Міністерство освіти і науки України

Національний технічний університет України «КПІ» імені Ігоря Сікорського

Кафедра обчислювальної техніки ФІОТ

ЗВІТ

з лабораторної роботи №2

з навчальної дисципліни «Комп’ютерна графіка»

Тема:

ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ПОБУДОВИ ТА ПЕРЕТВОРЕННЯ КООРДИНАТ ПЛОЩИННИХ (2D) ОБ’ЄКТІВ

Виконала:

Студентка 2 курсу кафедри ОТ ФІОТ,

Навчальної групи ІВ-92

Орлова Ю.Д.

Перевірив:

Професор кафедри ОТ ФІОТ

Писарчук О.О.

Київ 2020

**І. Мета:**

Виявити, дослідити та узагальнити особливості побудови та перетворення координат площинних (2d) об’єктів.

**ІІ. Завдання:**

Задача №1. Для мобільного додатку ретрогри здійснити синтез математичних моделей та розробити скрипт, що реалізує базові операції перетворення над графічними примітивами.

Технічні умови реалізації завдання наведені у таблиці Д.1 додатку 1.

|  |  |  |
| --- | --- | --- |
| Варіант | Технічні умови | Графічна фігура |
| 16 | Реалізувати операції: обертання – масштабування – переміщення. 3. операцію реалізувати циклічно, траєкторію зміни положення цієї операції скрити. Обрати самостійно: бібліотеку, розмір графічного вікна, розмір фігури, параметри реалізації операцій, кольорову гамму усіх графічних об’єктів. Всі операції перетворень мають здійснюватись у межах графічного вікна. | Квадрат |

**Завдання ІІІ рівня – максимально 7 балів.**

Реалізувати задачу №1 лабораторної роботи з використанням трьох підходів:

1. Шляхом здійснення матричних обчислень та технологій композиції перетворень. Вхідний вектор параметрів геометричної фігури має бути розширеним.

2. Шляхом здійснення послідовних матричних обчислень. Вхідний вектор параметрів геометричної фігури за структурою є вектором-стовпцем, що містить окремо координати кожної точки геометричної фігури.

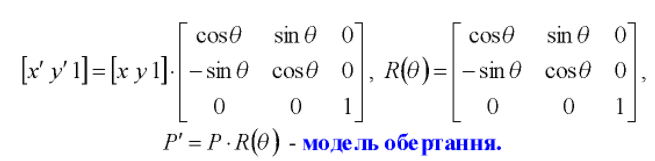
3. Із впровадженням скалярних обчислень. Для розробки коду використовувати механізми підпрограм (узагальнення повторюваних операцій з використанням функцій, методів).

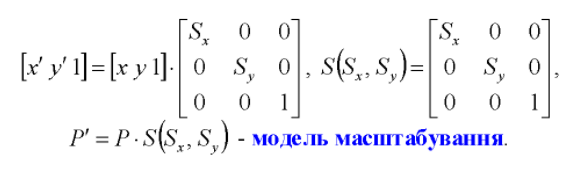
**ІІІ. Результати виконання лабораторної роботи.**

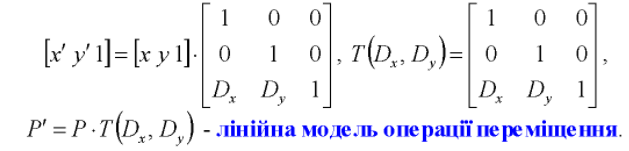
**3.1. Синтезована математична модель перетворень графічних об’єктів відповідно до індивідуального завдання.**

Відповідно до умов задачі синтезовано математичну модель операцій над структурою вхідного графічного об’єкту.

Модель реалізує послідовність дій обертання – масштабування – переміщення.

Обертання реалізується за узагальненою моделлю:

Масштабування реалізується за узагальненою моделлю:

Переміщення реалізується за узагальненою моделлю:

**3.2. Блок схема алгоритму та її опис.**

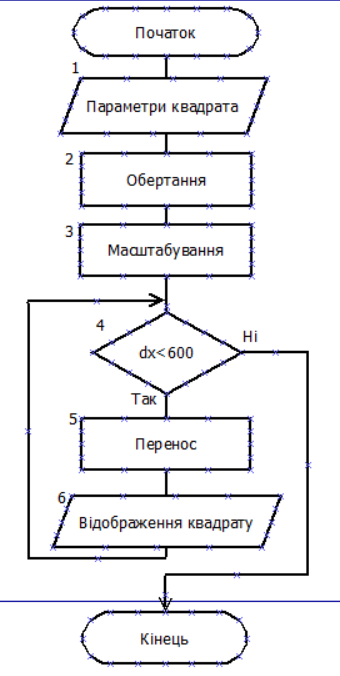


Рис.1. Блок-схема алгоритму програми.

Робота алгоритму розпочинається із формування сегменту початкових даних, що описують квадрат. Квадрат формується з 4 вершин, кожна з яких має 2 координати. Описані дії реалізуються у блоці 1 блок-схеми алгоритму рис.1.

Відповідно до математичної моделі (1) в блоці 2 реалізується обертання квадрата. В блоці 3 реалізується масштабування фігури. В блоці 4 перевіряється умова циклу, при виконанні якої у блоці 5 реалізується перенос, а в блоці 6 – відображення фігури у графічному вікні. Далі знову виконується блок 4. Коли умова не виконується, робота алгоритму завершується.

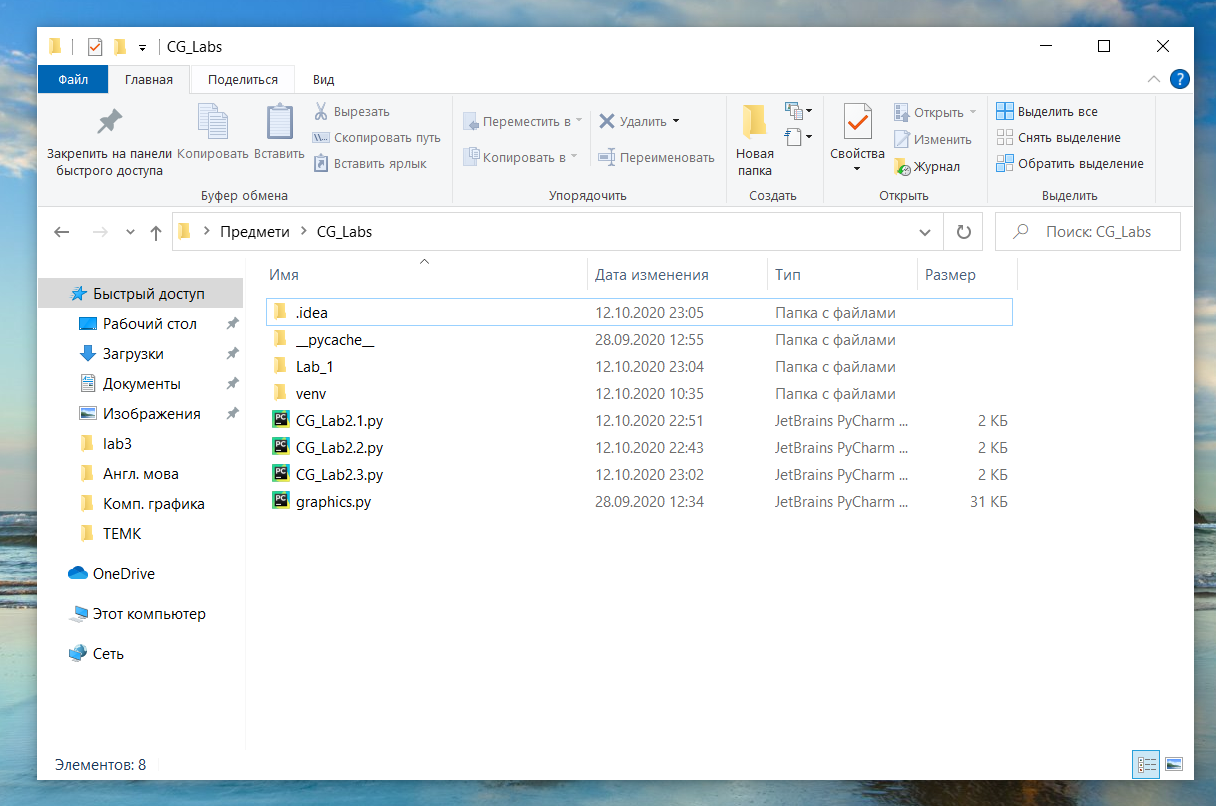
**3.3. Опис структури проекту програми в середовищі PyCharm.**

Рис.2. Структура проекту.

KG\_Labs – головний каталог проекту

KG\_Lab2.1.py, KG\_Lab2.2.py, KG\_Lab2.3.py – файли програмного коду лабораторної роботи;

graphics.py – файл графічної бібліотеки graphics.

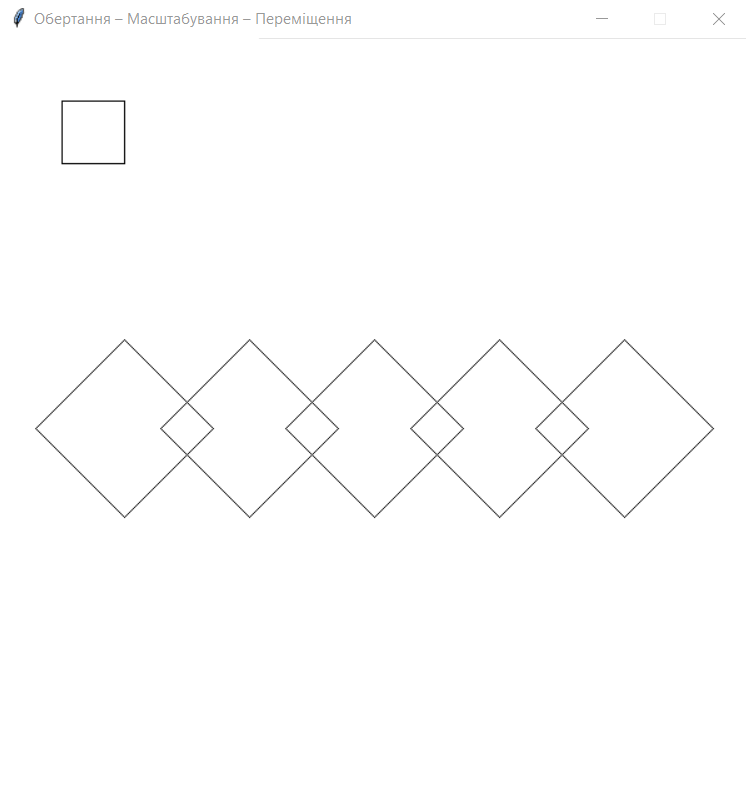
**3.4. Результати роботи програми відповідно до завдання.**

Рис.3. Первинне положення паралелепіпеду.

**3.5. Програмний код.**

Програмний код послідовно реалізує алгоритм рис.1 та спрямовано на отримання результатів, поданих на рис.3.

from graphics import \*  
import numpy as np  
import time  
  
x\_window = 600  
y\_window = 600  
step = 50  
win = GraphWin("Обертання – Масштабування – Переміщення", x\_window, y\_window)  
win.setBackground('white')  
  
x1 = 50; y1 = 50; x2 = 100; y2 = 50  
x3 = 100; y3 = 100; x4 = 50; y4 = 100  
  
dx = 100; dy = 100  
sx = 2; sy = 2  
teta = np.pi/4  
  
obj = Polygon(Point(x1, y1), Point(x2, y2), Point(x3, y3), Point(x4, y4))  
obj.draw(win)  
  
while dx < 600:  
 time.sleep(0.2)  
 p = np.array([[sx \* (x1 \* np.cos(teta) - y1 \* np.sin(teta)) + dx, sy \* (x1 \* np.sin(teta) + y1 \* np.cos(teta)) + dy, 1],  
 [sx \* (x2 \* np.cos(teta) - y2 \* np.sin(teta)) + dx, sy \* (x2 \* np.sin(teta) + y2 \* np.cos(teta)) + dy, 1],  
 [sx \* (x3 \* np.cos(teta) - y3 \* np.sin(teta)) + dx, sy \* (x3 \* np.sin(teta) + y3 \* np.cos(teta)) + dy, 1],  
 [sx \* (x4 \* np.cos(teta) - y4 \* np.sin(teta)) + dx, sy \* (x4 \* np.sin(teta) + y4 \* np.cos(teta)) + dy, 1]])  
 x11 = p[0, 0]; y11 = p[0, 1]  
 x22 = p[1, 0]; y22 = p[1, 1]  
 x33 = p[2, 0]; y33 = p[2, 1]  
 x44 = p[3, 0]; y44 = p[3, 1]  
  
 obj2 = Polygon(Point(x11, y11), Point(x22, y22), Point(x33, y33), Point(x44, y44))  
 obj2.draw(win)  
 dx += 100  
  
win.getMouse()  
win.close()

from graphics import \*  
import numpy as np  
import time  
  
x\_window = 600  
y\_window = 600  
step = 50  
win = GraphWin("Обертання – Масштабування – Переміщення", x\_window, y\_window)  
win.setBackground('white')  
  
x1 = 50; y1 = 50; x2 = 100; y2 = 50  
x3 = 100; y3 = 100; x4 = 50; y4 = 100  
  
dx = 100; dy = 100  
sx = 2; sy = 2  
teta = np.pi/4  
  
obj = Polygon(Point(x1, y1), Point(x2, y2), Point(x3, y3), Point(x4, y4))  
obj.draw(win)  
  
p = np.array([[x1, y1, 1], [x2, y2, 1], [x3, y3, 1], [x4, y4, 1]])  
r = np.array([[np.cos(teta), -np.sin(teta), 0], [np.sin(teta), np.cos(teta), 0], [0, 0, 1]])  
fr = r.T  
total = np.dot(p, fr)  
s = np.array([[sx, 0, 0], [0, sy, 0], [0, 0, 1]])  
fs = s.T  
total2 = np.dot(total, fs)  
  
while dx < 600:  
 time.sleep(0.2)  
 t = np.array([[1, 0, dx], [0, 1, dy], [0, 0, 1]])  
 ft = t.T  
 total3 = np.dot(total2, ft)  
 x11 = total3[0, 0]; y11 = total3[0, 1]  
 x22 = total3[1, 0]; y22 = total3[1, 1]  
 x33 = total3[2, 0]; y33 = total3[2, 1]  
 x44 = total3[3, 0]; y44 = total3[3, 1]  
 dx += 100  
 obj2 = Polygon(Point(x11, y11), Point(x22, y22), Point(x33, y33), Point(x44, y44))  
 obj2.draw(win)  
  
  
win.getMouse()  
win.close()

from graphics import \*  
import numpy as np  
import time  
  
x\_window = 600  
y\_window = 600  
step = 50  
win = GraphWin("Обертання – Масштабування – Переміщення", x\_window, y\_window)  
win.setBackground('white')  
  
x1 = 50; y1 = 50; x2 = 100; y2 = 50  
x3 = 100; y3 = 100; x4 = 50; y4 = 100  
  
dx = 100; dy = 100  
sx = 2; sy = 2  
teta = np.pi/4  
  
obj = Polygon(Point(x1, y1), Point(x2, y2), Point(x3, y3), Point(x4, y4))  
obj.draw(win)  
  
  
while dx < 600:  
 time.sleep(0.2)  
 x11 = (x1\*np.cos(teta)-y1\*np.sin(teta))\*sx + dx  
 y11 = (x1\*np.cos(teta)+y1\*np.sin(teta))\*sy + dy  
 x22 = (x2\*np.cos(teta)-y2\*np.sin(teta))\*sx + dx  
 y22 = (x2\*np.cos(teta)+y2\*np.sin(teta))\*sy + dy  
 x33 = (x3\*np.cos(teta)-y3\*np.sin(teta))\*sx + dx  
 y33 = (x3\*np.cos(teta)+y3\*np.sin(teta))\*sy + dy  
 x44 = (x4\*np.cos(teta)-y4\*np.sin(teta))\*sx + dx  
 y44 = (x4\*np.cos(teta)+y4\*np.sin(teta))\*sy + dy  
  
 obj2 = Polygon(Point(x11, y11), Point(x22, y22), Point(x33, y33), Point(x44, y44))  
 obj2.draw(win)  
  
 dx += 100  
  
win.getMouse()  
win.close()

**3.6. Аналіз результатів відлагодження та верифікації результатів роботи програми.**

Результати від лагодження та тестування довели працездатність розробленого коду. Верифікація функціоналу програмного коду, порівняння отриманих результатів з технічними умовами завдання на лабораторну роботу доводять, що усі завдання виконані у повному обсязі.

**IV. Висновки.**

У ході виконання лабораторної роботи проведено дослідження особливостей побудови площинних (2d) об’єктів з використанням можливостей алгоритмічної мови високого рівня Python. Було використано графічну бібліотеку Grafics. Дослідження довели, що моделювання 2d об’єктів та обробка площинних форм відбуваються в реальному їх вигляді без змін у геометричній структурі. Математично все це втілюється у формі перетворень графічних об’єктів з використанням матричних форм.

Виконала: студентка Орлова Ю.Д.