МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

ПРИЛОЖЕНИЕ ДЛЯ РАСЧЕТА НЕГАБАРИТНОСТИ ГРУЗОВ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Целикиной Милены Андреевны

Научный руководитель
к. ф.-м. н., доцент _____ С. В. Миронов

Заведующий кафедрой
к. ф.-м. н., доцент _____ А. С. Иванов

Саратов 2020

ВВЕДЕНИЕ

Данная дипломная работа является частью проекта для компании «Российские железные дороги» в рамках конкурса по оптимизации перевозок. Проект составлен научными сотрудниками и преподавателями географического факультета СГУ в сотрудничестве с представителем филиала компании РЖД в Поволжье.

Данная работа направлена на исследование особенностей перевозки негабаритных грузов и частичная автоматизация расчетов для пропуска таких грузов по маршруту. Для реализации приложения был выбран язык Python, поскольку он является достаточно мощным и многофункциональным языком, который поддерживает и объектно-ориентированный, и функциональный стили программирования. Кроме того, язык Python имеет большой набор библиотек. Для реализации интерфейса был выбран фреймворк PyQt, а для вывода графики дополнительно подключен OpenGL.

Составленный проект является актуальным, поскольку сейчас большинство расчетов при перевозке негабаритных и тяжеловесных грузов производится вручную или с помощью различных уже составленных таблиц. Однако этот способ является довольно трудозатратным. Поэтому был предложен данный проект, который позволяет автоматизировать около 80% работы.

Помимо актуальности стоит рассмотреть цель изучения данной темы. Главная цель исследования заключается в изучении предметной области и автоматизации части расчётов, необходимых при перевозке негабаритных грузов.

Цель настоящей дипломной работы порождает ряд задач, которые позволяют достигнуть поставленной цели:

- научиться работать с заказчиком;
- исследовать литературу, связанную с предметной областью;
- изучить правила перевозки негабаритных грузов;
- разобрать формулы для расчётов, необходимых при перевозке негабаритных грузов;
- разработать план приложения для автоматизации необходимых расчётов;
- реализовать приложение на языке Python с использованием фреймворка PyQt5.

Объектом данного исследования является предметная область, т.е. пра-

вила перевозки негабаритных грузов и изучение возможности автоматизации расчётов для негабаритных грузов.

Предмет исследования — язык программирования Python и его возможности, в том числе фреймворк PyQt5.

Дипломная работа состоит из двух глав:

1. Теоретические аспекты.
2. Практическая часть.

1 Основные теоретические выкладки

Конечной целью данного проекта является снижение трудозатрат на формирование оптимального маршрута пропуска негабаритных и тяжеловесных грузов на 80%.

При пропуске негабаритных и тяжеловесных грузов владельцы железных дорог обязаны устанавливать условия пропуска по ним подвижного состава, т.е. представить условия оптимального маршрута движения подвижного состава с учетом негабаритных мест (мест потенциальной опасности столкновения с подвижным составом), технических особенностей ж.д. пути (наличие кривых малого радиуса), допустимых нагрузок искусственных сооружений.

Кроме того, если груз является сверхнегабаритным, то в этом случае недостаточно даже расчётов. Тогда строится модель груза из дерева и эта модель пускается по одному из предположительно оптимальных путей. Если модель заденет какой-либо объект на пути, то принимается решение либо о смене пути, либо о переносе этого объекта (например, столба) дальше от путей. Так на данный момент решается проблема перевозки сверхнегабаритных грузов.

Задача автоматизации перевозки негабаритных грузов делится на 2 этапа:

1. Выбор вагона-транспортера и расчёт всех характеристик груза;
2. На основе данных о грузе построение оптимального маршрута.

В рамках данной работы решается первая задачи автоматизации.

Все грузы, которые выходят за пределы габарита погрузки, являются негабаритными грузами. Для них определяется индекс негабаритности.

Индекс негабаритности представляет собой строку, состоящую из 5 знаков вида «Нхххх», где:

— Н — первая буква слова «негабаритность»;
— 1й знак х — нижняя негабаритность;
— 2й знак х — боковая негабаритность;
— 3й знак х — верхняя негабаритность;
— 4й знак х — вертикальная сверхнегабаритность.

1.1 Примитивы и критические точки

При погрузке грузов на вагон-транспортер работники РЖД пытаются это сделать как можно более эффективно и оптимально, поскольку чем мень-

ше негабаритность груза, тем дешевле будет его транспортировка. Все грузы пытаются подогнать под некоторый примитив, который будет в основном (или даже полностью) умещаться в габарите погрузки. Однако в большинстве случае груз невозможно полностью подогнать под примитив. В этом случае необходимо либо увеличить этот примитив, что приведёт к тому, что заявленная площадь поперечного сечения груза станет больше (то есть увеличится стоимость транспортировки), либо какая-то часть останется за пределами примитива. Тогда крайние точки примитива и эти выступающие части будут образовывать критические точки груза.

Критические точки — наиболее удаленные от продольной оси пути и от уровня головок рельсов точки выступающих узлов и деталей железнодорожного подвижного состава [1].

Для критических точек (в том числе для крайних точек примитива) необходимо отдельно рассчитывать выносы. Именно они и будут определять степень негабаритности груза. Существует 4 типа примитива, под которые подгоняют груз: эллипс, прямоугольник, треугольник и прямоугольник, совмещенный с трапецией. При отправке негабаритного груза оператор указывает, под какой примитив подогнан груз. Для того, чтобы задать примитив, оператор должен ввести определенные характеристики этой фигуры.

1.2 Барицентрические координаты

Для решения нескольких задач в рамках проекта необходимо понятие барицентрических координат.

Допустим есть некоторая внутренняя точка K треугольника ABC. Тогда можно подобрать такие положительные числа α, β, γ, для которых точка K будет центром масс. Очевидно, что числа α, β, γ не будут однозначны [2]. Если есть некоторое положительное число m, то массы $\alpha * m, \beta * m, \gamma * m$ обладают тем же свойством, что и массы α, β, γ. Т. е. та же точка K является центром масс $(\alpha * m)A, (\beta * m)B, (\gamma * m)C$. Исходя из этого, можно сделать сумму масс равной единице. Барицентрические координаты α, β, γ, для которых выполняется условие $\alpha + \beta + \gamma = 1$, называются абсолютными барицентрическими координатами (АБК). АБК определяются однозначно [3].

Такие числа α, β, γ называются *барицентрическими координатами* [4].

Барицентрические координаты хоть и описаны на основе такого понятия как центр масс, но применяются для решения ряда задач, не связанных с этим

понятием. Так, барицентрические координаты используются, например, в компьютерной графике для определения местонахождения точки: находится она внутри фигуры или нет. Причем расчеты можно производить как в двумерном, так и в трехмерном виде.

В данной задаче барицентрические координаты используются для определения местонахождения точки. В качестве примитива берется треугольник. Остальные многоугольники делятся на конечное число треугольников, если нет способа проще определить местоположение точки.

1.3 Математические выкладки

При транспортировке груза любого вида (даже укладывающегося в габарит) нужно учитывать параметры железнодорожного пути. Путь может иметь некоторые характеристики:

- радиус кривой и направления закругления, в этом случае вынос критических точек может увеличиться, а значит изменится и индекс негабаритности груза на данном участке пути;
- продольный уклон (поскольку очень незначителен, им часто пренебрегают при расчетах), за исключением парка горочных путей, через которые запрещен пропуск негабаритных грузов;
- возвышение рельса и его сторона (поперечный уклон), который также может повлиять на индекс негабаритности.

1.4 Схема груза и визуальное отображение индекса негабаритности

Предполагается, что в программе должна присутствовать схема для визуализации индекса негабаритности. Т.е. оператор должен видеть, где находится критическая точка и почему индекс негабаритности рассчитан именно таким. О грузе мы знаем только примитив, под который он был подогнан при упаковке, и координаты всех критических точек. Других предположений о поперечном сечении груза мы делать не можем, поскольку данных для этого недостаточно, т. к. оператор в ручном или полуавтоматическом вводе не может указать тысячи точек для точного определения сечения. Поэтому в программе отрисовывается только сам примитив и все критические точки груза без соединений. Эта реализация позволяет наглядно показать, где груз выходит за габарит, и скорректировать упаковку груза, в случае, если индекс негабаритности не устраивает.

2 Структура проекта

Программный код всего приложения для расчетов негабаритности грузов разделен на несколько пакетов модулей:

— `Classes` — классы для элементов предметной области: груз, вагон, путь;
— `DataBase` — прослойка для работы с базой данных;
— `image` — изображения для вагонов;
— `Interface` — пакет с интерфейсом;
— `Logic_work` содержит дополнительные файлы для реализации логики приложения;
— `MainFile.py` — главный файл приложения.

2.1 Приложение с точки зрения разработчика

2.1.1 Классы для элементов предметной области

Элементами предметной области являются груз, вагон и путь. Все параметры этих объектов описаны в разных файлах и реализованы с помощью разных классов. Рассмотрим подробнее каждый файл:

1. В файле `Way` реализован одноименный класс, который описывает все параметры определенного участка пути, на котором необходимо произвести расчеты. Он хранит параметры участка пути, которые понадобятся при расчетах негабаритности груза в других методах и функциях.
2. В файле `TrainCar.py` описан одноименный класс для описания параметров вагона-транспортера. Он содержит данные о вагоне, например, название модели вагона, его тип или базу.
3. В файле `Load.py` описаны самые важные классы и методы, которые отвечают за основную массу расчетов: `CriticalPoint` и `Load`.

   Класс `CriticalPoint` описывает все параметры критической точки груза: название точки, ее координаты, расстояния от сечений и выносы. Класс `Load` описывает все параметры груза: его вес, длину и массив критических точек.

   Кроме того, в классе `Load` описан ряд важных методов:

   а) `calculate_take_pu(self, way)` вычисляет вынос все критических точек при поперечном уклоне.

   б) `calculate_pu(self, x_p, y_p, way)` вычисляет вынос одной критической точки груза при поперечном уклоне.

в) `__find_middle_end_points(self, primitive, base)` — поиск средней и крайних точек груза. Эти точки создают наибольший вынос при симметричном грузе, поэтому в массив критических точек они также добавляются.

г) `find_all_points(self, primitive, base)` — поиск всех потенциальных критических точек примитива груза.

д) `calc_takeaway(self, train, way)` — расчет выноса для каждой критической точки при наличии кривой некоторого радиуса для вагонов с жесткой сцепкой.

е) `calc_axis_load(self, train)` — расчет осевой нагрузки от одной колесной пары на рельс.

ж) `calc_all_weight(self, train)` — расчет массы всей конструкции.

з) `calc_takeaway_tsch(self, train, way, state)` — расчет выноса при наличии радиуса для транспортеров сочлененного и сцепного типа.

и) `__take_500k(*args)`, `__take_400(*args)`, `__take_300m(*args)`, `__take_480(*args)` предназначены для расчета выноса у конкретной модели транспортера, т. к. транспортеры с подвижной сцепкой имеют специфичные вычисления, формулы которых были представлены в пункте 1.4.1.

2.1.2 Взаимодействие с базой данных

В папке `DataBase` хранится файл `DB.py`, который содержит логику взаимодействия с базой данных (скрипт для создания таблиц базы данных и для добавления информации в них находится в файле `CreateDBTrainCar.sql`). База состоит из двух таблиц: тип и модель вагона. Для взаимодействия с базой с помощью языка Python были использованы библиотеки `sqlalchemy` для подключения к базе и `pandas` для обработки полученных данных.

2.1.3 Классы примитивов

Примитивы описаны в файле `Primitive.py`. Каждый примитив описан в отдельном классе:

— `Rectangle` — прямоугольник;

— `Ellipse` — эллипс;

— `Trapz` — прямоугольник, совмещенный с трапецией;

— `Triangle` — треугольник.

Примитив задан в виде координат вершин (кроме эллипса). В классах примитивов существуют аналогичные методы, которые реализованы для того, чтобы интерфейс работы с ними был аналогичен. К таким методам относятся:

— `check_in_primitive(self, coord)` — метод для проверки на то, что точка с координатами $coord$ находится внутри примитива.

— Статические методы вида `build_xxx()`, где xxx — название примитива. Данные методы создают примитивы по переданным параметрам. Например, для создания прямоугольника передаются длины сторон, а метод определяет координаты вершин. Это необходимо для поиска местонахождения точки с помощью барицентрических координат.

— Статические методы вида `check_xxx()`, где xxx — название примитива. Эти методы проверяют, существует ли примитив с заданными параметрами. Например, треугольник не может существовать, если сумма каких-либо двух сторон меньше либо равна третьей стороне. В таком случае примитив не будет создан и вернется $None$ и др.

Кроме того, в файле `NeedsFunc.py` описаны некоторые функции, которые необходимы для расчетов во многих файлах, но которые нельзя отнести к какому-либо классу. Например, метод для перевода строки, введенной через интерфейс, в double или валидаторы вещественных чисел.

2.1.4 Классы степени негабаритности

В файле `Indexes.py` для описания каждой зоны негабаритности было создано несколько классов:

— `IndexNotGabDown` — зона нижней негабаритности;

— `IndexNotGabMiddle` — зона боковой негабаритности. Является наследником класса `IndexNotGabDown`;

— `IndexNotGabUp` — зона верхней негабаритности.

Каждый из описанных выше классов содержит конструктор и метод `isInto(self, x, y)`, который определяет для каждого типа, входит точка в данную зону или нет. Для простых типов (в виде прямоугольников) просто определяется вхождение в виде границ примитива. Для сложных используются барицентрические координаты.

Главный класс для определения индекса негабаритности груза — `DetermineIndex`. Он содержит в себе массивы с объектами классов зон негаба-

ритности. Каждый такой объект определяет конкретный тип негабаритности в конкретной зоне. Кроме того, в классе `DetermineIndex` присутствуют методы, необходимые для отрисовки схемы, визуализирующей индекс и груз. К таким методам относятся: `get_coord_down(self)`, `get_coord_up(self)`, `get_coord_middle(self)`, `get_coord_gab()`.

2.1.5 Интерфейс приложения

Весь интерфейс приложения описан в отдельном модуле `Interface`, который содержит следующие файлы:

1. `AdditParam.py` — файл с реализацией окна для ввода данных о грузе;
2. `ChoosePrimitive.py` — файл с реализацией окна для выбора примитива груза;
3. `rzhd.py` — файл с реализацией интерфейса главного окна;
4. `Styles.py` — файл со стилями `QSS`.

Главным файлом является `MainFile.py`, с которого и начинается запуск приложения.

Кроме того, в файле `rzhd.py` (и всех файлах, отвечающих за интерфейс) присутствуют конструкции, позволяющие адекватно отображать интерфейс окна при использовании любого расширения экрана, а также при изменении размеров самого окна. Для этого были экспериментально вычислены коэффициенты масштабирования объектов на форме.

2.1.6 Схема для визуализации груза

Для визуализации схемы груза используется 2 инструмента: встроенная в PyQt5 сцена и OpenGL.

Начнем с первого способа. Его логика описана в файле `SceneSchema.py`:

1. `add_polygon(self, gab, pen)` — добавление на сцену многоугольника.
2. `add_gab(self)` — отрисовка габарита погрузки.
3. `add_middle(self, indexes)`, `add_up(self, indexes)`, `add_down(self, indexes)` — отрисовка верхней, боковой и нижней негабаритностей.
4. `add_points(self, list_points)` — добавление критических точек на схему.
5. `add_primitive(self, primitive, x_delta=0)` — добавление примитива на схему.

6. `rotate_primitive(self, prim, angle, dir)` — поворот примитива на угол angle в направлении dir.
7. `choose_and_add_primitive(self, primitive, list_points)` — выбор и отрисовка примитива.
8. `rad_paint(self, primitive, list_points)` — метод для определения точек примитива, которые сдвигают данный примитив. Необходим, поскольку при креплении груза на ТСЧ существует вынос только вовнутрь кривой.

Рассмотрим методы из файла `openGL.py`, позволяющие рисовать то же изображение на сцене `QOpenGLWidget`:

1. `initializeGL(self)` — инициализация. Запускается при создании объекта класса `OpenGL`.
2. `paintGL(self)` — событие перерисовки графических элементов.
3. `resizeGL(self, width, height)` — реакция на событие изменения размеров окна.
4. `paintGab(self)` — создание списка для габаритной области.
5. `paintUp(self, indexes)`, `paintDown(self, indexes)`, `paintMiddle(self, indexes)` — создание списков точек для всех видов негабаритности.
6. `makeNotGab(self)` — создание всех элементов сцены.
7. `rad_paint(self)` — метод для определения точек примитива, которые сдвигают данный примитив. Необходим, поскольку при креплении груза на ТСЧ существует вынос только вовнутрь кривой.
8. `draw(self, primitive)` — отрисовка элементов.
9. `paintPoints(self)` — ввод критических точек.
10. `paintPrimitives(self, trans=None, angle=None, plus=0)` — ввод точек для примитивов.
11. `fillEllipse(self, xCenter, yCenter, rx, ry, plus=0, pointCount = 360)` — создание эллипса через треугольники.

2.2 Взаимодействие пользователя с готовым приложением

Взаимодействие оператора с приложением состоит из 3 этапов:

1. Ввод данных о вагоне и грузе.
2. Ввод критических точек.
3. Получение результатов работы.

Рассмотрим этапы более подробно.

2.2.1 Ввод данных о вагоне и грузе

При открытии приложения перед операторов появляется начальное окно, на котором видна только левая панель для ввода параметров вагона и груза. Лучше всего при вводе данных идти по параметрам сверху вниз, но это не обязательно. Рассмотрим параметры на панели подробнее.

Сначала пользователю предлагается выбрать тип и модель вагона-транспортера. Эти данные представлены в виде выпадающих списков. Типы и модели берутся из базы данных. Существует 5 типов вагонов-транспортеров. При выборе типа обновляется список моделей. Выводятся только те, которые относятся к данному типу вагонов-транспортеров.

Далее идут две кнопки для ввода параметров груза и примитива. Рассмотрим сначала параметры груза. При нажатии на кнопку «Параметры груза» выводится окно для ввода длины и массы груза. Эти данные являются обязательными для последующих расчетов. В полях ввода параметров возможно ввести только вещественные положительные числа до 100000. Для того, чтобы сохранить параметры груза, нужно заполнить оба поля. В противном случае при попытке сохранить данные будет выведено сообщение об ошибке.

Затем идет кнопка «Выбор примитива груза». При нажатии на данную кнопку открывается окно для ввода параметров примитива груза. Окно представлено в виде выпадающего списка с возможными примитивами и формы для ввода параметров. Параметров у примитивов различны. Например, у прямоугольника необходимо ввести длины сторон, а у эллипса — полуосей. Треугольник можно ввести тремя способами. Для этого предусмотрен отдельный выпадающий список с возможными способами ввода. Если примитив с введенными параметрами не существует, то выводится соответствующее сообщение об ошибке. Также присутствуют ошибки, аналогичные вводу параметров груза.

Далее пользователю предлагается выбрать один из способов ввода критических груза. После выбора расположена кнопка «Подтвердить», позволяющая сохранить введенные данные и перейти к вводу критических точек. Если на данном этапе не были введены параметры груза, примитива или модель вагона, то появилось бы сообщение об ошибке.

2.2.2 Ввод критических точек

Функционал данного приложения предполагает 3 способа ввода информации о критических точках груза:

1. Ручной способ.
2. Полуавтоматический способ.
3. Автоматический способ.

На предыдущем шаге пользователь выбирает один из этих способов и переходит к форме ввода точек.

При выборе ручного способа ввода на форме появляется таблица, в которую необходимо ввести данные о критических точках. Для добавления и удаления точки существуют кнопки «Добавить точку» и «Удалить точку». При вводе параметров критической точки в ячейках таблицы можно указывать только вещественные числа (кроме названия точки). В противном случае при попытке перейти на следующий этап программа выведет сообщение об ошибке. После ввода всех критических точек можно переходить к следующей форме по нажатию кнопки «Перейти к расчетам».

При выборе автоматического способа ввода на форме также появляется таблица и кнопки, а также открывается файловый диалог, который предлагает выбрать файл с результатами наземного лазерного сканирования. При выборе файла происходит проверка на соответствие формату. Если файл не соответствует формату, выводится сообщение об ошибке. При успешной проверке, файл считывается и обрабатывается. Обработка состоит в отсечении точек, не являющихся критическими. Кроме того, можно повторно выбрать файл, при этом предыдущие данные будут очищены. Аналогично ручному способу ввода по нажатию кнопки «Перейти к расчетам» можно перейти на следующий этап.

2.2.3 Получение результатов работы

На 3 шаге открывается форма для расчетов. Сразу оператору представляются все вычисленные значения для прямого пути: вес конструкции, осевая нагрузка и индекс негабаритности. Также при нажатии на кнопку «Использовать сцену» или «Использовать OpenGL» можно увидеть графическое представление всех введенных данных: примитива груза и критических точек.

При переключении на следующую вкладку «Поворот пути» можно рассчитать параметры груза на участке кривой некоторого радиуса. Данная фор-

ма выглядит аналогично предыдущей, однако здесь появились поля для ввода радиуса кривой поворота. Для расчета параметров груза на кривом пути необходимо ввести величину радиуса поворота и нажать на кнопку «Рассчитать». При отсутствии значения радиуса выведется соответствующая ошибка. В поле радиуса также можно вводить только вещественные положительные числа.

Подсчитанные данные — индекс негабаритности на кривой и координаты всех точек с учетом выноса — будут выведены в поле и таблицу соответственно. Здесь, как и в предыдущем случае, можно визуально увидеть то, что получилось. По нажатию на кнопку «Использовать сцену» появится визуальное представление груза на встроенной сцене с учетом кривой. Аналогичным будет изображение, построенное с помощью OpenGL.

Наконец, перейдем к последней вкладке данной формы — «Поперечный уклон». Здесь также есть поля, но они используются для ввода поперечного уклона. Поперечный уклон в железнодорожной отрасли принято исчислять в промилле. Максимальный поперечный уклон равен 40 ‰.

Для расчета параметров груза при присутствии поперечного уклона необходимо ввести величину данного уклона и нажать на кнопку «Рассчитать». Аналогично полю ввода радиуса в поле ввода уклона можно вводить только вещественные числа, но до 40 ‰. При отсутствии значения поперечного уклона выведется соответствующая ошибка.

Данные будут выведены в те же поля, что и в предыдущей вкладке, но с учетом поперечного уклона. Мы также можем увидеть визуальное представление этих данных.

Кроме того, пользователь может возвращаться назад для изменения введенных ранее данных о грузе, вагоне или критических точках.

ЗАКЛЮЧЕНИЕ

В данной дипломной работе была достигнута главная цель — реализовано приложение для частичной автоматизации расчетов негабаритности груза. Для достижения цели были решены следующие задачи:

- изучены аспекты работы с заказчиком;
- исследована литература, связанная с предметной областью;
- изучены правила перевозки негабаритных грузов;
- разобраны формулы для расчётов, необходимых при перевозке негабаритных грузов;
- разработан план приложения для автоматизации необходимых расчётов;
- реализовано приложение на языке Python с использованием фреймворка PyQt5.

Таким образом, этот проект, действительно, является актуальным, поскольку сейчас большинство расчетов при перевозке негабаритных грузов производится вручную или с помощью различных уже составленных таблиц. Разработанное приложение позволяет автоматизировать часть этих расчетов.

В результате решения поставленной задачи были исследованы особенности перевозки негабаритных грузов и возможности частичной автоматизации расчетов для пропуска таких грузов по маршруту. Исследованы возможности языка Python и его инструментов: фреймворка PyQt5 и библиотеки PyOpenGL.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 ГОСТ 9238-2013 Габариты железнодорожного подвижного состава и приближения строений (с поправками). — Москва: Стандартинформ, 2014. — С. 173.

2 *Макаров, Е. М.* Линейные и аффинные пространства в компьютерной геометрии. Учебно-методическое пособие / Е. М. Макаров. — Нижний Новгород: Нижегородский госуниверситет, 2019. — С. 36.

3 *Kimberling, C.* Encyclopedia of Triangle Centers [Электронный ресурс] / C. Kimberling. — URL: https://faculty.evansville.edu/ck6/encyclopedia/ (Дата обращения 04.05.2020). загл. с экр. яз. англ.

4 *Балк, М. Б.* Геометрия масс / М. Б. Балк, В. Г. Болтянский. — Москва: Наука, 1987. — С. 160.