Київський національний університет імені Тараса Шевченка

ННІ "Інститут геології"

Кафедра геоінформатики

**Практична робота № 1**

**Просторовий аналіз та статистичне моделювання в геології**

Виконала студентка 3 курсу, групи «Big Data»

Яковенко Марія Ігорівна

Київ-2023

Завдання

1. У просторі заданий вектор . Визначити, яким буде результат застосування в заданій послідовності наступних перетворень:
   1. – обертання у площині на θ градусів;
   2. – зсув на площині;
   3. – масштабування (розтягування та/або стиснення).

Вирішити завдання для двох варіантів вхідних даних.

1. Запрограмувати мовою Python операцію лінійного відображення вхідного вектора (матриці) у вихідних за допомогою перетворень: обертання , зсуву та масштабування .

Вхідні дані

Таблиця 1. Варіант 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Послідовність* |  | |  |  | |  | |
| *S-R-T* |  |  |  |  |  |  |  |
| *T-R-S* |  |  |  |  |  |

Таблиця 2. Варіант 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Послідовність* |  | |  |  | |  | |
| *S-R-T* |  |  |  |  |  |  |  |
| *T-R-S* |  |  |  |  |  |

Хід роботи (GitHub: <https://github.com/mashaven/sheva-stats/task-1>)

Для спрощення побудови алгоритму в якості нульової операції для даної задачі будемо розглядати операцію тотожного перетворення :

,

де матриця трансформації – одинична матриця:

Вигляд матриць трансформації для операцій обертання , зсуву та масштабування , відповідно, буде наступним:





Операція лінійного перетворення матриці володіє властивістю асоціативності, тобто:

Таким чином, якщо відомі операції трансформацій та послідовність їх застосування – можна розрахувати матрицю лінійного відображення вхідного вектора за наступним алгоритмом:

transformMatrix(operations: dict[Name: Matrix], sequence: list[Name]):   
 Tr: **Matrix** = identity(3)   
 for each name in sequence:   
 Tr = Tr \* operations[name]   
 return Tr

Запрограмуємо розрахунок матриць (1-3) мовою Python:

1. обертання,

def rotate(theta):   
 theta = theta \* np.pi / 180  
 return np.array([  
 [np.cos(theta), -np.sin(theta), 0],   
 [np.sin(theta), np.cos(theta), 0],   
 [0, 0, 1]   
 ])

1. зсув,

def move(x, y):  
 return np.array([  
 [1, 0, x],  
 [0, 1, y],  
 [0, 0, 1]  
 ])

1. масштабування,

def scale(x, y):  
 return np.array([  
 [x, 0, 0],  
 [0, y, 0],  
 [0, 0, 1]  
 ])

Та виконаємо реалізацію методу modelMatrix(inMatrix, sequence) як функції мовою Python для вирішення поставленого завдання за отриманими варіантами:

def modelMatrix(**inMatrix**, **sequence**):  
 V = np.array(**inMatrix**['V'] + (1,)) # V(x, y, 1)  
 # calculate transformation matrix for each operation  
 Tr = {}  
 Tr['R'] = rotate( **inMatrix**['R'])  
 Tr['T'] = move (\***inMatrix**['T'])  
 Tr['S'] = scale (\***inMatrix**['S'])  
 # calculate a transformation matrix for the `sequence`  
 T = np.identity(3)  
 for op in **sequence**:  
 T = T @ Tr[op] # @ is a binary operator for the matrix mul.

# apply the transformation matrix to the vector `V`  
 Vtr = T @ V  
 return Vtr

За допомогою отриманої функції обчислимо розв’язок поставленого завдання:

1. варіант 2

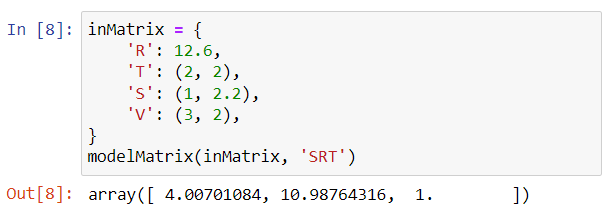


Рисунок 1. Результати перетворень вектору в послідовності ***S-R-T***

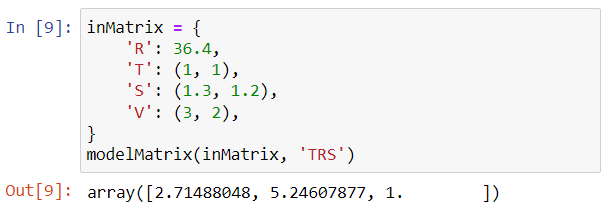


Рисунок 2. Результати перетворень вектору в послідовності ***T-R-S***

1. варіант 3

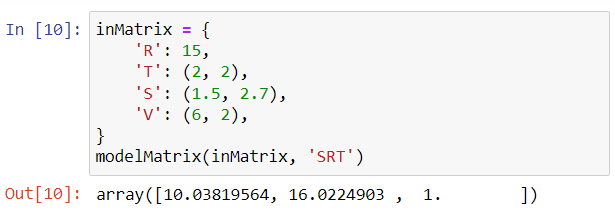


Рисунок 3. Результати перетворень вектору в послідовності ***S-R-T***

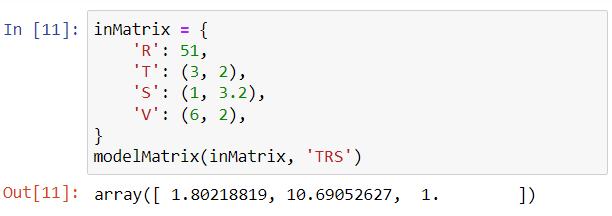


Рисунок 4. Результати перетворень вектору в послідовності ***T-R-S***