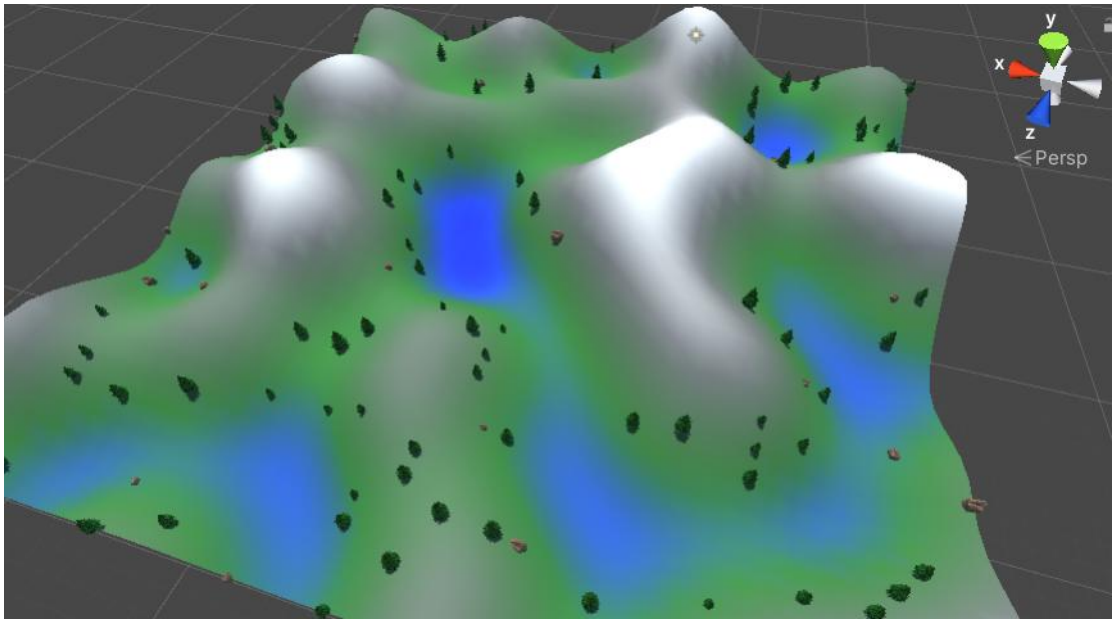
Daniel Ostrowski, Kamil Wolka

Lab2. Generowanie terenu




```
IEnumerator StworzKsztalt()
    {

        punkty = new Vector3[(xSize + 1) * (zSize + 1)];

        for(int index = 0, z = 0; z <= zSize; z++)
        {
            for (int x = 0; x <= xSize; x++)
            {
                float y = Mathf.PerlinNoise(x * .1f, z * .1f) * 10f;
                punkty[index] = new Vector3(x, y, z);
                index++;

                if (y > maxY)
                    maxY = y;
                if (y < minY)
                    minY = y;

                if(y > 3 && y < 5)
                {
                    rand = Random.Range(0, 10);
                    if (rand == 5)
                    {
                        int rand2 = Random.Range(10, 25);
                        int rand3 = Random.Range(0, 90);
                        GameObject obj = Instantiate(tree,
                        new Vector3(x, y, z),
                        Quaternion.identity) as GameObject;
                        obj.transform.localScale = new Vector3((float)rand2 /
100, (float)rand2 / 100, (float)rand2 / 100);
                        obj.transform.rotation = Quaternion.Euler(0, rand3, 0);
                    }
                    int rand4 = Random.Range(0, 25);
                    if (rand4 == 3)
                    {
```

```
                        int rand2 = Random.Range(10, 25);
                        int rand3 = Random.Range(0, 90);
                        GameObject obj = Instantiate(rock,
                        new Vector3(x, y, z),
                        Quaternion.identity) as GameObject;
                        obj.transform.localScale = new Vector3((float)rand2 /
1000, (float)rand2 / 1000, (float)rand2 / 1000);
                        obj.transform.rotation = Quaternion.Euler(0, rand3, 0);
                    }
                }

            }
        }
        print(minY);
        print(maxY);
        trojkaty = new int[xSize * zSize * 6];

        int vert = 0;
        int tris = 0;
        for (int z = 0; z < zSize; z++)
        {
            for (int x = 0; x < xSize; x++)
            {
                trojkaty[tris + 0] = vert + 0;
                trojkaty[tris + 1] = vert + xSize + 1;
                trojkaty[tris + 2] = vert + 1;
                trojkaty[tris + 3] = vert + 1;
                trojkaty[tris + 4] = vert + xSize + 1;
                trojkaty[tris + 5] = vert + xSize + 2;

                vert++;
                tris += 6;
                yield return new WaitForSeconds(.001f);
            }
            vert++;
        }

        kolory = new Color[punkty.Length];

        for (int index = 0, z = 0; z <= zSize; z++)
        {
            for (int x = 0; x <= xSize; x++)
            {
                float y = Mathf.InverseLerp(minY, maxY, punkty[index].y);
                kolory[index] = gradient.Evaluate(y);
                index++;
            }
        }
                                        }
```

Najpierw tworzony jest mesh, w którym ustawiana jest wysokość za pomocą funkcji szumów. Kolory są wybierane z gradientu w zależności od wysokości. Na końcu dodawane są modele drzew i kamienie o losowej skali, rotacji i pozycji. Pozycja jest wybrana z takiego zakresu, aby nie mogły się one pojawić na szczytach i w rzekach.