УДК 004

ПОСТРОЕНИЕ ВЕКТОРОВ В НАСТОЛЬНОМ ПРИЛОЖЕНИИ ДЛЯ АВТОМАТИЗАЦИИ РАСЧЕТОВ

Сиводедова М.В. – студентка группы Пим-241,
Буйная Е.В. – научный руководитель, доцент кафедры
прикладных информационных технологий, к.э.н.
Кузбасский государственный технический университет имени Т.Ф. Горбачева
Россия, г. Кемерово

Анализ векторных данных составляет фундаментальную основу проектно-изыскательских работ. В существующей практике для обработки пространственной информации и формирования векторных моделей традиционно применяется программный комплекс AutoCAD. Однако, при всей своей распространённости, данный инструмент не всегда обеспечивает необходимую эффективность при решении специализированных расчетных задач. В качестве демонстрации ограниченности стандартного подхода рассмотрим следующую ситуацию.

Исходными данными для анализа служат результаты натурных измерений, выполненных с применением геодезического оборудования в различные хронологические периоды. В соответствии с методикой, полный вектор смещения репера отстраивается между начальной и конечной сериями наблюдений за весь мониторинговый период. Для визуализации рассчитанных векторных характеристик требуется специализированное программное обеспечение, способное корректно интерпретировать результаты примененной методологии расчета. Представленные данные формата CSV содержат необходимую пространственную информацию, однако их обработка в универсальных САПР (таких как AutoCAD) оказывается невозможной в силу специфики использованных алгоритмов вычисления векторных параметров.

Для визуализации результатов расчета выполним построение полного вектора смещения средствами графического редактора. Векторная диаграмма создается вручную на основе вычисленных координат конечной точки вектора относительно начального положения репера. Каждое векторное представление формируется с соблюдением метрических параметров и пространственной ориентации, соответствующих результатам натурных измерений.

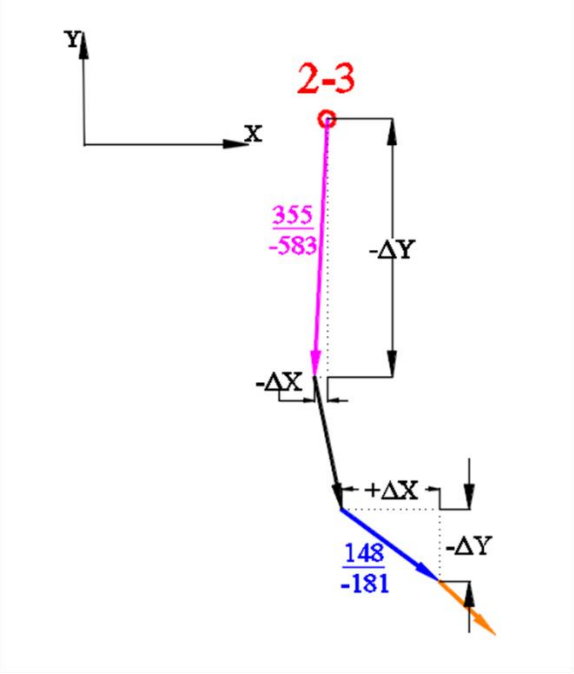

Рисунок 1 – Схема построения векторов смещения.

Для решения поставленной задачи была разработана программная реализация алгоритма построения векторов смещения, основанная на использовании объектно-ориентированного подхода и современных библиотек визуализации.

Класс Vector реализует алгоритм построения полных векторов смещения реперов в графическом интерфейсе на основе фреймворка PySide6 и библиотеки Matplotlib. При инициализации объекта создается холст размером 10×8 дюймов с интегрированной панелью навигации и элементами управления.

Основной метод drawMultiple выполняет построение векторных диаграмм для заданного набора идентификаторов реперов. Алгоритм включает следующие этапы:

1. Инициализация системы координат и очистка предыдущих построений

2. Проверка наличия достаточного количества исходных данных (не менее двух координатных точек для каждого репера)

3. Формирование массивов координат на основе результатов натурных измерений

4. Визуализация полных векторов смещения между начальной и конечной точками

```
colors = ['b', 'g', 'y', 'c', 'm', 'k']
# Создаем фигуру и оси
ax.text(coordinates[0][0], coordinates[0][1], f'({coordinates[0][0]:.1f}, {coordinates[0][1]:.1f})',
        fontsize=8,
        verticalalignment='bottom',
        horizontalalignment='right')
# Рисуем векторы и отмечаем точки
for i in range(len(coordinates) - 1):
    x, y, reper_name = coordinates[i]  # Начальная точка вектора
    x_end, y_end, reper_name = coordinates[i + 1]  # Конечная точка вектора
    dx, dy = x_end - x, y_end - y  # Смещение
    color = colors[i % len(colors)]  # Цвет из списка (по кругу)
    # Рисуем вектор
    ax.quiver(x, y, dx, dy, angles='xy', scale_units='xy', scale=1, color=color,
              label=f'Вектор {i + 1} ' + str(reper_name))
    # Отмечаем начало и конец вектора на графике (точки)
    ax.scatter([x, x_end], [y, y_end], color='black', s=20)
    # Подписываем координаты около точек
    ax.text(x_end, y_end, f'({x_end:.1f}, {y_end:.1f})', fontsize=8, verticalalignment='bottom',horizontalalignment='left')
    # Проекции точек на оси X и Y
    ax.axhline(y, color='gray', linestyle='dashed',
               linewidth=0.7)  # Горизонтальная линия для начальной точки
    ax.axvline(x, color='gray', linestyle='dashed', linewidth=0.7) # Вертикальная линия для начальной точки
    ax.axhline(y_end, color='gray', linestyle='dotted',
               linewidth=0.7)  # Горизонтальная линия для конечной точки
    ax.axvline(x_end, color='gray', linestyle='dotted',
               linewidth=0.7)  # Вертикальная линия для конечной точки
```

Рисунок 2 – Фрагмент программного кода

Особенностью интерфейса является реализация диалогового окна настроек, позволяющего оперативно изменять параметры визуализации в процессе анализа.

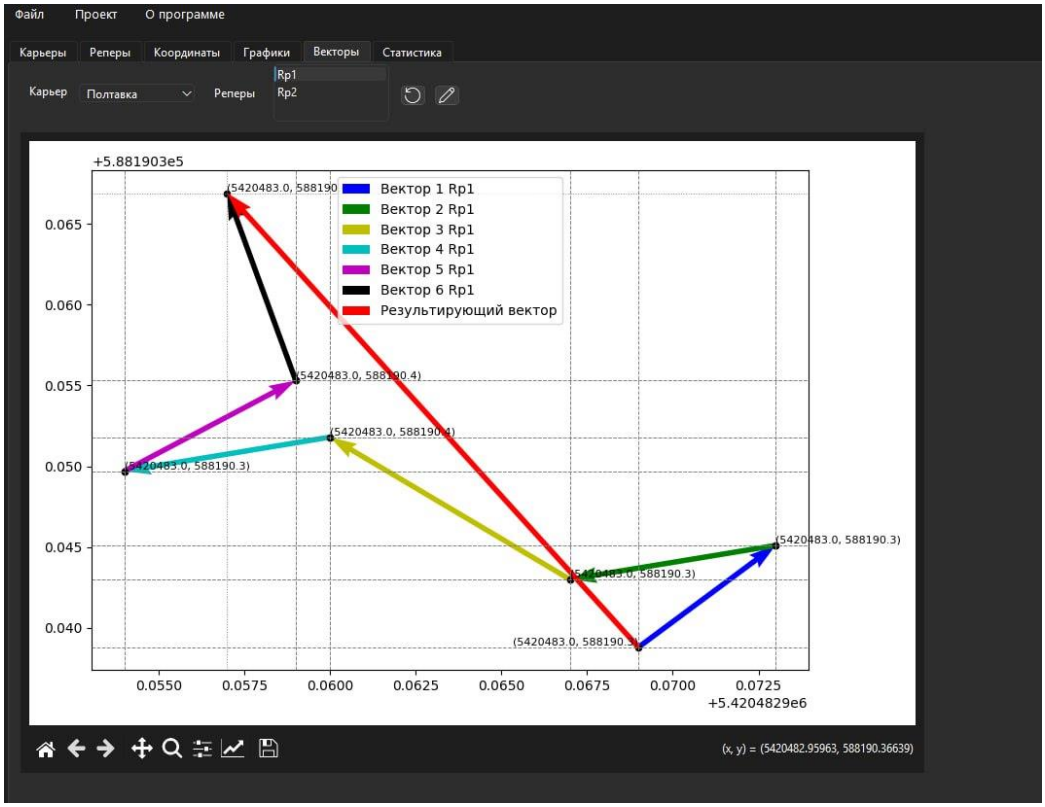

Рисунок 3 – Результат построения графика в программе

Проведенное исследование подтвердило эффективность разработанного программного решения в сравнении с традиционными методами визуализации. Реализованный подход обеспечивает комплексную автоматизацию процесса построения данных и расширенный функционал, включая систему

классификации параметров с цветовым кодированием для оперативной оценки критических состояний.

Интеграция библиотеки визуализации с графическим фреймворком создает основу для интерактивной работы, предоставляя пользователям инструменты масштабирования, панорамирования и динамической корректировки параметров отображения. Значимым преимуществом решения является гибкий механизм настройки критических значений через специализированный интерфейс, позволяющий адаптировать систему под конкретные задачи мониторинга.

Программная реализация исключает необходимость ресурсоемкого ручного построения диаграмм и гарантирует воспроизводимость результатов. Разработанное решение демонстрирует универсальность применения и может быть адаптировано для обработки различных типов пространственных данных в системах технического мониторинга.

Список литературы:

1. Маккинни, У. Python и анализ данных / У. Маккинни; перевод с английского А. А. Слинкина. — 2-ое изд., испр, и доп. — Москва: ДМК Пресс, 2020. — 540 с. — ISBN 978-5-97060-590-5. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/131721 (дата обращения: 18.11.2025). — Режим доступа: для авториз. пользователей.

2. Жорняк, А. Г. Обработка больших массивов данных и визуализация результатов научных исследований с использованием библиотек NumPy и Matplotlib языка программирования Python: учебное пособие / А. Г. Жорняк, Т. А. Морозова. — Москва: МАИ, 2025. — 92 с. — ISBN 978-5-00246-342-8. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/506712 (дата обращения: 23.11.2025). — Режим доступа: для авториз. пользователей.

3. Титов, А. Н. Визуализация данных в Python. Работа с библиотекой Matplotlib: учебно-методическое пособие / А. Н. Титов, Р. Ф. Тазиева. — Казань: КНИТУ, 2022. — 92 с. — ISBN 978-5-7882-3176-1. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/331025 (дата обращения: 21.11.2025). — Режим доступа: для авториз. пользователей.

4. Леонова, Н. И. Инженерная и компьютерная графика: учебное пособие / Н. И. Леонова, Ю. Г. Параскевопуло, В. В. Сальникова. — Санкт-Петербург: ПГУПС, 2024. — 54 с. — ISBN 978-5-7641-2000-3. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/505204 (дата обращения: 23.11.2025). — Режим доступа: для авториз. пользователей.