МИНОБРНАУКИ РОССИИ
ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ОТЧЕТ
ПО УЧЕБНОЙ ОЗНАКОМИТЕЛЬНОЙ ПРАКТИКЕ

Студент
гр. БИН-22-2 _____ Е.К. Гордиенко

Руководитель
старший преподаватель каф. ИТС _____ Е.Г. Лаврушина

Владивосток 2024

МИНОБРНАУКИ РОССИИ
ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ЗАДАНИЕ

на учебную ознакомительную практику

Студенту гр. БИН-22-2 Гордиенко Елена Константиновна

1 Тема работы: Разработка программы

2 Срок сдачи работы: 13.07.2024.

3 Содержание задания

3.1 Решение практических задач

Задача 1.

Используя произвольный язык программирования и среду разработки создайте программу, которая отображает на экране монитора график кривой или поверхности (в соответствии с вариантом задания № 6, Гипоциклоида) в декартовой и полярной системах координат с центром в центре экрана монитора (окна или иной прямоугольной области экрана). При изменении размеров окна, график и все его атрибуты (координатная сетка, метки на шкале, подписи и т.д.) должны автоматически масштабироваться.

Параметры уравнения кривой или поверхности должны вводиться в специально отведённые ячейки экранной формы программы.

Задача 2.

Используя результаты предыдущего задания создайте анимацию примитива, движущегося по траектории построенной кривой:

- для чётных вариантов в качестве примитива используется закрашенная окружность красного цвета радиуса $r > 2$;

- для нечётных вариантов в качестве примитива используется закрашенный квадрат синего цвета со стороной $a > 2$.

3.2 Оформление и защита отчета

Отчет по учебной исследовательской практике должен содержать: титульный лист; индивидуальное задание; содержание; цель и задачи; описание выполненных заданий (в соответствии с методическим руководством и требованиями преподавателя); выводы и

предложения; список использованных источников; графический материал (схемы, графики, технологические карты).

Календарный план-график работ прилагается к отчету отдельным листом.

Требования к оформлению отчетов.

Отчёт предоставляется в печатном виде с выполнением требований нормоконтроля и состоит из следующих разделов:

Введение. Во введении обосновывается цель и задачи прохождения практики.

В разделе 1 выполняется краткий обзор литературы по теме индивидуального задания, подбирается необходимый математический инструментарий (аппарат), производится его анализ и разрабатывается алгоритм решения задачи.

В разделе 2 осуществляется обоснование выбора среды реализации разработанного алгоритма.

В разделе 3 описывается процесс кодирования в выбранной среде и языке программирования и основные элементы управления создаваемым приложением.

В разделе 4 описывается процесс тестированию и отладки программного кода.

Раздел 5 (рекомендуемый) должен содержать краткое иллюстрированное руководство пользователя.

Заключение. В заключении обобщается изложенный в отчёте материал, делаются выводы.

Объем отчёта составляет 20-25 страниц.

Отчёт по практике оформляется в соответствии с «Требованиями к оформлению текстовой части выпускных квалификационных работ, курсовых работ (проектов), рефератов, контрольных работ, отчётов по практикам, лабораторным работам (СК-СТО-ТР04-1.005-2020)». Для оформления графического материала блок-схем, алгоритмов использовать ГОСТ 19.701-90 который распространяется на условные обозначения (символы) в схемах алгоритмов, программ, данных и систем и устанавливает правила выполнения схем, применяемых для отображения различных видов задач обработки данных и средств их решения.

Руководитель
старший преподаватель каф. ИТС _____ Е.Г. Лаврушина

Задание получил: _____ Е.К. Гордиенко

Содержание

Введение

В ходе данной учебной практики были успешно достигнуты поставленные цели и выполнены задачи, направленные на разработку интерактивной программы для визуализации Гипоциклоиды в декартовой и полярной системах координат, а также анимации движения примитива по её траектории. Программа включает в себя не только пользовательский интерфейс с возможностью ввода параметров уравнения кривой, но и функционал автоматического масштабирования графика и его элементов при изменении размеров окна.

Разработка плавной анимации движения красного шара по траектории Гипоциклоиды демонстрирует умение использовать результаты визуализации для создания интерактивных и наглядных примеров математических объектов. Этот процесс не только укрепил понимание особенностей Гипоциклоиды, но и позволил освоить современные методы программирования для визуализации и анимации в Python.

Итоговая программа представляет собой полезный инструмент для обучения и исследования математических кривых, обеспечивая пользователям возможность интерактивного моделирования и визуализации движения, что способствует более глубокому и наглядному усвоению материала.

1. Гипоциклоида

1.1 История

История гипоциклоиды уходит корнями в древние времена, связываясь с изучением геометрии и движения тел в пространстве. Гипоциклоида — это геометрическая фигура, которая возникает при движении точки на окружности, которая в свою очередь катится без скольжения по внутренней стороне другой фиксированной окружности. Это движение создает интересные и красивые формы, которые находят применение не только в математике, но и в других областях, таких как физика, инженерное дело, и даже в изобразительном искусстве.

Первые упоминания о гипоциклоиде ведут нас к трудам древнегреческих математиков, таких как Евклид и Архимед. 17 век ознаменовался возрождением интереса к этой кривой. Одним из первых, кто исследовал гипоциклоиды, был голландский математик Христиан Гюйгенс в 1673 году. Он заметил, что при качении одной окружности внутри другой, точка на внутренней окружности описывает гипоциклоиду. Математики того времени, такие как Блез Паскаль и Христиан Гюйгенс, исследовали ее геометрические свойства и применяли их для решения задач механики.

1.2 Понятие и математический инструментарий

Гипоциклоида – это плоская кривая, которую рисует точка на окружности, катящейся по внутренней стороне другой, более крупной окружности. В свою очередь гипоциклоида является частным случаем гипотрохоиды - линии, которую вычерчивает точка, жестко связанная с окружностью, катящейся без проскальзывания по неподвижной окружности. Существует несколько способов записать уравнение гипоциклоиды. Параметрические уравнения выглядят следующим образом:

$$x = (R - r)\cos(t) + r\cos((R - r)t / r) \quad (1)$$
$$y = (R - r)\sin(t) + r\sin((R - r)t / r),$$

где R определяется как радиус большей окружности, а как r радиус меньшей окружности. Параметр t – угол, пройденный точкой на меньшей окружности. В зависимости от отношения радиусов окружностей получаются различные по форме кривые. Увидеть Гипоциклоиду с отношением длин большой и маленькой окружности равным 3 в декартовой системе координат можно на рисунке 1.

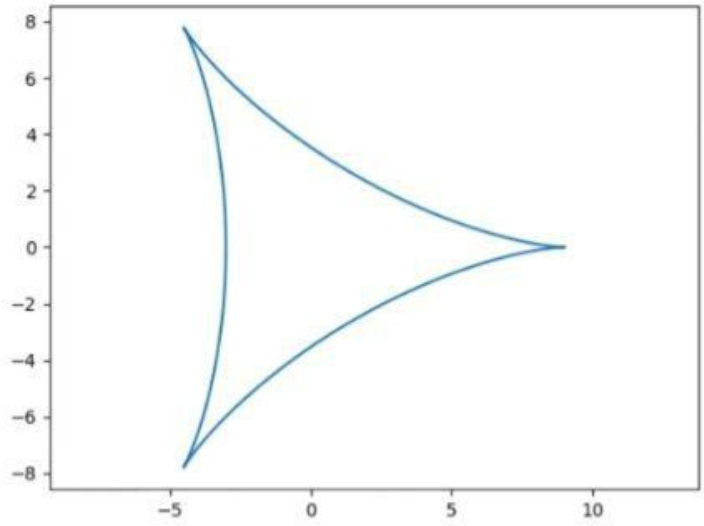

Рисунок 1 – Вид Гипоциклоиды в декартовой системе координат

Уравнение в полярной системе координат имеет следующий вид:

$$r(\theta)=(a-b)+b*\cos((b-a)/b)*\theta), \quad (2)$$

где r — радиус-вектор, θ — угол в полярной системе координат, a — радиус неподвижной окружности и b — радиус катящейся окружности. Увидеть Гипоциклоиду с отношением длин большой и маленькой окружности равным 3 в полярной системе координат можно на рисунке 2.

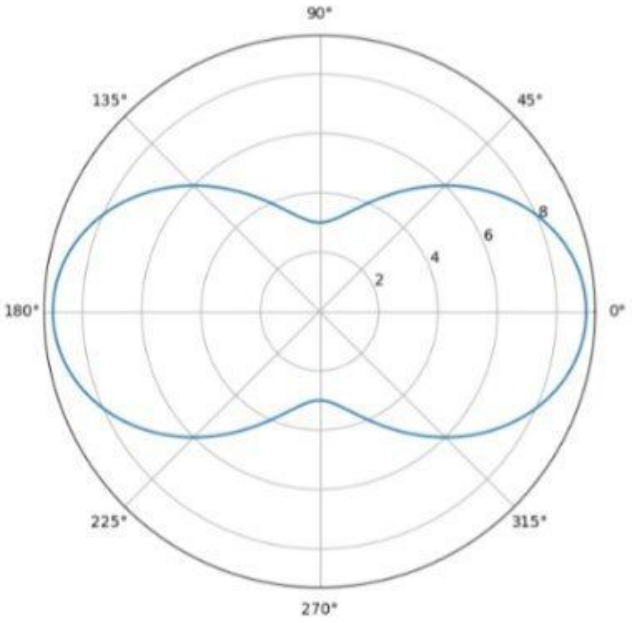

Рисунок 2 – Вид Гипоциклоиды в полярной системе координат

1.3 Алгоритм решения задачи

Задача 1: Создание кривой

Для достижения цели по созданию гипоциклоиды необходимо пройти через следующие этапы:

1. Определение параметров. На данном этапе важно задать значения радиусов (R и r) для создания гипоциклоиды.
2. Вычисление координат. Необходимо реализовать алгоритм для расчета координат точек кривой в декартовой (x, y) и полярной (r, θ) системах координат.
3. Визуализация. Этот этап подразумевает отображение гипоциклоиды в графическом окне, центрировав ее и масштабируя для оптимального отображения без искажений. Также важно разработать алгоритм автоматического масштабирования при изменении размеров окна.
4. Масштабирование. На данном этапе реализуется алгоритм автоматического изменения масштаба графика при изменении размеров окна, сохраняя пропорции и четкость изображения.
5. Оформление. Этап подразумевает добавление координатной сетки для облегчения ориентирования, а также размещение метки шкалы и подписи осей для отображения значений координат. Кроме того, необходимо обеспечить корректное отображение и адаптацию всех элементов интерфейса при изменении размеров окна.

Задача 2: Анимация движения шара по гипоциклоиде

Для решения второй задачи о создании анимации движения шара по гипоциклоиде требуется выполнить следующие этапы:

1. Определение параметров. На данном этапе необходимо задать значение радиуса шара (a). Важным условием будет являться: a> 2 (шар должен быть заметен, но не перекрывать кривую).
2. Траектория движения. Определение последовательности точек на гипоциклоиде, через которые будет проходить шар, и разработка алгоритма перехода между этими точками обеспечит движение примитива по заданному маршруту.
3. Плавная анимация. На этом этапе обеспечивается реализация алгоритма для создания плавного и равномерного движения шара по траектории. Для достижения плавности используется линейная интерполяция между точками.

4. Синхронизация. Необходимо обеспечить синхронизацию анимации шара с изменениями графика. При масштабировании размеров окна шар должен корректно следовать по кривой.

В результате выполнения этих шагов будет разработано программное обеспечение, которое позволяет создавать визуализации и анимации математических кривых, отвечающих заданным требованиям, блок схему работы программы можно увидеть в приложении А.

2. Выбор среды разработки и языка программирования

Для выполнения проекта были выбраны PyCharm, как среда разработки, и Python, в качестве языка программирования. Этот выбор обусловлен их удобством и функциональными возможностями, позволяющими эффективно реализовать требования задачи.

2.1 Язык программирования

При выборе языка программирования для решения задачи по созданию графика кривой с автоматическим масштабированием окна, следует учитывать различные аспекты, такие как простота разработки, гибкость работы с графикой и удобство ввода параметров. На протяжении двух курсов обучения, изучались языки программирования Python, C++, Java, поэтому именно они стали предметом сравнения перед выбором основного языка.

Python выделяется среди трех языков благодаря своей простоте и выразительности. Он предлагает богатые библиотеки для работы с графикой, такие как matplotlib, которые позволяют легко создавать и настраивать графики. Этот язык программирования имеет простой и понятный синтаксис, что упрощает разработку и поддержку кода. Благодаря динамической типизации, Python позволяет быстро прототипировать и тестировать идеи.

C++, в свою очередь, известен своей высокой производительностью и низкоуровневым доступом к аппаратным ресурсам, что делает его предпочтительным выбором для высоконагруженных приложений. Однако, разработка в C++ требует больше усилий из-за необходимости управления памятью и более сложного синтаксиса.

Java сочетает в себе простоту разработки, благодаря автоматическому управлению памятью и простому синтаксису, с высокой переносимостью кода. Он хорошо подходит для крупных проектов, требующих масштабируемости и надежности, однако может быть избыточен для небольших задач из-за того, что требует больше ресурсов и времени для выполнения операций, по причине своей виртуальной машины и строгой системы типов.

Таблица 1

Критерий	C++	Python	Java
Простота и читаемость кода	Требует более строгой структуры кода	Простота, чистота синтаксиса	Требует строгой структуры кода
Производительность	Высокая	Низкая	Средняя

Продолжение таблицы 1

Критерий	C++	Python	Java
Экосистема, библиотеки	Меньше стандартных библиотек, чем Python и Java	Обширные библиотечные базы, активное сообщество	Богатая экосистема, множество библиотек
Области применения	системное программирование, игровые индустрии, высокопроизводительные приложения, при создании операционных систем	быстрое прототипирование, научные вычисления, анализ данных, веб-разработка	корпоративная среда, большие приложения, веб-серверы, Android-разработка

После тщательного анализа всех этих аспектов было решено использовать Python для разработки проекта. Этот язык программирования представляет собой оптимальное решение для создания графика кривой с автоматическим масштабированием по ряду причин. Python обеспечивает быструю разработку по сравнению с другими языками благодаря своему простому и понятному синтаксису. Кроме того, он предлагает мощные инструменты для визуализации данных, такие как библиотека matplotlib, и находится в активном использовании в крупном сообществе разработчиков, что облегчает получение поддержки и решение возникающих задач.

2.2 Среда разработки

Для выполнения задачи на языке Python, связанной с созданием графиков и анимации, я выбрал среду разработки PyCharm. PyCharm представляет собой мощное интегрированное средство разработки (IDE), специально настроенное для работы с Python, обеспечивающее обширные возможности для создания и отладки программного кода.

В отличие от Visual Studio Code (VS Code), PyCharm предлагает ряд ключевых преимуществ, делающих его предпочтительным выбором для данной задачи. Во-первых, PyCharm обладает интегрированной поддержкой для настройки виртуальных сред Python, что упрощает управление зависимостями проекта и изоляцией окружения. Это особенно важно при работе с библиотеками для создания графиков и анимации, такими как matplotlib и numpy.

Во-вторых, PyCharm предоставляет богатые возможности автодополнения кода, статического анализа и быстрой навигации по проекту. Это значительно повышает производительность разработчика, ускоряя процесс написания и отладки кода, что особенно важно при создании сложных визуализаций и анимаций.

Третье преимущество PyCharm — это интегрированная поддержка систем контроля версий, что делает управление изменениями в проекте более удобным и безопасным.

Сравнивая с VS Code, которая также популярна среди разработчиков Python, можно отметить, что VS Code предоставляет хорошую настраиваемость и легковесность. Однако, для задачи, требующей создания сложных визуализаций и анимаций, PyCharm предлагает более глубокие инструменты, включая интеграцию с научными библиотеками и расширенные возможности отладки.

Таблица 2

Критерий	PyCharm	VS Code
Функциональность	Является полнофункциональной интегрированной средой разработки, с широким набором инструментов	Представляет собой легкий текстовый редактор с возможностью расширения функциональности через плагины
Удобство использования	Имеет более узкую специализацию на Python и обладает мощными инструментами для разработки Python-приложений. Интерфейс интуитивно понятен для разработчиков, знакомых с IDE	Имеет более общий подход к разработке, что делает его доступным для широкого круга разработчиков, включая тех, кто работает с различными языками программирования
Сообщество и поддержка	Активное сообщество, обширные ресурсы для поддержки пользователей.	большое сообщество, однако его экосистема может быть менее фокусирована на специфических инструментах и решениях для Python, чем у PyCharm.

Продолжение таблицы 2

Критерий	PyCharm	VS Code
Цена	Есть возможность выбора между Community (бесплатная) и Professional (платная) версиями	Бесплатный и с открытым исходным кодом. Все основные функции и расширения также бесплатны.

Выбор PyCharm для выполнения задачи обусловлен его фокусировкой на Python, обширными интеграционными возможностями с научными библиотеками и удобством работы с проектами любого размера. Эти аспекты особенно важны при создании и отладке программы, которая отображает графики и выполняет анимацию в реальном времени.

3. Процесс написания программного кода

Написание программного кода для визуализации гипоциклоиды с использованием библиотек Tkinter и Matplotlib представляет собой многоэтапный процесс, включающий в себя несколько ключевых этапов: начиная с импорта необходимых библиотек и создания интерфейса до настройки визуализации, анимации.

На первом этапе необходимо импортировать нужные для программы библиотеки. В данном случае используются библиотеки для работы с числами (numpy), графиками (matplotlib), анимацией (matplotlib.animation), визуализацией в Tkinter (matplotlib.backends.backend_tkagg) и сам Tkinter.

Следующим шагом разработки кода для визуализации гипоциклоиды происходит определение функций, которые выполняют расчеты необходимые для построения графика, это функции hypocycloid_cartesian и hypocycloid_polar. Hypocycloid_cartesian функция, которая вычисляет координаты точек гипоциклоиды в декартовой системе координат. Hypocycloid_polar функция, вычисляющая радиус-вектор точек гипоциклоиды в полярной системе координат. Эти функции предназначены для выполнения математических расчетов, необходимых для построения графика гипоциклоиды. Они используются непосредственно в функции update_graph, которая обновляет график в зависимости от выбранной пользователем системы координат (декартовой или полярной). Они также интегрируются с библиотекой Matplotlib, что позволяет легко и эффективно построить как декартов, так и полярные графики гипоциклоиды. Matplotlib используется для визуализации данных, предоставляя мощные инструменты для создания графиков различных типов. Этот этап обеспечивает необходимые математические расчеты, основанные на заданных параметрах, что в дальнейшем используется для создания графического представления данных.

Затем создается главное окно Tkinter, в котором размещаются виджеты для ввода параметров гипоциклоиды (a, b, r), переключатель для выбора системы координат. Использование библиотеки Tkinter позволяет интегрировать график Matplotlib в оконное приложение, что обеспечивает динамическое обновление и отображение гипоциклоиды при изменении параметров или выборе системы координат.

Настройка интерфейса и графика на этом этапе позволяет учитывать размеры и разрешения экрана, что делает программу более удобной для использования. Для этого создается объект фигуры fig и подграфики для декартовой и полярной систем координат, и используется метод FigureCanvasTkAgg, который создает холст для встраивания графика

Matplotlib в оконное приложение Tkinter. Таким образом происходит подготовка к отображению графика, необходимая для последующего визуального представления данных.

Далее создается функция для обновления графика в зависимости от изменений пользовательских параметров. Это ключевой момент в процессе, так как функция обеспечивает динамическое изменение визуализации в реальном времени в ответ на действия пользователя. Функция update_graph вызывается для перерисовки графика при изменении параметров (a, b, r) или выборе системы координат. В зависимости от выбранной системы координат обновляется соответствующий подграфик.

Для создания анимации используется функция animate, которая вызывается каждый кадр анимации. В данном случае анимируется движение по гипоциклоиде с помощью изменения угла или времени. Функция animate(i) создается для обновления графика на каждом кадре анимации. Параметр i представляет собой текущий номер кадра. Из переменных a_var, b_var, r_var получаются текущие значения параметров a, b и r. Создаются массивы данных t и theta для генерации точек времени и углов соответственно. Далее вычисляются координаты гипоциклоиды с использованием функций hypocycloid_cartesian() и hypocycloid_polar() для декартовой и полярной систем координат и координаты текущей точки на гипоциклоиде с помощью параметра i. Обновление содержимого графика для каждого типа координат происходит по схеме: очищается текущий подграфик, строится график гипоциклоиды в выбранной системе координат, добавляется текущая точка в виде красной окружности. Анимация продолжается до достижения последнего кадра в диапазоне frames. Пользователи могут наблюдать, как точка перемещается по гипоциклоиде, вводя изменения в параметры (a, b, r), что способствует более глубокому и интуитивному пониманию характеристик кривой.

Также добавлены кнопки для управления масштабом графика и обработчики событий для изменения размера окна, что автоматически перенастраивает масштаб и размеры графика. Создается функция zoom, которая изменяет текущий масштаб графика в зависимости от выбранной системы координат. Эта функция вызывает zoom_cartesian или zoom_polar для соответствующей системы координат. Важно, чтобы график автоматически адаптировался к изменению размеров окна. Для этого создается функция on_resize(event), которая пересчитывает размеры фигуры и обновляет компоновку элементов графика при изменении размеров окна.

В завершении происходит запуск основного цикла обработки событий в Tkinter, он является конечным этапом разработки приложения. Этот этап отвечает за поддержание работоспособности интерфейса и обработку всех взаимодействий пользователя с программой. Весь цикл работы Tkinter управляется функцией mainloop(), которая запускает основной цикл

обработки событий. Эта функция заставляет программу ожидать взаимодействия с пользователем, таких как нажатие кнопок, ввод данных или изменение размеров окна.

Процесс написания данного кода требует интеграции знаний Matplotlib для визуализации графиков, Tkinter для создания графического интерфейса, а также использования функций и методов NumPy для расчетов. Каждый этап разработки, начиная от проектирования интерфейса до настройки анимации и обработки событий, важен для создания полнофункционального приложения для визуализации гипоциклоиды.

4. Тестирование программного кода

Визуализация математических фигур, таких как гипоциклоид, требует точности в отображении и управлении графиками. В процессе разработки интерактивного приложения для построения гипоциклоида возникла проблема с корректным масштабированием и центрированием графиков как в декартовой, так и в полярной системах координат. Ошибка заключалась в смещении центра графиков после масштабирования, что затрудняло восприятие и анализ фигур. Основная проблема заключалась в неправильной настройке пределов осей графиков. В исходной реализации пределы осей для декартовой системы задавались через минимальные и максимальные значения координат, умноженные на масштабный фактор. Это приводило к тому, что после каждого изменения масштаба центр графика смещался, и график не оставался в центре экрана. Аналогичная проблема наблюдалась и в полярной системе координат, где максимальный радиус задавался без учета корректного центрирования. Чтобы решить проблему, были внесены значительные изменения в код, направленные на корректное центрирование и масштабирование графиков. Ранее для задания пределов осей использовались минимальные и максимальные значения координат x и y. Это приводило к смещению центра графика при изменении масштаба. В новой реализации учитывается максимальное значение среди координат x и y, умноженное на масштабный фактор. Это позволяет сохранить график в центре экрана при изменении масштаба. Этот подход гарантирует, что после каждого изменения масштаба график остается симметрично центрированным относительно начала координат. Аналогично, для полярной системы координат максимальный радиус теперь задается с учетом масштабного фактора. Это позволяет правильно центрировать график и избежать смещения при изменении масштаба. Внесенные изменения обеспечивают, что радиус-вектор гипоциклоида сохраняет пропорции и остается правильно центрированным. Внесенные изменения значительно улучшили визуализацию гипоциклоида, обеспечивая корректное масштабирование и центрирование графиков. Теперь пользователи могут без труда изменять масштаб и наблюдать за изменениями в графике, не теряя при этом центрирование фигуры.

Также при тестировании была обнаружена ошибка связаная с неправильным применением масштабирования в полярной системе координат. Конкретно, при нажатии на кнопку "Zoom In" график отдаляется, а при нажатии на кнопку "Zoom Out" — приближается. Такое поведение является противоположным ожидаемому. Для полярной системы координат изменение масштаба реализовано через переменную polar_scale_factor. При нажатии на кнопку "Zoom In" переменная увеличивается, что, в свою очередь, приводит к увеличению

максимального радиуса отображаемого графика (rmax). Из-за этого график визуально отдаляется, так как отображаемая область становится больше. Аналогично, при нажатии на кнопку "Zoom Out" переменная уменьшается, что приводит к уменьшению области отображения и, как следствие, к визуальному приближению графика. Для исправления этого поведения необходимо изменить масштабирование так, чтобы "Zoom In" уменьшал область отображения, а "Zoom Out" увеличивал. Это достигается путем инверсии операций умножения и деления для переменной polar_scale_factor. В новой версии при нажатии на "Zoom In" вызывается zoom_polar(1.2), что уменьшает polar_scale_factor на 20%, а при нажатии на "Zoom Out" вызывается zoom_polar (1/1.2), что увеличивает polar_scale_factor на 16.67%. Это приводит к правильному визуальному поведению.

Благодаря всестороннему тестированию кода были выявлены и успешно исправлены ошибки, связанные с неправильным масштабированием в полярной системе координат. Этот процесс не только улучшил работу кнопок, обеспечивая правильное изменение масштаба графика, но и существенно повысил качество программы.

Заключение

В ходе данной учебной практики были успешно достигнуты поставленные цели и выполнены задачи, направленные на разработку интерактивной программы для визуализации Гипоциклоиды в декартовой и полярной системах координат, а также анимации движения примитива по её траектории. Программа включает в себя не только пользовательский интерфейс с возможностью ввода параметров уравнения кривой, но и функционал автоматического масштабирования графика и его элементов при изменении размеров окна.

Разработка плавной анимации движения красного шара по траектории Гипоциклоиды демонстрирует умение использовать результаты визуализации для создания интерактивных и наглядных примеров математических объектов. Этот процесс не только укрепил понимание особенностей Гипоциклоиды, но и позволил освоить современные методы программирования для визуализации и анимации в Python.

Итоговая программа представляет собой полезный инструмент для обучения и исследования математических кривых, обеспечивая пользователям возможность интерактивного моделирования и визуализации движения, что способствует более глубокому и наглядному усвоению материала.

Список использованных источников

1. Гипоциклоида // Википедия: свободная энциклопедия [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Гипоциклоида (дата обращения: 18.06.2024)

2. Python Docs [Электронный ресурс] URL: https://docs.python.org/3/library/tk.html (дата обращения: 20.06.2024)

3. ИНФОУРОК [Электронный ресурс] URL: https://infourok.ru/issledovatelskiy-proekt-po-matematike-uchaschihsya-klassa-gipocikloida-3773229.html (дата обращения: 20.06.2024)

4. СК-СТО-ТР-04-1.005-2015. Требования к оформлению текстовой части выпускаемых квалификационных работ, курсовых работ (проектов), рефератов, контрольных работ, отчетов по практикам, лабораторным работам [Электронный ресурс] // ВГУЭС – URL: https://www.vvsu.ru/files/ED115C05-F320-42DE-BFB4-8A6348317716.pdf

Приложение А