Санкт-Петербургский государственный университет

Программная инженерия

Кафедра системного программирования

Лень Ирина Александровна

Система оптимизации распределения наблюдателей между источниками сигналов

Бакалаврская работа

Научный руководитель:
д. ф.-м. н., профессор Граничин О. Н.

Рецензент:
к. ф.-м. н., ст. науч. сотрудник ИПМаш РАН Шалымов Д. С.

Санкт-Петербург
2019

SAINT PETERSBURG STATE UNIVERSITY

Software engineering

Irina Len

System for optimizing the assignment of observers between signal sources

Graduation Thesis

Scientific supervisor:
Doctor of Science, Professor O. N. Granichin

Reviewer:
PhD, Senior researcher IPME RAS D. S. Shalymov

Saint Petersburg
2019

Оглавление

Введение **4**

Цель работы **6**

1. Математическая модель **7**

1.1. Модель наблюдения 7

1.2. Доверительные эллипсоиды 8

1.3. Функционал качества 9

1.4. Алгоритм полного перебора (Brute Force) 11

1.5. Линейные матричные неравенства 11

2. Описание системы **13**

2.1. Общее описание и функциональность 13

2.2. Визуализация 14

2.3. Реализации алгоритма оптимизации 15

3. Эксперименты **19**

3.1. Сравнение математических моделей 19

3.2. Нагрузочное тестирование 20

Заключение **22**

Список литературы **23**

Введение

Активное развитие средств беспроводной коммуникации привело к повышенному интересу со стороны исследователей, специализирующихся в области использования сенсорных сетей (например, группы роботов или наблюдателей). Наиболее часто упоминаются задачи, связанные с интеллектуальным слежением в местах большого скопления людей, исследованием миграций животных, изучением космоса [1, 2, 3].

Использование данных с нескольких датчиков может помочь понизить погрешность итогового измерения, вызванного неидеальной точностью самого прибора или возмущениями из внешней среды. Однако при увеличении количества сенсоров и объектов наблюдения возникают проблемы, связанные с коммуникациями, ограничениями пропускной способности каналов связи и вычислительной мощности устройств [4]. Следовательно, желательно ограничить количество целей, за которыми следит каждый сенсор и при этом не сильно потерять в точности вычислений. Это актуализирует задачу оптимизации распределения целей между наблюдателями.

В последнее время возрос интерес к построению и развитию адаптивных систем, использующих мультиагентные технологии. Существуют несколько подходов к организации сенсорных сетей: централизованный, распределенный и гибридный. “Традиционным подходом” считается создание или выделение одного главного (центрального) узла, в котором собираются все данные наблюдений. Это дает возможность получить наиболее точную оценку измерения.

В настоящее время больший упор при разработке сложных систем делается на создание распределенных и гибридных сетей, состоящих из нескольких небольших групп независимых друг от друга агентов. Эти подходы обладают более высокой степенью отказоустойчивости, масштабируемостью и меньшими, в сравнении с централизованным подходом, требованиями к пропускной способности сети. В таких сетях каждый агент имеет взаимосвязи только с неким подмножеством агентов своей подгруппы, а не со всеми. Благодаря объединению этих агентов и

достигается поставленная для них общая задача. Значит, имеется возможность заменить решение сложной задачи большой размерности на набор более простых задач с одной целью [5, 6, 7].

Фильтр Калмана [8, 9] является одним из популярных алгоритмов для наблюдения за разным количеством целей, однако у этого алгоритма есть один недостаток: при увеличении количества целей вычислительная нагрузка значительно возрастает. В статьях [10, 11] описано решение этой проблемы с помощью распределенной схемы фильтрации Калмана. Дальнейшие исследования в этой области привели к разработке алгоритмов с учетом проблем сенсорных сетей, описанных выше [12, 13].

Вышеизложенное указывает актуальность задачи разработки алгоритмов разбиения сенсоров небольшие группы слежения за объектами, если это возможно. Эта задача была сформулирована в строгой математической форме в статье [14], где для ее решения авторы предложили алгоритм на основе метода линейных матричных неравенств (LMI). В статье [15] эксперементально подтверждена эффективность этого метода в сравнении с методом полного перебора. Однако выяснено, что некоторые сенсоры вынуждены следить за всеми объектами, а значит система загружена неравномерно. Таким образом актуализируется вопрос о сбалансированности нагрузки между сенсорами внутри сети.

Цель работы

Целью работы является разработка прототипа системы для автоматизации процесса оптимизации распределения наблюдателей между источниками сигналов.

Для достижения вышеупомянутой цели были сформулированы следующие задачи.

1. Расширить существующую модель распределения сенсоров и целей.

2. Разработать прототип системы, включающий в себя возможность визуализации задачи и настройки параметров для ее расчета.

3. Сравнить распределение наблюдателей между источниками сигналов в существующей и новой моделях. Провести нагрузочное тестирование для алгоритма на основе линейных матричных неравенств (LMI) и полного перебора (Brute Force).

1. Математическая модель

1.1. Модель наблюдения

Рассмотрим математическую модель из [14], в которой есть сеть из m сенсоров и n целей (Рис. 1).

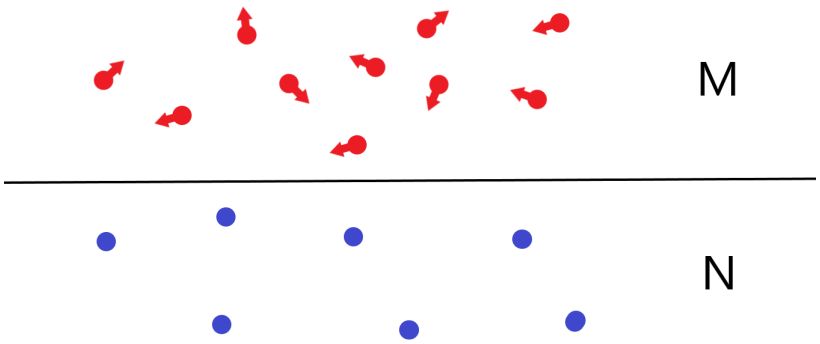


Рис. 1: Иллюстрация к постановке задачи, где датчики (сенсоры) обозначены синим, их $n = 7$, а цели — красным, их $m = 10$

Пусть $N = \{1, 2, \ldots, n\}$ — множество наблюдателей (сенсоров), $s_t^j \in \mathbb{R}^d, j \in N$ — вектор текущего состояния сенсора j (его положение в пространстве) в момент времени t.

$M = \{1, 2, \ldots, m\}$ — множество целей (источников сигналов), $x_t^i \in \mathbb{R}^p, i \in M$ — вектор состояния i-ой цели в момент времени t.

$$x_{t+1}^i = f^i(x_t^i) + w_t^i,$$

где f^i — некая функция динамики, $\{w_t^i\}$ — белый шум с нулевым мат. ожиданием $\mathrm{E}w_t^i = 0$.

Пусть от каждого датчика j поступают данные $y_t^{i,j} \in \mathbb{R}^d, i \in M, j \in N$ о наблюдаемой траектории i-ой цели в момент времени t, которые задаются следующим уравнением:

$$y_t^{i,j} = \varphi(s_t^j, x_t^i) + v_t^{i,j},$$

где $\varphi(s_t^j, x_t^i)$ — функция, отражающая измерения сенсором j объекта i, $\{v_t^{i,j}\}$ — независимые помехи в измерениях с нулевым мат. ожида-

нием $\mathrm{E}v_t^{i,j} = 0$ и ковариацией $\mathrm{E}v_t^{i,j}(v_t^{i,j})^{\mathrm{T}} = \Sigma_t^{i,j}$.(В работе символом $\cdot^{\mathrm{T}}$ обозначена операция транспонирования вектора или матрицы, верхние индексы являются номерами). Сенсоры позволяют хранить информацию о последних r наблюдениях в моменты времени $t-1, t-2, \ldots, t-r$.

Далее в работе будем рассматривать двумерное пространство. Расположение целей и сенсоров опишем следующим образом $x_t^i = \begin{bmatrix} x_t^{i,1} & x_t^{i,2} \end{bmatrix} \in \mathbb{R}^2$, $s_t^j = \begin{bmatrix} s_t^{j,1} & s_t^{j,2} \end{bmatrix} \in \mathbb{R}^2$. Определим функцию φ следующим образом:

$$\varphi(s_t^j, x_t^i) = \begin{bmatrix} \psi(s_t^j, x_t^i) \\ \rho(s_t^j, x_t^i) \end{bmatrix},$$

где $\psi(s_t^j, x_t^i) = \operatorname{arctg}\left[\frac{x_t^{i,1} - s_t^{j,1}}{x_t^{i,2} - s_t^{j,2}}\right]$ — угол меду направлением на цель и направлением на север;

$\rho(s_t^j, x_t^i) = \sqrt{\left(s_t^{j,1} - x_t^{i,1}\right)^2 + \left(s_t^{j,2} - x_t^{i,2}\right)^2}$ — расстояние от сенсора до цели.

1.2. Доверительные эллипсоиды

Поскольку датчики работают с погрешностью, для каждого его измерения мы будем строить доверительный эллипсоид (эллипс рассеивания в R^2) с уровнем достоверности $p \in [0, 1]$.

Пусть для функции наблюдения существует обратная по второму аргументу функция

$$\varphi^{-1}(s_t^i, y_t^{i,j}) = x_t^i + \xi_t^{i,j} = \eta^{i,j}, \tag{1}$$

где $\xi_t^{i,j}$ — независимые с нулевым средним $\mathrm{E}\xi_t^{i,j} = 0$ и ковариацией $\mathrm{E}\xi_t^{i,j}(\xi_t^{i,j})^T = \Xi_t^{i,j}$. Тогда доверительный эллипсоид задается формулой:

$$\mathcal{E}_t^{i,j} = \{x_t^i \mid (x_t^i - \eta^{i,j})^T (\Xi_t^{i,j})^{-1} (x_t^i - \eta^{i,j}) \leq \chi^2_{p,d}, \} \tag{2}$$

где $\eta^{i,j}$ — точка результата наблюдения сенсора j для цели i из формулы (1), $\chi^2_{p,d}$ — коэффициент χ^2 распределения с d степенями свободы.

Зададим эллипсоид одной из следующих эквивалентных формул:

$$\mathcal{E} = \{x \mid (x - x_c)^T P^{-2}(x - x_c) \leq 1\} = \{Pz + x_c \mid \|z\| \leq 1\}, P = P^T > 0 \quad (3)$$

— как прообраз единичной сферы при аффинном преобразовании.

$$\mathcal{E} = \{x \mid x^T Ax + 2x^T b + c \leq 0\}, A = A^T > 0, b^T A^{-1} b - c > 0 \quad (4)$$

— через выпуклые квадратичные функции.

1.3. Функционал качества

Пересечение нескольких эллипсоидов даст точную оценку возможного расположения объекта (рис. 2). В работе [15] доказано, что вероятность нахождения искомой точки внутри пересечения двух доверительных эллипсоидов равна $1 - 3p$.

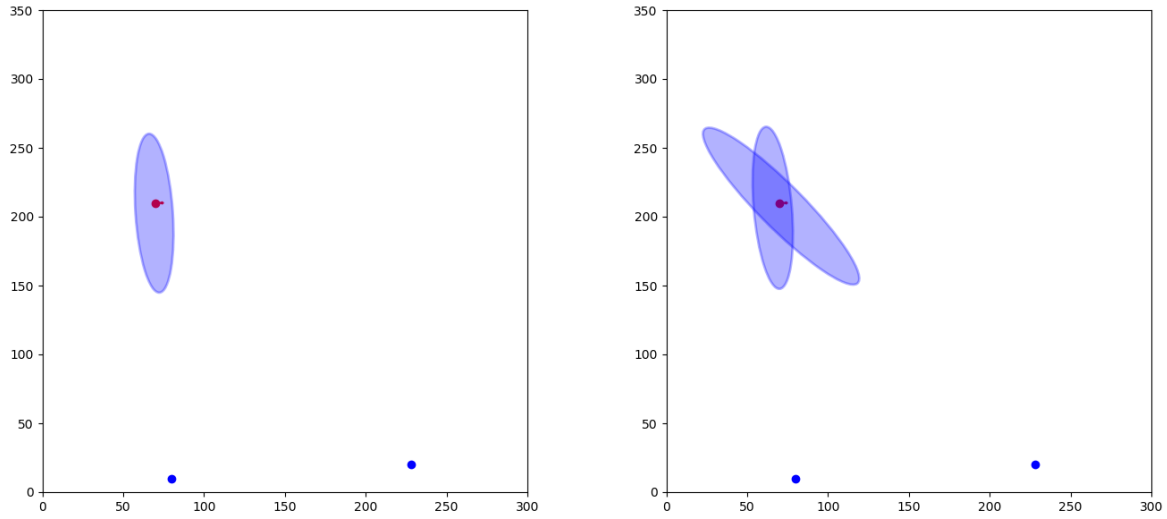


Рис. 2: Данные о положении цели при использовании 1 и 2 сенсоров соответственно

В статье [15] предлагалось для предсказания траектории цели i минимизировать объем пресечения эллипсоидов и учитывать количество сенсоров, которое следит за ней. Исходя из этого, решалась следующая

задача минимизации:

$$F_t = \sum_{i=1}^{m} \Phi_t^i + \alpha \sum_{i=1}^{m} |S_t^i| \to \min, \tag{5}$$

где Φ_t^i — объем пересечения доверительных эллипсоидов для i-ой цели, S_t^i — множество сенсоров, следящих за i-ой целью (группа слежения), $|\cdot|$ — мощность множества (количество элементов), α — регуляризующий коэффициент.

Однако такая функция минимизация имеет один недостаток — некоторые сенсоры вынуждены следить за всеми целями. Отсюда возникла идея добавления ограничения на количество целей, за которым следит каждый из сенсоров.

$$F_t = \sum_{i=1}^{m} \Phi_t^i + \alpha \sum_{i=1}^{m} |S_t^i| + \beta \sum_{j=1}^{n} |T_t^j| \to \min, \tag{6}$$

где $\Phi_t^i,\ S_t^i,\ \alpha$ определяются как в (5), T_t^i — множество целей, за которым следит j-ый сенсор, β — малый коэффициент регуляризации. По сути α отвечает за вклад в функционал качества стоимостей коммуникации между сенсорами, а β — за ограниченность ресурсов сенсоров.

Введем матрицу ресурсов $G_t = [g_t^{i,j}] : g_t^{i,j} > 0$, если сенсор j следит за i-ой целью, иначе $g_t^{i,j} = 0$.

Предлагается решить задачу (6) "овыпуклением" с помощью специальных матричных норм для получения разреженного субоптимального решения, что приведет нас к задаче следующего вида:

$$F_t' = \sum_{i=1}^{m} \Phi_t^i + \alpha \sum_{i=1}^{m} \|G_t^{(i,\cdot)}\|_1 + \beta \sum_{j=1}^{n} \|G_t^{(\cdot,j)}\|_1 \to \min, \tag{7}$$

где $\|\cdot\|_1$ — l_1-норма: $\|x\|_1 = \sum_{i=1}^{v} |x_i|$, $G_t^{(i,\cdot)}$ — множество сенсоров, следящих за i-ой целью, $G_t^{(\cdot,j)}$ — множество целей, за которым следит j-й сенсор.

1.4. Алгоритм полного перебора (Brute Force)

Brute Force, или алгоритм полного перебора, принадлежит к классу способов поиска решения исчерпыванием всех возможных вариантов. В текущей задаче необходимо проверить все варианты распределения целей между сенсорами и комбинации этих распределений, чтобы узнать эффективность каждого из вариантов и найти среди них оптимальное значение. Этот алгоритм дает оптимальное решение, но на практике является ресурсозатратным и имеет экспоненциальную временную сложность.

Для его реализации необходимо заполнить матрицу ресурсов G_t всеми возможными комбинациями из 0 и 1, исключая варианты с нулевыми строками (т.е за целью обязательно должен кто-нибудь следить). После этого расчитываем суммы объемов и значения функционала из формулы (7) для каждой такой матрицы.

1.5. Линейные матричные неравенства

В теории робастного управления в процессе исследования систем с неопределенностью возникла l_1 оптимизация [17], которая совместно с линейными матричными неравенствами (LMI) [18, 19] дала удобный инструмент решения ряда задач. Развитие этой темы можно проследить в статьях [20, 21, 22, 23]

1.5.1. Алгоритм поиска аппроксимирующего эллипса

Задача поиска пересечения l эллипсоидов является NP-полной задачей. Поэтому введем эллипсоид $\mathcal{E}_0 \supseteq \bigcap_{i=1}^{l} \mathcal{E}_i$, который содержит пересечение эллипсоидов $\mathcal{E}_i, i = 1, \ldots, l$.

Чтобы получить наименьший по объему $\mathcal{E}_0$, необходимо решить следующую LMI-задачу [19]:

$$\text{minimize } \log\det A_0^{-1} \tag{8}$$

$$\text{subject to } A_0 > 0, \tau_1 \geq 0, \ldots, \tau_p \geq 0$$

$$\begin{bmatrix} A_0 & b_0 & 0 \\ b_0^T & -1 & b_0^T \\ 0 & b_0 & -A_0 \end{bmatrix} - \sum_{i=1}^{p} \tau_i \begin{bmatrix} A_i & b_i & 0 \\ b_i^T & c_i & 0 \\ 0 & 0 & 0 \end{bmatrix} \leq 0,$$

где A_i, b_i, c_i — параметры, описывающие i-й эллипсоид по формуле (4). Решением задачи (8) являются параметры $A_0, b_0, c_0 = b_0^T A_0^{-1} b_0 - 1$, аналогично определяющие $\mathcal{E}_0$.

1.5.2. Алгоритм оптимизации

В статье [15] для поиска оптимального распределения целей и сенсоров предлагался алгоритм на основе LMI. По аналогии с ним, запишем следующую задачу полуопределенного программирования:

$$\text{minimize } \delta \tag{9}$$

$$\text{subject to } \forall i \in [1, m]\ \hat{\mathbf{A}}_t^i > 0,\ g_t^{i,1} \geq 0, \ldots, g_t^{i,n} \geq 0,$$

$$\begin{bmatrix} \hat{\mathbf{A}}_t^i & \hat{\mathbf{b}}_t^i & 0 \\ (\hat{\mathbf{b}}_t^i)^T & -1 & (\hat{\mathbf{b}}_t^i)^T \\ 0 & \hat{\mathbf{b}}_t^i & -\hat{\mathbf{A}}_t^i \end{bmatrix} - \sum_{j=1}^{n} g_t^{i,j} \begin{bmatrix} \mathbf{A}_t^{i,j} & \mathbf{b}_t^{i,j} & 0 \\ (\mathbf{b}_t^{i,j})^T & \mathbf{c}_t^{i,j} & 0 \\ 0 & 0 & 0 \end{bmatrix} \leq 0$$

$$\sum_{i=1}^{m} \log\det(\hat{\mathbf{A}}_t^i)^{-1} + \alpha \sum_{i=1}^{m} \|\mathbf{G}_t^{(i,\cdot)}\|_1 + \beta \sum_{j=1}^{n} \|\mathbf{G}_t^{(\cdot,j)}\|_1 \leq \delta,$$

где $\mathbf{A}_t^{i,j}, \mathbf{b}_t^{i,j}, \mathbf{c}_t^{i,j}$ — описание доверительных эллипсоидов (4), построенных для цели i сенсором j в момент времени t, α — стоимость коммуникации между сенсорами, а β — ограничение на ресурсы сенсоров.

Решением вышеупомянутой задачи являются матрица ресурсов $\mathbf{G}_t$, отвечающая за распределение наблюдателей между источниками сигналов, и параметры $\hat{\mathbf{A}}_t^i, \hat{\mathbf{b}}_t^i, \hat{\mathbf{c}}_t^i = (\hat{\mathbf{b}}_t^i)^T (\hat{\mathbf{A}}_t^i)^{-1} (\hat{\mathbf{b}}_t^i) - 1$ эллипсоидов пересечения для i-й цели.

Другими словами, задача состоит в минимизации объема аппроксимирующих пересечение эллипсоидов, количества сенсоров в группе слежения за целью и количества целей, за которым следит каждый сенсор.

2. Описание системы

2.1. Общее описание и функциональность

В процессе выполнения выпускной квалификационной работы был разработан прототип системы оптимизации распределения наблюдателей между источниками сигналов, помогающий пользователю решить задачу распределения целей между сенсорами, визуализировать их состояния, сравнить результаты моделирования при разных значениях параметров. В качестве языка программирования был выбран Python 3 [24], так как он поддерживает библиотеку cvxpy [25], решающий задачи полуопределенного программирования, необходимый для реализации алгоритмов.

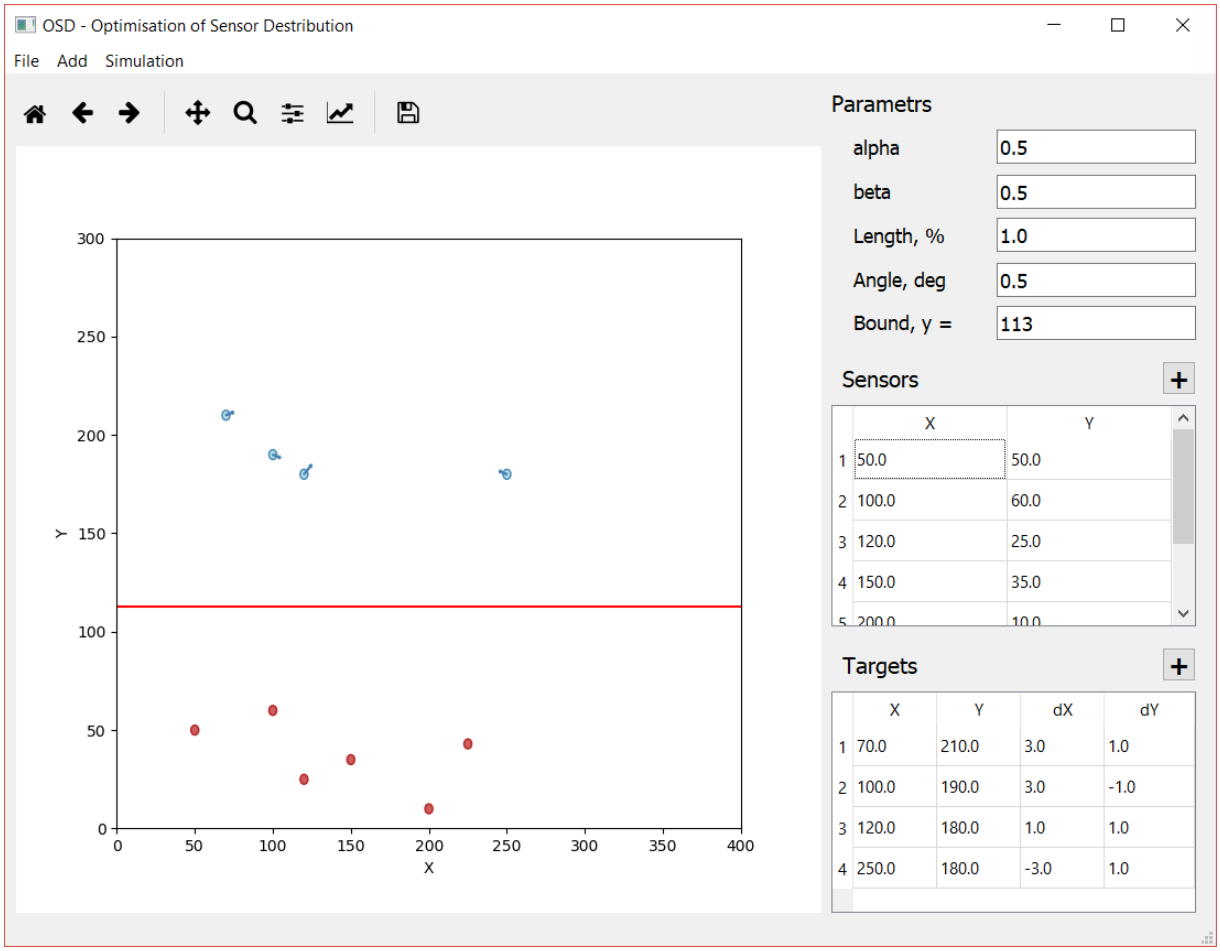


Рис. 3: Графический интерфейс системы.

На рисунке 3 изображен пользовательский интерфейс системы. На-

вигационная панель позволяет создавать, сохранять и загружать данные о модели, выставлять настройки отображения эксперимента. Также представлена возможность добавления и удаления сенсоров и целей в уже существующую модель, настройка параметров системы: ошибок измерения сенсором угла (в градусах) и расстояния (в процентах), добавление границы опасной для целей зоны. Для хранения информации о математической модели, сенсорах и целях используется XML-файл.

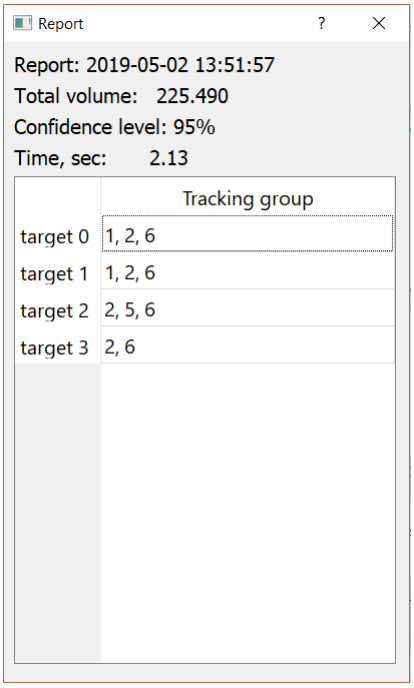


Рис. 4: Пример отображения результата моделирования.

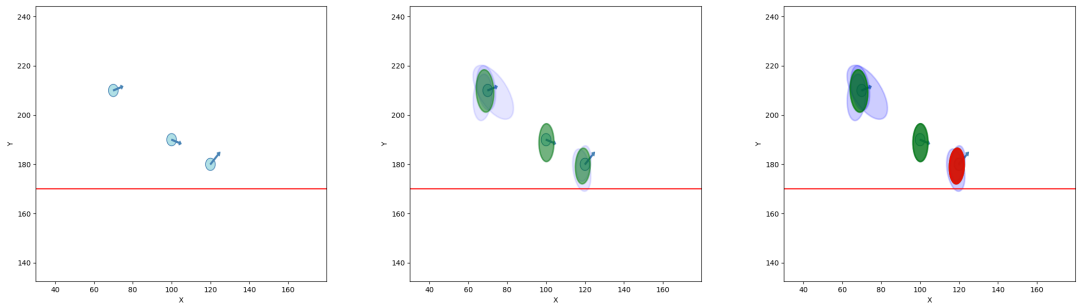
В отдельном окне, как на рисунке 4, отображается отчет о результатах моделирования, который содержит информацию о дате и времени эксперимента, сумме объемов аппроксимирующих эллипсоидов, скорости работы алгоритма в секундах и таблицы с оптимальными группами слежения для каждой цели. Дополнительно формируется отчет в формате текстового файла.

2.2. Визуализация

Для отображения используется “поле” размером 300×400 км^2, для которого реализованы функции перемещения “поля”, масштабирования,

возврата к предыдущему, последующему и начальному значениям масштаба, сохранение его изображения.

Система позволяет настроить следующие параметры (рис. 5): отображение доверительных эллипсоидов, траектории целей, подсвечивать цели, находящиеся слишком близко к границе.



(a) (b) (c)

Рис. 5: Пример отображения ”поля” при разных настройках параметров визуализации: a) все параметры отключены; b) включено отображение эллипсоидов; c) включено отображение эллипсоидов и выделение целей, находящихся близко к границе.

2.3. Реализации алгоритма оптимизации

Для реализации алгоритма оптимизации (9) была выбрана библиотека cvxpy и пакет cvxopt для Python 3. В качестве альтернативы рассматривался вариант разработки системы на matlab с использованием пакета cvx, однако в силу политики лицензирования, высокой стоимости matlab, проприетарности используемых алгоритмов и проблем с дальнейшей разработкой интегрируемой библиотеки, был выбран иной язык программирования.

Алгоритм оптимизации состоит из следующих шагов:

1. Для каждой цели и сенсора строим доверительный эллипсоид.

2. Пусть d — размерность пространства (в нашем случае $d = 2$), m — количество целей, n — количество сенсоров, $alpha, beta$ —

коэффициенты регуляризации, $ells$ — матрица, где $ells[i, j]$ — эллипс рассеивания, полученный сенсором j для цели i. Рассмотрим фрагмент реализации алгоритма (9) с помощью библиотеки cvxpy для Python 3:

```
import numpy as np
import cvxpy as cvx

#Задаем переменные
#матрица ресурсов
G = cvx.Variable((m, n))
#аппроксимирующие эллипсоиды
A = []
b = []
for i in range(m):
  A.append(cvx.Variable((d, d), PSD=True))
  b.append(cvx.Variable((n, 1)))

#Объект минимизации
objSum = 0
for i in range(m):
  objSum += -cvx.log_det(A[i])
  objSum += alpha * cvx.norm(G[i, :], 1)
for j in range(n):
  objSum += beta * cvx.norm(G[:, j], 1)
obj = cvx.Minimize(objSum)

#Ограничения
constraints = [tau >= 0]
for i in range(m):
  ellSum = np.zeros((2*d+1, 2*d+1))
  for j in range(n):
    M = np.vstack(
```

```
        np.column_stack((ells[i, j].A, ells[i, j].b,
            np.zeros((d, d)))),
        np.hstack((ells[i, j].b.reshape((1, d)),
            ells[i, j].c,
        np.zeros((d, 2*d+1))
        )
      ellSum += G[i, j] * M
    bigM = cvx.vstack(
      cvx.hstack([A[i], b[i], np.zeros(d, d)]),
      cvx.hstack([b[i].T, -1, b[i].T]),
      cvx.hstack([np.zeros((d, d)), b[i], -A[i]])
      )
    constaints.append(bigM - ellSum << 0)
#Задача птимизации
prob =  cvx.Problem(obj, constraints)
prob.solve(solver='CVXOPT', verbose=True, max_iters=1000)
```

Решатель применяет метод внутренней точки [26], имеющий полиномиальную временную сложность.

3. По полученным на шаге 2 значениям A, b, G формируем отчет для каждой цели: по A, b для строим аппроксимирующий эллипс и находим его объем, по матрице ресурсов G формируем группу слежения.

Рассмотрим пример значений матрицы ресурсов $\mathbf{G}_t$, являющейся частью результата алгоритма, решающего LMI, при параметрах $\alpha = 0.5, \beta = 0.5$:

$$G_t = \begin{pmatrix} 3.56e-01 & 6.84e-01 & 9.26e-07 & 9.27e-02 & 2.91e-07 & 1.07e-06 \\ 2.54e-01 & 7.94e-01 & 6.20e-07 & 1.31e-01 & 4.21e-07 & 8.57e-07 \\ 1.11e-01 & 7.40e-01 & 5.46e-07 & 2.65e-01 & 5.24e-01 & 7.62e-07 \\ 2.31e-07 & 2.64e-07 & 6.63e-07 & 3.29e-07 & 3.46e-07 & 9.99e-01 \end{pmatrix}$$

где значение элемента $g_t^{i,j}$ размеру вклада j-го сенсора в точность оценивания траектории i-й цели. Соответственно, "абсолютных" нулей в

матрице никогда не будет. Для решения этой проблемы введем порог значимости вклада — если значение $g_t^{i,j}$ меньше этого порога, то сенсор j не участвует в наблюдении за целью i. В работе значение этого порога бралось равным 0.001.

В ходе работы возникали проблемы при решении пакетом задачи (9) (например, вырождение некоторой вспомогательной матрицы), для которых было необходимо настроить дополнительные параметры: количество итерационных шагов для уточнения после решения системы KKT (Karush-Kuhn-Tucker conditions), максимальное число итераций, значения абсолютной и относительной точности вычислений.

3. Эксперименты

3.1. Сравнение математических моделей

Для эксперимента были выбраны $n = 6$ и $m = 4$, расположенные как на рисунке 6.

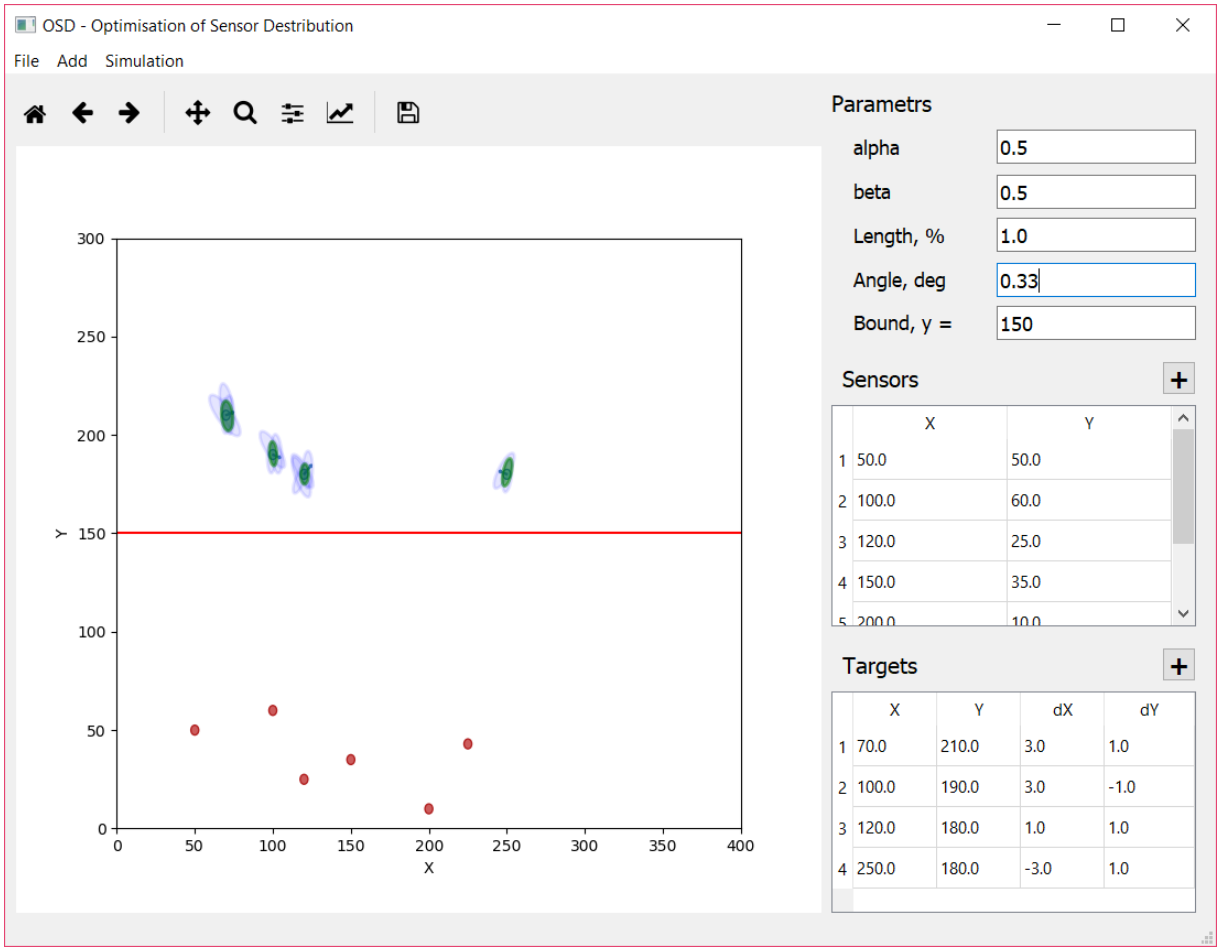
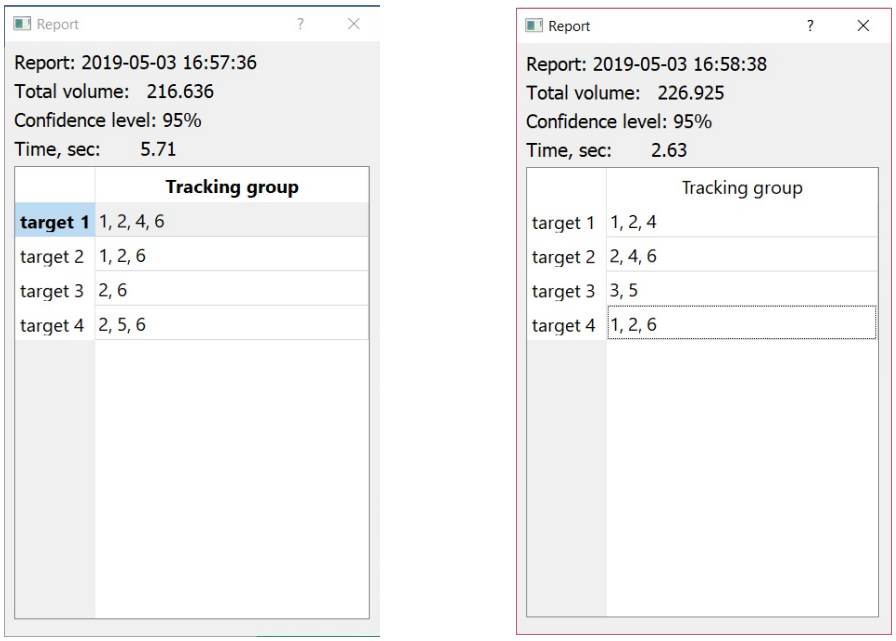


Рис. 6: Иллюстрация эксперимента

При значениях параметров $\alpha = 0.5, \beta = 0$ решение оптимизационной задачи (6) эквивалентно задаче (5). По отчету представленному на рисунке 7а, видно, что 2 и 6 сенсоры должны следить за всеми целями, в то время, как 3 сенсор не входит ни в одну из групп слежения.

Добавим ограничение на количество целей, за которым следит каждый сенсор с помощью регуляризующего коэффициента $\beta = 0.5$. В результате, распределение стало более сбалансированным: сенсоры, которые следили за всеми целями или не участвовали в процессе теперь

отсутствуют.



(a) $\alpha = 0.5, \beta = 0$ (b) $\alpha = 0.5, \beta = 0.5$

Рис. 7: Результаты моделирования для разных значениях β

3.2. Нагрузочное тестирование

На этапе нагрузочного тестирования было проведено сравнение скорости работы алгоритмов на основе LMI и полного перебора при возрастающем количестве целей и сенсоров.

Для получения результатов, приведенных в таблице 1, было проведено моделирование, состоящее из серий экспериментов. В каждой серии принимало участие определенное количество целей и сенсоров, для которых расчитывалось среднее время работы каждого из алгоритмов по следующей формуле:

$$E = \frac{1}{k} * \sum_{i=1}^{k} \mathbf{T}_i,$$

где k – количество экспериментов в серии, $\mathbf{T}_i$ — время работы алгоритма в i-м эксперименте.

Моделирование проводилось на компьютере со следующими характеристиками: процессор Intel Core i5-7300HQ с тактовой частотой 2.50GHz,

ОЗУ 16 ГБ. Значения параметров α и β равно 1.

Серия	Кол-во сенсоров	Кол-во целей	LMI, с	Полный перебор, с
1	2	3	1,603	0,077
2	5	6	3,042	>1200
3	8	8	6,932	-
4	16	16	39,109	-
5	25	25	197,20	-
6	30	30	346,70	-
7	35	35	531,99	-
8	40	40	913,73	-
9	45	45	>1200	-

Таблица 1: Результаты нагрузочного тестирования при $\alpha = 1, \beta = 1$

Результаты алгоритма показали, что полный перебор быстрее работает только при очень маленьких значениях m и n (количество целей и сенсоров), однако с их ростом LMI начинает в разы выигрывать по скорости работы у алгоритма полного перебора.

Заключение

В ходе выпускной квалификационной работы были получены следующие результаты:

1. Расширена существующая модель с помощью регуляризующего коэффициента, отвечающим за количество сенсоров, следящим за каждой целью.

2. Реализован прототип системы оптимизации распределения наблюдателей между источниками сигналов на Python3 с применением библиотеки cvxpy и пакета cvxopt для решения задач полуопределенного программирования.

3. Расширенная модель дает более сбалансированное распределение целей между сенсорами относительно существующей модели. Нагрузочное тестирование показало, что алгоритм на основе линейных матричных неравенств решает оптимизационную задачу эффективнее, чем метод полного перебора с ростом количества наблюдателей и источников сигналов.

Список литературы

[1] *Hanif A., Mansoor A.B., Imran A.S.* Deep multi-view correspondence for identity-aware multi-target tracking // In Proc. of 2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA). 2017. P. 1–8.

[2] *Thite A., Mishra A.* Optimized multi-sensor multi-target tracking algorithm for air surveillance system // In Proc. of 2016 2nd International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB). 2016. P. 637–642.

[3] *Jia B., Pham K.D., Blasch E., Shen D., Wang Z., Chen G.* Cooperative space object tracking using space-based optical sensors via consensus-based filters // IEEE Transactions on Aerospace and Electronic Systems. 2016. V. 52. No. 4. P. 1908 – 1936.

[4] *Wei B., Nener B., Liu W., Ma L.* Centralized multi-sensor multi-target tracking with labeled random finite sets // In Proc. of International Conference on Control, Automation and Information Sciences (ICCAIS). 2016. P. 82–87.

[5] *И. В. Бычков, Г. А. Опарин, А. Г. Феоктистов и др.* Сервисориентированное мультиагентное управление распределенными вычислениями // Автоматика и телемеханика. 2015. №11. С. 118-131.

[6] *Rzevski G., Skobelev P.* Managing Complexity. Wit Press, 2014. 216 p.

[7] *Wooldridge M.* An introduction to Multiagent Systems. John Wiley & Sons, 2009. 484 p.

[8] *Kalman R.E. et al.* (1960). A new approach to linear filtering and prediction problems // Journal of basic Engineering. 1960. V. 82. No. 1. P. 35–45.

[9] *Uhlmann J.K.* Algorithms for multiple-target tracking // American Scientist. 1992. V. 80. 2. P. 128–141.

[10] *Olfati-Saber R.* Distributed kalman filtering for sensor networks. In Proc. of 46th IEEE Conference on Decision and Control, 2007. P. 5492-5498.

[11] *Cattivelli F. S., Sayed A. H.* Diffusion strategies for distributed Kalman filtering ad smoothing // IEEE Transactions on automatic control. 2010. Vol. 55, no. 9. P. 2069-2084

[12] *Olfati-Saber R., Sandell N.F.* Distributed tracking in sensor networks with limited sensing range // In Proc. of American Control Conference. 2008. P. 3157–3162.

[13] *Petitti A., Di Paola D., Rizzo A., Cicirelli G.* Consensus-based distributed estimation for target tracking in heterogeneous sensor networks // In Proc. of 50th IEEE Conference on Decision and Control and European Control Conference. 2011. P. 6648–6653.

[14] Erofeeva V., Granichin O., Granichina O. Multi-Sensor Task Assignment Using Linear Matrix Inequalities in the Multiple Target Tracking Problem. In Proc. of the 18th IFAC Symposium on System Identication, 2018.

[15] *Erofeeva V., Granichin O., Leonova A.* Comparison of multi-sensor task assignment inequalities vs. brute force methods: linear matrix // IFAC-PapersOnLine, vol. 51, is. 32, 2018, pp. 648-653

[16] *Ерофеева В.А.* Оптимизация распределения целей между наблюдателями и оценивание состояний с помощью циклического подхода // Стохастическая оптимизация в информатике. 2018. Т. 52. С. 330.

[17] *Баранов А.Е., Граничин О.Н.* Оптимальный регулятор линейного объектов с ограниченной помехой // Автоматика и телемеханика. 1984. №5. С. 3946.

[18] *Якубович В.А.* Метод матричный неравеств в теории устойчивости нелинейных регулируемых систем. I. Абсолютная устойчивость вынужденных колебаний // Автоматика и телемеханика. 1964. Т. 25, №7. С. 10171029.

[19] *S. Boyd, L. El Ghaoui, E. Feron et al. Siam* Linear matrix inequalities in system and control theory. 1994. Vol. 15.

[20] *Calaore G., Polyak B. T.* Stochastic algorithms for exact and approximate feasibility of robust LMIs. IEEE Transactions on Automatic Control, 2001. Vol. 46. P. 17551759.

[21] *Polyak B., Khlebnikov M., Shcherbakov P.* An LMI approach to structured sparse feedback design in linear control systems. In Proc. 2013 European Control Conference (ECC), 2013. P. 833838.

[22] *Железнов К.О., Хлебников М.В.* Синтез обратной связи для линейной системы управления с возмущением на входах и выходах: робастная постановка, Пробл. управл., 2017, №3, 11–16

[23] *Li, H., Jing, X., Karimi, H.R.* Output-feedback-based H_∞ control for vehicle suspension systems with control delay. 2014, IEEE Transactions on Industrial Electronics 61(1),6419815, c. 436-446

[24] Python 3.5. URL: https://www.python.org/

[25] CVXPY. URL: http://www.cvxpy.org

[26] *Dantzig, George B.* Linear Programming 2: Theory and Extensions / George B. Dantzig, Mukund N. Thapa. — Springer-Verlag, 2003.