Trees in environment

Aby zasymulować środowisko wykorzystamy voxel space. W środowisku będziemy rozpatrywać tylko jedną cechę - zacienienie.

Klasa Environment

Klasa environment zawiera voxel space z informacją o zacienieniu.

addShadow

Funkcja addShadow zwiększa wartość cienia w zadanej pozycji. Następnie przechodzi w pentlach w dół, zmniejszając z każdym poziomem siłę cienia, ale zwiększając szerokość oddziałowywania.

```
for(int j = (int)voxelPosition.y; j >= 0; j--)
{
      for(int i = (int)voxelPosition.x-rotj; i <=
(int)voxelPosition.x+rotj; i++)
      {
        for(int k = (int)voxelPosition.z-rotj; k <=
(int)voxelPosition.z+rotj; k++)
        {
            float secondaryStrength = 1.0f*strength;
            secondaryStrength = secondaryStrength / ((Mathf.Abs(k-
(int)voxelPosition.z) + 1 + Mathf.Abs(i-(int)voxelPosition.x))/8.0f);
            (...)
        }
      }
}
```

shadowStrength

Zwraca siłę cienia w zadanej pozycji

inVoxelSpace

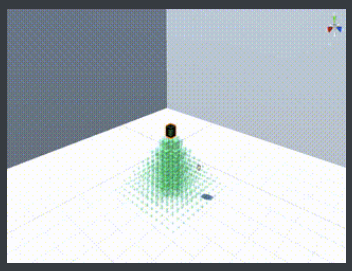
Sprawdza, czy dane współrzędne się mieszczą w voxel space

```
bool inVoxelSpace(int a, int b, int c)
  {
    return (a<sizeX&&a>=0&&b<sizeY&&b>=0&&c<sizeZ&&c>=0);
  }
```

positionInVoxel i positionInWorld

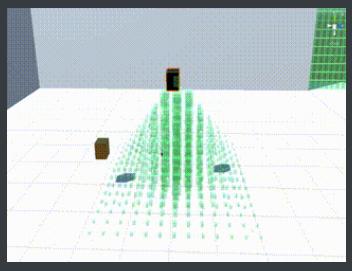
```
 public Vector3 positionInVoxel(Vector3 positionInWorld)

  {
    Vector3 voxelPosition = new Vector3(0.0f,0.0f,0.0f);
    positionInWorld -= gameObject.GetComponent<Transform>().position;
    voxelPosition.x = (int)(0.5f + (positionInWorld.x / voxelSize));
    voxelPosition.y = (int)(0.5f + (positionInWorld.y / voxelSize));
    voxelPosition.z = (int)(0.5f + (positionInWorld.z / voxelSize));
    return voxelPosition;
  }
  public Vector3 positionInWorld(Vector3 positionInVoxel)
  {
    return positionInVoxel*voxelSize+gameObject.GetComponent<Transform>
().position;
  }
```

Zadanie 1



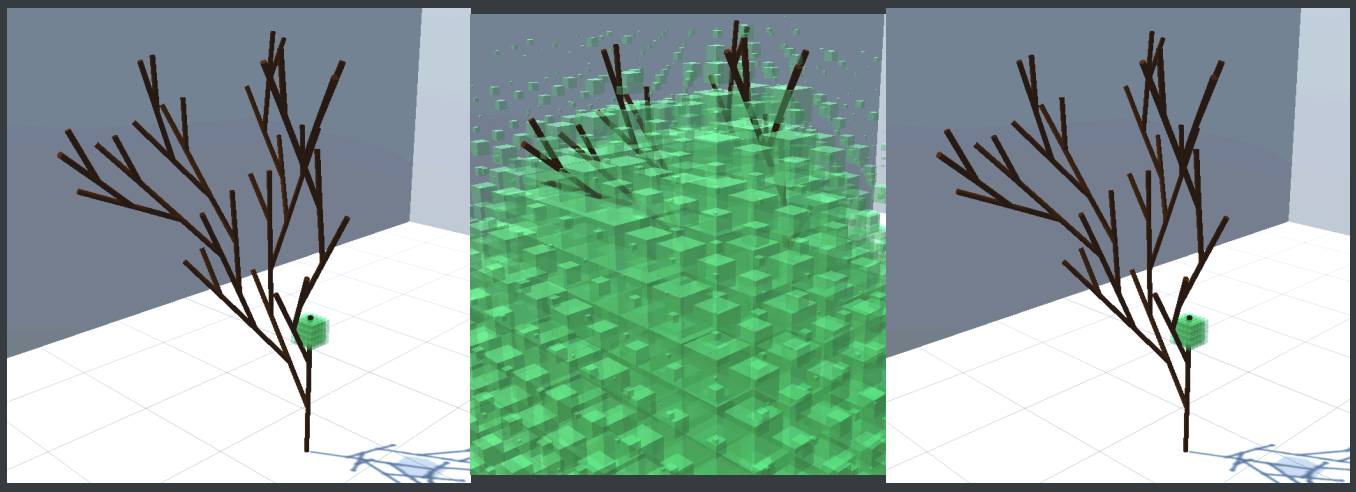
1. Otwórz Unity Project, wersję na te ćwiczenia. Następnie otwórz scenę "Pipe Model"
2. Do objektu shadowBox jest przypięty skrypt Cast Shadow. Otwórz ten skrypt.
3. Zmodyfikuj skrypt, tak aby ten obiekt rzucał cień w dół.
    1. Wykorzystaj funkcję środowiska (Environment)
    2. public void addShadow(Vector3 position, sbyte strength) // ujemna siła, aby usunąć

Zadanie 2

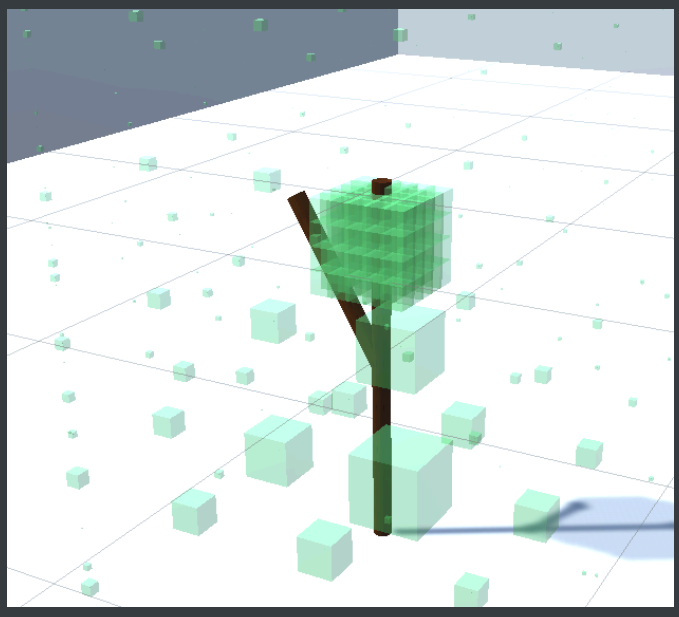


1. W objekcie inShadow jest skrypt receiveShadow
2. Zmodyfikuj skrypt - jak na ten objekt spadnie cień, to zmień materiał
    1. Wykorzystaj funkcję środowiska (Environment)
    2. public byte shadowStrength(Vector3 pos)

Zadanie 3



1. Wybierz objekt tree (001)
2. Jeśli nie jest ustawione, to ustaw odpowiednią ścieżkę (L-System Path) do *ShadowModel.txt*
3. Uruchom grę i przejdź kilka kroków (Load File na start i Evaluate - krok)
4. Gałęzie w cieniu nadal się rozwijają. Wylicz zacienienie do L-Systemu
    1. Skrypt **TurtleLSystemEnvironment**, funkcja **lightDirection** (20 linijka), od 38 linijki
    2. Zacienienie mierzymy w prostopadłościanie wokół obecnego elementu (2***lookForLightLength** x **lookForLightLength** x 2***lookForLightLength**)

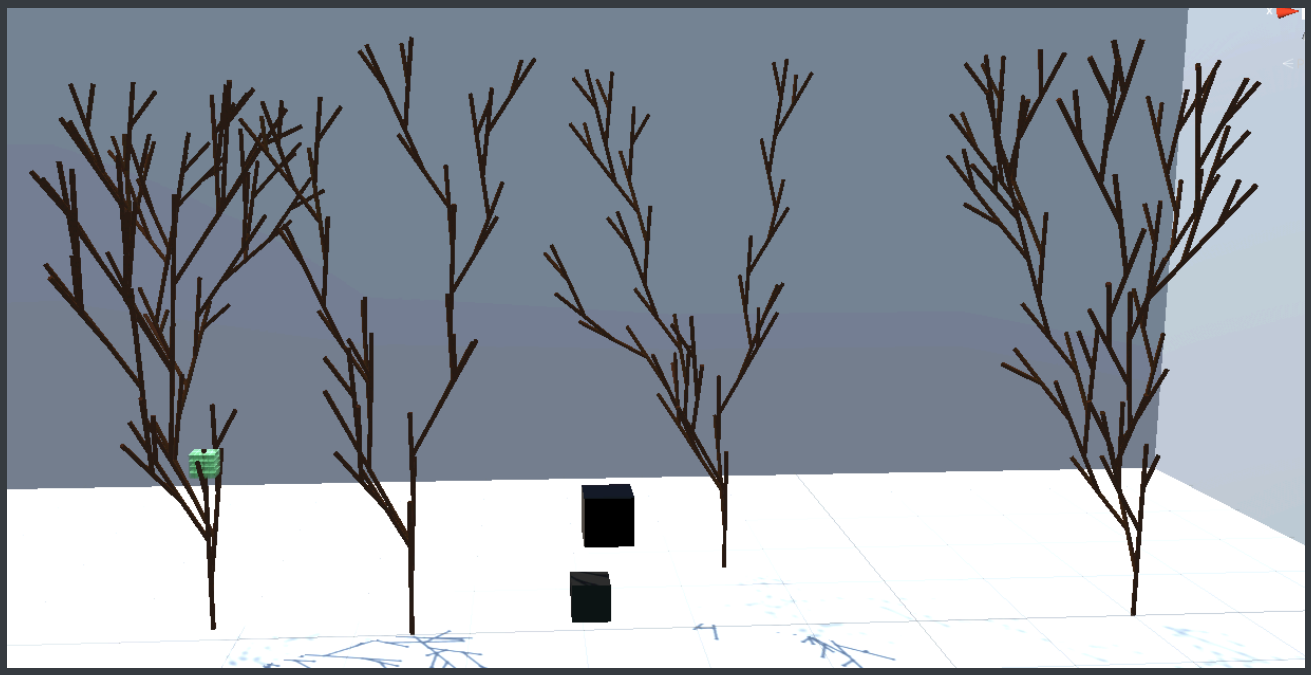


    3. Transformacja z obecną pozycją (**transformation***resultTransformation**)** funkcja transformacja.**ExtractPosition**() zwraca pozycję dla transformacji
5. Wyślij wartość zacienienia do L-Systemu

```
#ignore + - \ / ^ &
#axiom
S(0,0)
#rules
S(a,c) : c>=50 -> S(a+1,0)
S(a,c) : c<50 -> G\(90)[-S(0,0)]S(0,0)
```

1. 1. L-System będzie odczytywał jako cień drugą wartość - dla **S(a, c)** to będzie **c**
   2. **node.literal.values** jest tablicą z wartościami L-Systemu

Zadanie 4



1. Dodaj do objektu Environment więcej drzew (Trees) i zobacz jak drzewa wpływają na siebie nawzajem
2. Napisz skrypt, który utworzy 9 drzew (3x3) rozmieszczonych co równą odległość
3. Zobacz jak się w takim układzie rozwijają

Zadanie 5 – domowe

Wybierz zdjęcie dowolnego drzewa (każdy inne) i stwórz L-System podobny do tego drzewa, ale zależny od cienia

Do rozwiązania dodaj zdjęcie drzewa i zdjęcie odtworzonego drzewa. L-System nazwij imięnazwiskoShadow.