ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

канд. техн. наук, доцент		А. В. Аграновский
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 2

ИССЛЕДОВАНИЕ АФФИННЫХ ПРЕОБРАЗОВАНИЙ НА ПЛОСКОСТИ

по курсу:

КОМПЬЮТЕРНАЯ ГРАФИКА

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. № 4326		Г. С. Томчук
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

1 Цель работы

Целью работы является ознакомление с теоретическими основами линейных преобразований на плоскости, анализ и практическая реализация двумерных аффинных преобразований с целью изучения их особенностей и применения на практике.

2 Формулировка задания

Необходимо реализовать программу изображения движущегося по окружности бумеранга (вариант 20). Начало и окончание движения – в одной точке. Рисование фигур (точек, отрезков) на экране монитора можно, но не обязательно, осуществлять с помощью специализированных библиотек. Аффинные преобразования необходимо выполнять только путем композиции матричных вычислений (можно, но не обязательно, использовать библиотеки программ для матричных вычислений). Использование созданных другими авторами программ аффинных преобразований не допускается.

3 Теоретические сведения

Аффинные преобразования — это класс геометрических преобразований, которые сохраняют прямолинейность и параллельность линий, но могут изменять расстояния и углы. Эти преобразования включают в себя такие операции, как сдвиг, масштабирование, поворот, отражение и их комбинации. Аффинные преобразования важны для работы с графикой, так как позволяют изменять объекты на плоскости, не нарушая их топологических свойств. Основные виды аффинных преобразований:

1. Сдвиг

- Сдвиг перемещает объект на заданное расстояние вдоль осей координат. Это преобразование не изменяет форму и размеры объекта.

- Для вектора $v = (x, y)$ сдвиг описывается как добавление фиксированного вектора $t = (tx, ty)$, что выражается формулой:

$$x' = x + tx, y' = y + ty$$

- Матрица преобразования:

$$\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix}$$

2. Масштабирование

- Масштабирование изменяет размеры объекта, умножая его координаты на коэффициенты масштабирования вдоль осей x и y.

- Формула для масштабирования:

$$x' = sx * x, y' = sy * y$$

- Матрица масштабирования:

$$\begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Если $sx = sy$, это называется гомотетия, которая сохраняет пропорции фигуры.

3. Поворот

- Поворот изменяет ориентацию объекта вокруг начала координат на

угол θ.

- Формула поворота:

$$x' = x * cos(\theta) - y * sin(\theta), y' = x * sin(\theta) + y * cos(\theta)$$

- Матрица поворота:

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Поворот на плоскости сохраняет длину отрезков и углы между ними.

4. Отражение

- Отражение симметрично отображает фигуру относительно некоторой оси (например, оси x , y или произвольной прямой).

- Матрица отражения относительно оси x:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Матрица отражения относительно оси y:

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

5. Сжатие и сдвиг

- Сжатие изменяет форму объекта, двигая его точки вдоль одной из осей, что приводит к искажению углов, но сохраняет параллельность прямых.

- Формула преобразования:

$$x' = x + ky \text{ (сдвиг по оси } x), y' = y + kx \text{ (сдвиг по оси } y)$$

- Матрица сжатия:

$$\begin{bmatrix} 1 & k & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ или } \begin{bmatrix} 1 & 0 & 0 \\ k & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Композиция аффинных преобразований. Аффинные преобразования можно комбинировать путем перемножения их матриц. Итоговое преобразование зависит от порядка выполнения операций. Например, сначала можно выполнить сдвиг, затем поворот, а потом масштабирование, что

приведет к композиции этих преобразований.

Однородные координаты. Для удобства работы с аффинными преобразованиями часто используются однородные координаты, добавляя к двумерной системе координат *(x, y)* дополнительную координату *w*, которая позволяет представлять преобразования в виде матрицы 3x3. Это упрощает матричные вычисления и позволяет записывать все виды аффинных преобразований единообразно.

Применение. Аффинные преобразования широко применяются в компьютерной графике для моделирования движений объектов, их масштабирования, изменения ориентации и других операций. Они также используются для преобразования изображений, таких как трансформация координат изображения или картографическая проекция.

4 Описание алгоритма решения

Задача заключается в анимированном перемещении бумеранга по окружности с использованием аффинных преобразований. Алгоритм решения можно разделить на несколько этапов:

1. Определение фигуры бумеранга:
   - На первом этапе необходимо задать форму бумеранга на плоскости. В качестве упрощенной модели можно использовать набор координат, описывающих вершины треугольника или другой геометрической фигуры, которая будет служить моделью бумеранга.
2. Инициализация параметров движения:
   - Задается радиус окружности, по которой будет двигаться бумеранг, и угол поворота, на который бумеранг будет вращаться на каждом шаге анимации.
   - Определяются начальная точка и угол для начала движения.
3. Формирование матриц аффинных преобразований:
   - Матрица поворота: для того чтобы бумеранг перемещался по кругу, на каждом шаге применяется матрица поворота. Эта

матрица будет использоваться для поворота бумеранга относительно начала координат на заданный угол θ.

- Матрица сдвига: на каждом шаге бумеранг перемещается вдоль окружности, поэтому нужно вычислять сдвиг его положения на основе текущего угла и радиуса окружности.
- Композиция преобразований: итоговое положение бумеранга на каждом шаге движения рассчитывается с помощью композиции матрицы поворота и матрицы сдвига.

4. Вычисление новых координат фигуры на каждом шаге:
   - Для каждого шага анимации рассчитываются новые координаты всех вершин бумеранга с помощью умножения исходных координат на матрицу преобразований.
   - Перемещение происходит по окружности: для этого угол увеличивается на каждом шаге, а соответствующие новые координаты центра движения рассчитываются как $cx = R * cos(\theta)$ и $cy = R * sin(\theta)$, где R — радиус окружности, θ — текущий угол.
5. Анимация движения:
   - Реализуется цикл, который на каждом шаге обновляет положение бумеранга на экране, изменяя его координаты в соответствии с матрицей преобразований.
   - Используется библиотека для визуализации (например, matplotlib), которая отображает измененные координаты бумеранга.
   - Анимация продолжается до тех пор, пока бумеранг не совершит полный оборот по окружности и не вернется в начальную точку.
6. Завершение анимации:
   - Анимация завершается, когда угол θ достигает 360°, то есть

бумеранг вернется в исходную точку, завершив полный круг.

5 Выбор языка программирования и используемых библиотек

Для написания программы анимации бумеранга я выбрал язык программирования Python. Данный язык имеет богатую экосистему библиотек, в том числе и необходимых для работы с отрисовкой и графикой. Помимо этого, я уже имел опыт работы с данным языком, что ускорило и облегчило рабочий процесс.

Используемые библиотеки:

- NumPy — используется для работы с матрицами и векторами, что упрощает вычисления аффинных преобразований, таких как поворот, сдвиг и масштабирование. Это стандартная библиотека для эффективных математических операций в Python.
- Matplotlib — отвечает за визуализацию и анимацию движения бумеранга. В частности, модуль matplotlib.animation позволяет легко создавать анимации, обновляя координаты объектов на каждом шаге.

Обе библиотеки широко используются в научных и инженерных вычислениях, что делает их подходящими для задач, связанных с компьютерной графикой и аффинными преобразованиями.

6 Описание разработанной программы

Программа (см. Приложение А) реализует анимацию движения бумеранга по окружности с использованием аффинных преобразований на плоскости. Бумеранг представлен в виде треугольной геометрической фигуры, которая последовательно поворачивается и перемещается вдоль заданной окружности, возвращаясь в исходную точку. Основные компоненты программы:

1. Определение фигуры бумеранга. Фигура бумеранга задается как треугольник с тремя вершинами. Эти вершины определяются с помощью двумерных координат.
2. Аффинные преобразовани. Матрица поворота вычисляется для

вращения бумеранга вокруг начала координат на каждом шаге анимации с помощью библиотеки numpy. Сдвиг по окружности рассчитывается на основе радиуса окружности и текущего угла поворота. Центр перемещения бумеранга изменяется с каждым кадром для движения вдоль окружности. Эти преобразования комбинируются, чтобы обеспечить плавное перемещение и вращение бумеранга.

3. Анимация движения. Для создания анимации используется библиотека matplotlib. Каждый кадр анимации обновляет координаты бумеранга с учетом текущего поворота и положения на окружности. Бумеранг непрерывно вращается и перемещается до тех пор, пока не совершит полный оборот по окружности и не вернется в начальную точку.
4. Логика обновления. В каждый момент времени вычисляется текущий угол поворота θ, и на основе этого угла пересчитываются новые координаты бумеранга с помощью матриц аффинных преобразований. Результаты преобразований отображаются в каждом кадре анимации.
5. Окончание анимации. Программа завершает анимацию, когда бумеранг совершает полный оборот, пройдя 360° по окружности.

7 Скриншоты, иллюстрирующие результаты работы программы

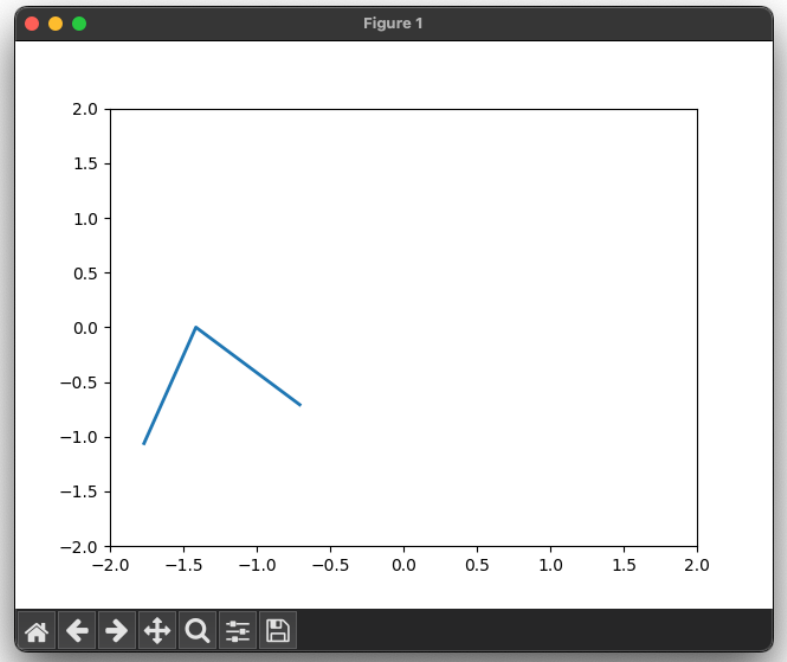

Рисунок 1

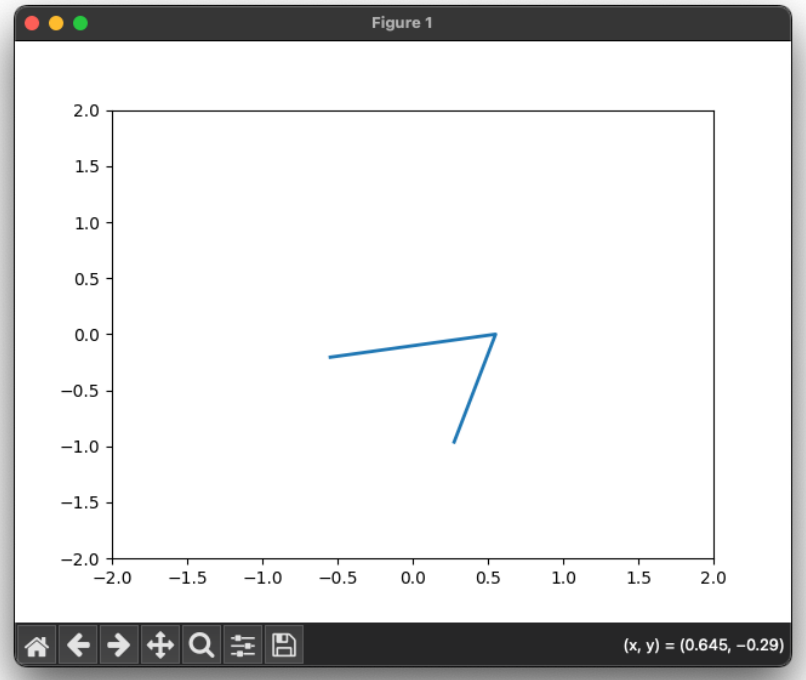

Рисунок 2

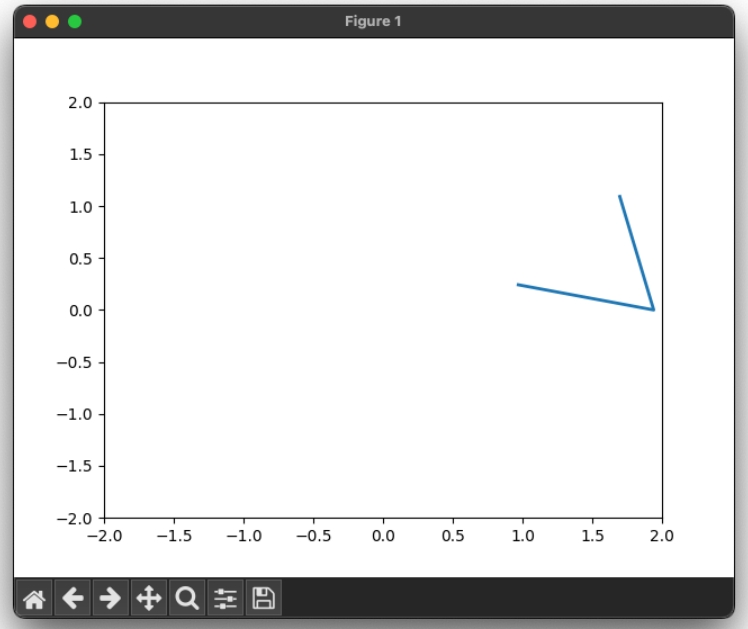

Рисунок 3

8 Вывод

В ходе выполнения лабораторной работы были изучены теоретические основы аффинных преобразований на плоскости, включая такие виды, как сдвиг, масштабирование, поворот и отражение. Аффинные преобразования — это важный инструмент для работы с двумерной графикой, поскольку они позволяют изменять объекты на плоскости без нарушения их топологических свойств, таких как параллельность линий и прямолинейность.

Полученные знания и навыки:

- Понимание различных типов линейных преобразований и их влияния на геометрические фигуры.
- Изучение свойств матричных преобразований и их применения в двумерной графике.
- Разбор однородных координат и их использования для удобного выполнения аффинных преобразований в одном формате.
- Реализация аффинных преобразований с использованием

матричных вычислений на Python.

- Разработка программы для анимации движения объекта по окружности с помощью матриц преобразований.
- Работа с библиотеками NumPy для матричных операций и Matplotlib для создания анимации и визуализации.

Проблемы и их решения:

1. Точность вычислений при повороте и сдвиге. При работе с аффинными преобразованиями и вращением важно учитывать точность вычислений тригонометрических функций. Возникли небольшие отклонения в вычислениях координат, которые приводили к неточному отображению движения бумеранга. Для повышения точности вычислений были использованы функции из библиотеки NumPy, которые обеспечивают более высокую точность при работе с матрицами и тригонометрическими операциями.
2. Сложности с синхронизацией анимации. При попытке плавной анимации возникла проблема с тем, что обновления экрана происходили слишком быстро или медленно, что делало движение неровным. Использование встроенного механизма управления частотой кадров в Matplotlib позволило синхронизировать обновление экрана и сделать анимацию более плавной.
3. Отображение возвращения объекта в начальную точку. Возникла задача плавного возвращения бумеранга в начальную точку по завершении движения по окружности. Тщательная настройка угла поворота и радиуса окружности позволила корректно завершить движение на исходной позиции без смещения.

В результате работы была успешно разработана программа, демонстрирующая движение бумеранга по окружности с использованием аффинных преобразований. Программа корректно реализует анимацию на основе матричных операций, что подтверждает понимание и умение

применять полученные теоретические знания на практике.

Таким образом, выполнение данной лабораторной работы позволило глубже понять принципы аффинных преобразований и их использование в компьютерной графике, а также развить навыки программирования анимации с использованием Python.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Аграновский, А. В. Использование методов преобразования координат для формирования растровых изображений. Учебно-методическое пособие / А. В. Аграновский. — СПб.: Редакционно-издательский центр ГУАП, 2024. — 40 с.
2. Бобылев, С. И. Основы компьютерной графики: учебное пособие / С. И. Бобылев. — М.: Издательство МГТУ им. Н. Э. Баумана, 2010. — 224 с.
3. Ласков, А. В. Математические основы компьютерной графики / А. В. Ласков, Ю. Г. Лоскутов. — СПб.: Питер, 2008. — 352 с.
4. NumPy Documentation [Электронный ресурс] — Режим доступа: https://numpy.org/doc/, свободный. — (дата обращения: 29.09.2024).

ПРИЛОЖЕНИЕ А

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation

# Задание фигуры бумеранга
boomerang = np.array([
    [0, 0],  # Вершина 1
    [1, 0],  # Вершина 2
    [0.5, 1]  # Вершина 3
])


# Функция для создания матрицы поворота
def rotation_matrix(angle):
    return np.array([
        [np.cos(angle), -np.sin(angle)],
        [np.sin(angle), np.cos(angle)]
    ])


# Функция для вращения и перемещения бумеранга
def transform_boomerang(boomerang, angle, radius):
    # Вращение (умножение матрицы)
    rotated_boomerang = boomerang @ rotation_matrix(angle)

    # Перемещение по окружности
    cx = radius * np.cos(angle) # координаты центра окружности по оси X
    cy = radius * np.sin(angle) # # координаты центра окружности по оси Y

    # Сдвиг фигуры (сложение матриц)
    return rotated_boomerang + np.array([cx, cy])


# Настройки анимации
# figure - объект фигуры, представляет собой окно для графического
отображения. В нем размещаются оси и другие элементы визуализации
# axes - объект осей внутри фигуры. Это область, где происходит построение
графиков и анимаций, включая рисование объектов и управление их свойствами
figure, axes = plt.subplots()
axes.set_xlim(-2, 2)
axes.set_ylim(-2, 2)
line, = axes.plot([], [], lw=2) # объект линии контура бумеранга


# Функция обновления для анимации
def update(frame):
    angle = np.radians(frame)
    transformed_boomerang = transform_boomerang(boomerang, angle, radius=1)

    line.set_data(transformed_boomerang[:, 0], transformed_boomerang[:, 1])
    return line,


# Создание анимации
# anim - объект анимации, отвечает за обновление отображения на каждом кадре
анимации и управляет ее динамическим поведением
anim = animation.FuncAnimation(figure, update, frames=range(0, 360),
interval=50, blit=True)
plt.show()
```