Busan science high school

2023 Ocean ICT Festival
2023 BOIF

B
25

QR 코드 영역
QR 삽입 후
테두리 삭제

Youtube 영상 QR

운하로 인한 생태계 교란 시뮬레이션

엄...
2215 엄재현
2219 최온재

탐구동기

이집트 지역의 수에즈 운하는 홍해와 지중해 두 수역을 연결시키며 항해 경로를 대폭 축소시켜 세계 물류의 공급을 원활하게 했다. 그러나 홍해와 지중해라는 두 바다를 연결하게 되어 서로 다른 환경에 살던 어종이 서로의 생태계를 침범하게 되는 리셉스의 이주 현상이 나타난다. 이러한 현상은 수에즈 운하 뿐만이 아닌 한반도와 미국 등 여러 나라의 운하에서 발생하게 된다. 이러한 생태계 교란의 영향이 지중해와 홍해 뿐만이 아닌 우리나라에서도 발생할 수 있어 이러한 현상의 심각성을 확인해 보고자 이번 탐구를 진행하게 되었다

알고리즘 설명

온도 분포 , 염분분포 함수 설정

↓

위치에 따른 특정어종 환경적합성 계산

↓

특정어종의 확산함수 설정

↓

시간에 따른 확산 정도 기록

작품 설명

서로 다른 생태계에 있던 생물들이 연결되면서 이동하는 양상을 시뮬레이션을 통해 파악해 보고자 했다. 실제 데이터와 지형 등을 이용하여 실제 생물의 분포를 모델링하게 하여 운하 건설시 생태계 교란의 가능성을 파악할 수 있다

코드설명

염분 및 온도 분포 함수 설계

```
import numpy as np
import sympy as sp
import math

def temperature():#온도 분포
    x_vals = np.linspace(-0.45, 0.45, 10)
    y_vals = np.linspace(0.45, -0.45, 10)
    x_grid, y_grid = np.meshgrid(x_vals, y_vals)
    coordinate_array = np.dstack((x_grid, y_grid))

    y = sp.symbols('y')
    ex = 30 - 0.3*(y + 10)
    numeric_function = sp.lambdify(y, ex, 'numpy')
    result_array = numeric_function(y_grid)
    return result_array
```

```
import numpy as np
import sympy as sp
import math

def saltinity(): #염분
    x_vals = np.linspace(-0.45, 0.45, 10)
    y_vals = np.linspace(0.45, -0.45, 10)
    x_grid, y_grid = np.meshgrid(x_vals, y_vals)
    coordinate_array = np.dstack((x_grid, y_grid))

    x, y = sp.symbols('x y')
    ey = 40+2*(x-y)

    numeric_function = sp.lambdify((x, y), ey, 'numpy')
    result_array = numeric_function(x_grid, y_grid)
    return result_array
```

생물의 환경적합성을 계산한 함수 설정

```
a=temperature()#나타낸 값들
b=saltinity()
print(a)
print(b)
```

```
print(a**2+b**2)
```

```
k=(np.sqrt(a**2+b**2))
```

```
def f(x):
    return 0.5*np.exp((-0.2)*(x-50)**2)
fishA=(f(k))
```

```
def f(x):
    return 0.5*np.exp((-0.2)*(x-45)**2)
fishB=(f(k))
```

```
print(fishA, end='₩n₩n') #물고기 2종에 대한 생존적합성 계산
print(fishB)
```

생태계 내에서 서로 다른 두 종의 이동을 확인

```
import numpy as np

def simulation_fishA(A, P):
    new_array = np.zeros_like(A)

    for i in range(A.shape[0]):
        if i != 0:
            for j in range(A.shape[1]):
                new_value = A[i, j]
                if j == 0:
                    new_value += (A[i + 1, j] * P[i + 1, j] + A[i, j + 1] * P[i, j])
                    if new_value>=1: new_value=1
                elif 0 < j < A.shape[1] - 1:
                    new_value += (A[i + 1, j] * P[i + 1, j] + A[i, j + 1] * P[i, j + 1] + A[i, j - 1] * P[i, j - 1])
                    if new_value>=1: new_value=1
                elif j == A.shape[1] - 1:
                    new_value += (A[i + 1, j] * P[i + 1, j] + A[i, j - 1] * P[i, j - 1])
                    if new_value>=1: new_value=1

                new_array[i, j] = new_value
        elif i == A.shape[0] - 1:
            for j in range(A.shape[1]):
                new_array[i, j] = A[i, j]

    return new_array
```

```
import numpy as np

def simulation_fishB(A, P):
    new_array = np.zeros_like(A)

    for i in range(A.shape[0]):
        if i != 0:
            for j in range(A.shape[1]):
                new_value = A[i, j]
                if j==0:
                    new_value += (A[i - 1, j] * P[i - 1, j] + A[i, i] * P[i, j])
                    if new_value>=1: new_value=1
                elif 0 < j < A.shape[1] - 1:
                    new_value += (A[i - 1, j] * P[i - 1, j] + A[i, j + 1] * P[i, j + 1] + A[i, j - 1] * P[i, j - 1])
                    if new_value>=1: new_value=1
                elif j == A.shape[1] - 1:
                    new_value += (A[i - 1, j] * P[i - 1, j] + A[i, j - 1] * P[i, j - 1])
                    if new_value>=1: new_value=1

                new_array[i, j] = new_value
        elif i == 0:
            for j in range(A.shape[1]):
                new_array[i, j] = A[i, j]

    return new_array
```

실행 결과 및 느낀점

```
def simulationA(a):
    cnt = 0
    temp = fishA_array  # fishA_array가 미리 정의되어 있다고 가정

    while cnt < a:  # cnt가 a보다 작을 때까지 루프 실행
        temp = simulation_fishA(temp, fishA)
        print(temp)
        cnt += 1
```

```
simulationA(20)
```

```
def simulationB(a):
    cnt = 0
    temp = fishB_array  # fishA_array가 미리 정의되어 있다고 가정

    while cnt < a:  # cnt가 a보다 작을 때까지 루프 실행
        temp = simulation_fishB(temp, fishB)
        print(temp)
        cnt += 1
```

```
simulationB(20)
```

10 X 10 배열에서 어종이 존재할 확률

```
[ 7.23741432e+00 5.61433575e-01]
[1.82799767e+02 2.30512758e+02 1.63714884e+02 8.12659450e+01
 3.13253188e+01 1.02425290e+01 3.07577013e+00 9.97375416e-01
 2.59603857e-01 6.93080015e-02]
[6.13243074e+01 5.96950069e+01 4.30265598e+01 1.81099225e+01
 5.83497585e+00 1.58495201e+00 3.93728033e-01 9.42314062e-02
 2.21006688e-02 4.76716066e-03]
[1.37536043e+01 1.37850416e+01 2.23482766e+00 2.53052643e+00
 6.70339251e-01 1.49286953e-01 3.03157527e-02 5.97914291e-03
 1.10324601e-03 1.89289956e-04]
[2.10416243e+00 1.81558991e+00 7.91141129e-01 2.25390993e-01
 4.83215183e-02 8.71163750e-03 1.42854003e-03 2.21562625e-04
 3.28076532e-05 4.40752240e-06]
[2.20830572e-01 1.59654502e-01 5.64336031e-02 1.28450579e-02
 2.19640583e-03 3.16682090e-04 4.14112937e-05 5.06728565e-06
 5.63729620e-07 6.04588718e-08]
[1.58362323e-02 9.33163718e-03 2.61267678e-03 4.66594161e-04
 6.27921101e-05 7.15619045e-06 7.36739679e-07 7.01424555e-08
 6.19676400e-09 4.87361450e-10]]
```

느낀 점

리셉스의 이주라는 흔하지 않은 주제를 선정하고 탐구를 진행해 보면서 관련 데이터를 찾는 데 어려움이 있었다. 이러한 주제를 이용하여 어떤 탐구를 진행할 수 있는지 생각해 보면서 시뮬레이션을 통한 영향을 확인해 보며 앞으로 진행해보고 싶은 탐구적 내용들이 떠올랐다.