделено реализации и анализу метода Левенберга-Марквардта как улучшенной версии метода Гаусса-Ньютона.

1 Введение

1.1 Постановка задачи

Имея наблюдения за положением вершин куба в каждый момент времени $V_{\text{obs}}[N, 8, 3]$, где N – число временных шагов, необходимо восстановить:

- Траекторию центра масс $\mathbf{r}(t)$ и скорость $\mathbf{v}(t)$
- Параметры движения: начальную ориентацию $\mathbf{q}_0$ и угловой момент $\mathbf{L}_0$
- Геометрический размер куба a
- Полное движение во все моменты времени, включая те, которые не наблюдались

1.2 Актуальность задачи

Задачи восстановления движения по видеонаблюдениям имеют широкое применение в компьютерном зрении, робототехнике, анализе спортивных движений и физических экспериментах.

2 Теоретическая часть

2.1 Кинематика твердого тела

Твердое тело характеризуется тем, что расстояния между любыми двумя его точками остаются неизменными. Движение твердого тела можно представить как сумму поступательного движения центра масс и вращательного движения вокруг центра масс.

2.1.1 Центр масс

Для куба с вершинами $\mathbf{v}_i$, $i = 1, \ldots, 8$, центр масс вычисляется как:

$$\mathbf{r}_{\text{CM}} = \frac{1}{8} \sum_{i=1}^{8} \mathbf{v}_i \tag{1}$$

2.1.2 Уравнения движения

Для свободного движения (без внешних сил):

$$\frac{d\mathbf{r}}{dt} = \mathbf{v} \tag{2}$$

$$\frac{d\mathbf{l}}{dt} = 0 \tag{3}$$

где $\mathbf{l} = m\mathbf{v}$ – импульс центра масс, m – масса куба.

2.2 Представление вращений

2.2.1 Кватернионы

Кватернион $\mathbf{q} = [w, x, y, z]$ компактно представляет вращение:

$$w = \cos(\theta/2) \tag{4}$$

$$[x, y, z] = \sin(\theta/2) \cdot \mathbf{n} \tag{5}$$

где θ – угол поворота, $\mathbf{n}$ – единичный вектор оси вращения.

Преимущества кватернионов:

- Отсутствие сингулярностей (проблемы блокировки карданова подвеса)
- Компактное представление (4 числа вместо 9 для матрицы)
- Эффективная интерполяция

3 Метод Гаусса-Ньютона: основа алгоритма

3.1 Математическая постановка

Метод Гаусса-Ньютона решает задачу нелинейного метода наименьших квадратов:

$$\min_{\mathbf{p}} \|\mathbf{r}(\mathbf{p})\|^2 = \min_{\mathbf{p}} \sum_{i=1}^{m} r_i^2(\mathbf{p}) \tag{6}$$

где:

- $\mathbf{p} = [r_x, r_y, r_z, l_x, l_y, l_z]^T$ – вектор параметров
- $\mathbf{r}(\mathbf{p}) = \mathbf{y}_{\text{model}}(\mathbf{p}) - \mathbf{y}_{\text{obs}}$ – вектор невязок
- $m = 3N$ – общее число наблюдений (N кадров $\times$ 3 координаты)

3.2 Итерационный алгоритм

Алгоритм метода Гаусса-Ньютона на каждой итерации выполняет следующие шаги:

1. Линеаризация невязок: $\mathbf{r}(\mathbf{p} + \Delta\mathbf{p}) \approx \mathbf{r}(\mathbf{p}) + \mathbf{J}(\mathbf{p})\Delta\mathbf{p}$
2. Решение нормальных уравнений: $\mathbf{J}^T\mathbf{J}\Delta\mathbf{p} = -\mathbf{J}^T\mathbf{r}$
3. Обновление параметров: $\mathbf{p}_{\text{new}} = \mathbf{p} - \alpha\Delta\mathbf{p}$

где α – коэффициент демпфирования.

3.3 Реализация на Python

Листинг 1: Реализация метода Гаусса-Ньютона

```
def gauss_newton_cm(y_obs_cm, dt, p0, npre=10, max_iter=20,
                    tol=1e-6, damping=1.0):
    p = p0.copy()
    for k in range(max_iter):
        J, r = jacobian_fd_cm(p, y_obs_cm, dt, npre=npre)
        JTJ = J.T @ J
        JTr = J.T @ r
        try:
            delta = np.linalg.solve(JTJ, JTr)  # p = (J J)    Jr
        except np.linalg.LinAlgError:
            break

        p_new = p - damping * delta
        if np.linalg.norm(delta) < tol:
            p = p_new
            break
        p = p_new
    return p
```

3.4 Вектор невязок

Вектор невязок $\mathbf{r}(\mathbf{p})$ вычисляется как разница между смоделированной и наблюдаемой траекториями:

Листинг 2: Функция вычисления невязок

```
def residuals_cm(params, y_obs_cm, dt, npre=10):
    y_model = simulate_observed_traj_cm(params, dt,
                                        y_obs_cm.shape[0], npre)

    return (y_model - y_obs_cm).ravel()
```

Если имеется N кадров наблюдений, то вектор $\mathbf{r}$ имеет размерность $3N$, где каждые три элемента соответствуют координатам (x, y, z) одного кадра.

3.5 Матрица Якобиана

Матрица Якобиана $\mathbf{J} \in \mathbb{R}^{3N \times 6}$ показывает чувствительность невязок к изменению параметров:

$$J_{ij} = \frac{\partial r_i}{\partial p_j} \tag{7}$$

Например, столбец $\mathbf{J}_{:,0}$ показывает, как изменятся все невязки при изменении начальной координаты x.

Листинг 3: Вычисление Якобиана методом конечных разностей

```
def jacobian_fd_cm(params, y_obs_cm, dt, npre, eps=1e-4):
    r0 = residuals_cm(params, y_obs_cm, dt, npre)  #
    m, n = r0.size, params.size
    J = np.zeros((m, n))

    for j in range(n):
        dp = np.zeros_like(params)
        dp[j] = eps  #                          j-

        r1 = residuals_cm(params + dp, y_obs_cm, dt, npre)

        J[:, j] = (r1 - r0) / eps

    return J, r0
```

3.6 Геометрическая интерпретация

Геометрический смысл метода Гаусса-Ньютона заключается в следующем:

1. Строится линейное приближение: $\mathbf{r}(\mathbf{p} + \Delta\mathbf{p}) \approx \mathbf{r} + \mathbf{J}\Delta\mathbf{p}$
2. Минимизируется квадрат линейного приближения: $\min_{\Delta\mathbf{p}} \|\mathbf{r} + \mathbf{J}\Delta\mathbf{p}\|^2$
3. Минимум достигается при решении системы: $\mathbf{J}^T\mathbf{J}\Delta\mathbf{p} = -\mathbf{J}^T\mathbf{r}$
4. Параметры обновляются: $\mathbf{p} \leftarrow \mathbf{p} - \alpha\Delta\mathbf{p}$

4 Численное интегрирование (RK4)

Для моделирования траектории используется метод Рунге-Кутты 4-го порядка (RK4), который интегрирует уравнения движения:

$$\frac{d\mathbf{r}}{dt} = \mathbf{v} \quad \text{(положение меняется со скоростью)} \tag{8}$$

$$\frac{d\mathbf{l}}{dt} = \mathbf{F} \quad \text{(импульс меняется под действием силы)} \tag{9}$$

Метод RK4 выполняется в несколько этапов:

1. Вычисление наклона (производной) в начальной точке

2. Вычисление наклона в средней точке с учетом первого наклона

3. Вычисление наклона в другой средней точке

4. Вычисление наклона в конечной точке

5. Усреднение всех четырех наклонов с соответствующими весами

Этот метод обеспечивает более высокую точность по сравнению с простым методом Эйлера.

5 Коррекция начальной ориентации

Наблюдения начинаются в момент времени $t = n_{\text{pre}}\Delta t$, тогда как нам нужны параметры в момент $t = 0$. Поэтому необходимо скорректировать начальную ориентацию:

Листинг 4: Коррекция начальной ориентации

```
def shift_orientation_back_to_start(q_obs0, omega, npre, dt):
    if np.isclose(np.linalg.norm(omega), 0.0):
        return q_obs0.copy()
    axis = omega / np.linalg.norm(omega)  #
    total_back_angle = -np.linalg.norm(omega) * (npre * dt)
    q_back = axis_angle_to_quat(axis, total_back_angle)
    q = quat_mult(q_back, q_obs0)

    return q
```

Умножение кватернионов $q_{\text{back}} \otimes q_{\text{obs0}}$ означает последовательное применение вращений:

1. Сначала применяется вращение q_{back} (поворот назад во времени на n_{pre} шагов)

2. Затем применяется вращение q_{obs0} (ориентация в момент начала наблюдений)

Таким образом, получаем начальную ориентацию куба в момент $t = 0$.

5.1 Метод Левенберга-Марквардта

5.1.1 Нелинейная задача наименьших квадратов

Восстановление параметров движения сводится к решению нелинейной задачи наименьших квадратов:

$$\min_{\mathbf{p}\in\mathbb{R}^6} f(\mathbf{p}) = \frac{1}{2}\sum_{i=1}^{3N} r_i^2(\mathbf{p}) = \frac{1}{2}\|\mathbf{r}(\mathbf{p})\|_2^2 \tag{10}$$

где:

- $\mathbf{p} = [r_x, r_y, r_z, l_x, l_y, l_z]^T$ – вектор параметров
- $\mathbf{r}(\mathbf{p}) = \mathbf{y}_{\text{model}}(\mathbf{p}) - \mathbf{y}_{\text{obs}}$ – вектор невязок
- N – число временных шагов

5.1.2 Линеаризация и квадратичная модель

Разложим функцию невязок в ряд Тейлора в окрестности текущей точки $\mathbf{p}_k$:

$$\mathbf{r}(\mathbf{p}_k + \Delta\mathbf{p}) \approx \mathbf{r}(\mathbf{p}_k) + \mathbf{J}(\mathbf{p}_k)\Delta\mathbf{p} \tag{11}$$

где $\mathbf{J}(\mathbf{p}_k) \in \mathbb{R}^{3N\times 6}$ – матрица Якоби:

$$\mathbf{J}(\mathbf{p}_k) = \begin{bmatrix} \frac{\partial r_1}{\partial p_1} & \cdots & \frac{\partial r_1}{\partial p_6} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_{3N}}{\partial p_1} & \cdots & \frac{\partial r_{3N}}{\partial p_6} \end{bmatrix}_{\mathbf{p}=\mathbf{p}_k} \tag{12}$$

Функция стоимости аппроксимируется квадратичной формой:

$$f(\mathbf{p}_k + \Delta\mathbf{p}) \approx \frac{1}{2}\|\mathbf{r}_k + \mathbf{J}_k\Delta\mathbf{p}\|^2 = f_k + \mathbf{g}_k^T\Delta\mathbf{p} + \frac{1}{2}\Delta\mathbf{p}^T\mathbf{H}_k\Delta\mathbf{p} \tag{13}$$

где:

- $f_k = \frac{1}{2}\|\mathbf{r}_k\|^2$
- $\mathbf{g}_k = \mathbf{J}_k^T\mathbf{r}_k$ – градиент
- $\mathbf{H}_k = \mathbf{J}_k^T\mathbf{J}_k$ – приближение Гессиана

5.1.3 Регуляризованная система уравнений

Метод Левенберга-Марквардта решает регуляризованную систему:

$$(\mathbf{J}_k^T\mathbf{J}_k + \lambda_k\mathbf{D}_k)\Delta\mathbf{p}_k = -\mathbf{J}_k^T\mathbf{r}_k \tag{14}$$

где:

- $\lambda_k > 0$ – параметр регуляризации
- $\mathbf{D}_k = \mathrm{diag}(\mathbf{J}_k^T\mathbf{J}_k)$ – диагональная матрица (метод Марквардта)

5.1.4 Геометрическая интерпретация

Задача (14) эквивалентна минимизации с ограничением:

$$\min_{\Delta\mathbf{p}} \|\mathbf{J}_k\Delta\mathbf{p} + \mathbf{r}_k\|^2 \quad \text{s.t.} \quad \|\mathbf{D}_k^{1/2}\Delta\mathbf{p}\| \le \delta_k \tag{15}$$

где δ_k – радиус доверительной области, а λ_k – множитель Лагранжа.

5.1.5 Адаптивное управление параметром λ_k

Параметр λ_k адаптивно изменяется на основе отношения фактического уменьшения стоимости к предсказанному:

$$\rho_k = \frac{f(\mathbf{p}_k) - f(\mathbf{p}_k + \Delta\mathbf{p}_k)}{\Delta f_{\text{pred}}} \tag{16}$$

где $\Delta f_{\text{pred}} = -\Delta\mathbf{p}_k^T\mathbf{g}_k - \frac{1}{2}\Delta\mathbf{p}_k^T\mathbf{H}_k\Delta\mathbf{p}_k$.

Алгоритм управления λ_k (эвристика Нильсена):

$$\lambda_{k+1} = \begin{cases} \lambda_k \cdot \max\left(\frac{1}{3}, 1 - (2\rho_k - 1)^3\right) & \text{если } \rho_k > 0 \\ \lambda_k \cdot \nu & \text{иначе} \end{cases} \tag{17}$$

где $\nu = 2 \div 10$ – фактор увеличения.

Аспект	**Гаусс-Ньютон**	**Левенберг-Марквардт**
Сходимость	Локальная	Глобальная
Устойчивость	Чувствителен к сингулярностям	Устойчив за счет регуляризации
Скорость сходимости	Квадратичная вблизи решения	Суперлинейная
Параметры настройки	Коэффициент демпфирования	Начальное значение λ

Таблица 1: Сравнение методов оптимизации

5.1.6 Сравнение с методом Гаусса-Ньютона

6 Алгоритмическая реализация

6.1 Общая схема алгоритма

Algorithm 1 Общая схема восстановления траектории

procedure ВосстановлениеТраектории(V_{obs})
  $\mathbf{y}_{\text{obs}}^{\text{CM}} \leftarrow$ ВычислитьЦентрМасс(V_{obs})
  $a \leftarrow$ ОценитьРазмерКуба($V_{\text{obs}}[0]$)
  $\mathbf{q}_0^{\text{obs}}, \mathbf{L}_0 \leftarrow$ ОценитьОриентациюИМомент(V_{obs}, a)
  $\mathbf{p}^* \leftarrow$ ЛевенбергМарквардт($\mathbf{y}_{\text{obs}}^{\text{CM}}, \mathbf{p}_0$)
  $\mathbf{q}_0 \leftarrow$ КоррекцияОриентации($\mathbf{q}_0^{\text{obs}}, \mathbf{L}_0/a^2$)
  Траектория $\leftarrow$ ВосстановитьПолнуюТраекторию($\mathbf{p}^*, a, \mathbf{q}_0, \mathbf{L}_0$)
  return Траектория, $\mathbf{p}^*, a, \mathbf{q}_0, \mathbf{L}_0$
end procedure

6.2 Ключевые этапы алгоритма

6.2.1 Восстановление траектории центра масс

Центр масс куба вычисляется как среднее арифметическое координат всех его вершин:

$$\mathbf{y}_{\text{obs}}^{\text{CM}}[k] = \frac{1}{8} \sum_{i=1}^{8} V_{\text{obs}}[k, i, :] \tag{18}$$

6.2.2 Оценка размера куба

Размер куба оценивается по геометрическим свойствам:

Algorithm 2 Оценка размера куба

function ОценитьРазмерКуба(вершины)
  Вычислить все попарные расстояния между вершинами
  Отсортировать расстояния по возрастанию
  Взять 12 минимальных расстояний (ребра куба)
  $a \leftarrow$ среднее(12 минимальных расстояний)/2
  return a
end function

6.2.3 Задача Кабеша (Kabsch) для оценки ориентации

Для каждого кадра находится оптимальное вращение $\mathbf{R}_k$, минимизирующее:

$$\min_{\mathbf{R}} \|\mathbf{RA} - \mathbf{B}\|_F^2 \tag{19}$$

где $\mathbf{A}$ – вершины канонического куба, $\mathbf{B}$ – наблюдаемые вершины.

Решение через SVD: если $\mathbf{H} = \mathbf{A}^T\mathbf{B} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, то $\mathbf{R} = \mathbf{V}\mathbf{U}^T$.

6.2.4 Вычисление Якобиана методом конечных разностей

Поскольку аналитические производные недоступны, используется метод центральных разностей:

$$\frac{\partial r_i}{\partial p_j} \approx \frac{r_i(\mathbf{p} + h\mathbf{e}_j) - r_i(\mathbf{p} - h\mathbf{e}_j)}{2h} \tag{20}$$

где $h = \epsilon \cdot \max(1, |p_j|)$ – адаптивный шаг.

6.3 Реализация метода Левенберга-Марквардта

Algorithm 3 Метод Левенберга-Марквардта

function ЛевенбергМарквардт($\mathbf{y}_{\text{obs}}, \mathbf{p}_0, \lambda_0, \text{tol}$)
  $\mathbf{p} \leftarrow \mathbf{p}_0$, $\lambda \leftarrow \lambda_0$
  for $k = 0$ to max_iter **do**
    $\mathbf{J}, \mathbf{r} \leftarrow$ ВычислитьЯкобиан($\mathbf{p}, \mathbf{y}_{\text{obs}}$)
    $\mathbf{g} \leftarrow \mathbf{J}^T\mathbf{r}$ ▷ Градиент
    $\mathbf{H} \leftarrow \mathbf{J}^T\mathbf{J}$ ▷ Приближение Гессиана
    $\mathbf{D} \leftarrow \text{diag}(\mathbf{H})$ ▷ Метод Марквардта
    $\mathbf{H}_{\text{reg}} \leftarrow \mathbf{H} + \lambda\mathbf{D}$
    $\Delta\mathbf{p} \leftarrow$ решить($\mathbf{H}_{\text{reg}}, -\mathbf{g}$) LinAlgError
    $\lambda \leftarrow 10\lambda$

    if $\|\Delta\mathbf{p}\| < 10^{-10}$ **then**
    end if
    $\mathbf{p}_{\text{new}} \leftarrow \mathbf{p} + \Delta\mathbf{p}$
    $\mathbf{r}_{\text{new}} \leftarrow$ ВычислитьНевязки($\mathbf{p}_{\text{new}}, \mathbf{y}_{\text{obs}}$)
    $f_{\text{old}} \leftarrow \frac{1}{2}\|\mathbf{r}\|^2$, $f_{\text{new}} \leftarrow \frac{1}{2}\|\mathbf{r}_{\text{new}}\|^2$
    if $f_{\text{old}} - f_{\text{new}} > 0$ **then**
      $\mathbf{p} \leftarrow \mathbf{p}_{\text{new}}$, $\mathbf{r} \leftarrow \mathbf{r}_{\text{new}}$
      $\lambda \leftarrow \max(10^{-12}, 0.1\lambda)$
    else
      $\lambda \leftarrow \min(10^{12}, 10\lambda)$
    end if
    if $\|\mathbf{g}\| <$ tol или $\|\Delta\mathbf{p}\| <$ tol$\|\mathbf{p}\|$ **then**
    end if
  end for
  return p
end function

7 Взаимодействие численного интегрирования и оптимизации

Численное интегрирование (метод Рунге-Кутты 4-го порядка) играет ключевую роль в оптимизации:

- Для вычисления невязок $\mathbf{r}(\mathbf{p})$ необходимо проинтегрировать уравнения движения с параметрами $\mathbf{p}$
- Для вычисления Якобиана $\mathbf{J}$ требуется $(n + 1)$ интегрирований на итерацию, где $n = 6$ – число параметров
- Каждое интегрирование моделирует движение куба на $N + n_{\text{pre}}$ шагов

Вычислительная сложность одной итерации: $O(mn^2 + n^3)$, где $m = 3N$ – число наблюдений, $n = 6$ – число параметров.

8 Экспериментальные результаты

8.1 Сходимость алгоритма

Алгоритм демонстрирует быструю сходимость:

```
Iter 0: cost=1.139e+01, =1.0e-04, cost=1.8e+06
Iter 1: cost=6.276e-05, =1.0e-05, cost=1.1e+01
Iter 2: cost=4.346e-05, =1.0e-06, cost=1.9e-05
Iter 3: cost=4.346e-05, =1.0e-07, cost=3.4e-13
```

8.2 Точность восстановления параметров

Параметр	**Истинное значение**	**Восстановленное**	**Ошибка**
r_x (м)	0.000000	0.000034	0.034 мм
r_y (м)	15.000000	14.999962	0.038 мм
r_z (м)	0.000000	-0.000121	0.121 мм
l_x (Н·с)	10.000000	9.999995	0.000005
l_y (Н·с)	10.000000	10.000003	0.000003
l_z (Н·с)	5.500000	5.500011	0.000011
Размер (м)	2.000000	2.000035	0.035 мм

Таблица 2: Точность восстановления параметров

Относительная ошибка не превышает 0.002%, что свидетельствует о высокой точности алгоритма.

9 Заключение

Разработанный алгоритм на основе метода Левенберга-Марквардта успешно решает задачу восстановления траектории и параметров движения куба. Основные результаты:

- Достигнута высокая восстановления параметров
- Алгоритм демонстрирует быструю сходимость (3-4 итерации)
- Метод Левенберга-Марквардта показал лучшую устойчивость по сравнению с методом Гаусса-Ньютона
- Реализация не требует ручной настройки параметров благодаря адаптивному управлению λ

Алгоритм может быть применен для решения широкого класса задач в компьютерном зрении, робототехнике и физическом моделировании.

Приложение: Основные формулы

$$\text{Центр масс: } \mathbf{r}_{\text{CM}} = \frac{1}{8}\sum_{i=1}^{8}\mathbf{v}_i \tag{21}$$

$$\text{Задача Кабеша: } \mathbf{R} = \mathbf{V}\mathbf{U}^T, \text{ где } \mathbf{A}^T\mathbf{B} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{22}$$

$$\text{Уравнение Левенберга-Марквардта: } (\mathbf{J}^T\mathbf{J} + \lambda \text{diag}(\mathbf{J}^T\mathbf{J}))\Delta\mathbf{p} = -\mathbf{J}^T\mathbf{r} \tag{23}$$

$$\text{Момент инерции куба: } I = \frac{ma^2}{6} \quad (m = 1 \text{ в работе}) \tag{24}$$