Procedurálně generovaná 2D plošinovka

Patrik Krulík
FIT – ČVUT
krulipat@fit.cvut.cz
03.01.2024

Úvod

Jako svoji semestrální práci jsem si vybral Procedurálně generovanou 2D plošinovou hru.

Hru jsem vytvářel pomocí knihovny pygame a mým cílem bylo vytvořit plošinovou hru, kde se mapa před začátkem hry vygeneruje sama a Vaším cílem bude za co nejrychlejší čas přeskákat vygenerovanou část.

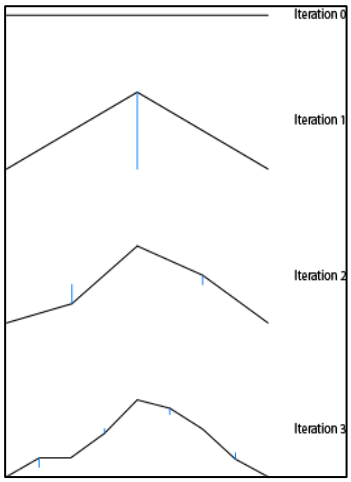
Pro generaci terénu jsem použil algoritmus Midpoint displacement, který mi vytvořil hodnoty pro 1D pole, které jsem poté přidal do 2D pole a vytvořil z něj mapu, po které hráč ve hře skáče. Dále jsem na prázdné plošiny přidal nějaké bodáky a bloby (nepřátelské slizy), aby hráč neměl průběh hry tak jednoduchý.

Použitý algoritmus – Midpoint displacement

Tento algoritmus funguje rekurzivně tak, že na začátku máme čáru, vezmeme její bod uprostřed o nějaký náhodný koeficient ho posuneme podle osy y, tímto nám vzniknou menší 2 čáry, které jsou oddělené prostředkem této původní čáry, algoritmus tedy zavoláme na tyto menší 2 čáry, dokud vzdálenost jednotlivé čáry nebude 1.[1]

Konkrétně pro náš případ:

Vytvoříme si 1D pole a zvolíme 2 body (začátek a konec pole), kterým přiřadíme nějakou navzájem lišící se hodnotu. V každé iteraci algoritmu se zvolí prostřední bod těchto dvou bodů, do něj přiřadíme průměr hodnot jeho sousedů, a ještě jeho hodnotu pře násobíme koeficientem vynásobeným náhodným číslem mezi hodnotami -1 a 1. Jelikož máme teď 2 prázdné části pole (jedna mezi začátkem a prostředkem pole a druhá mezi prostředkem pole a koncem pole), zavoláme algoritmus na tyto 2 pod části. Algoritmus voláme, dokud není zaplněno celé pole.



Obrázek 1 - ukázka algoritmu

Závěr

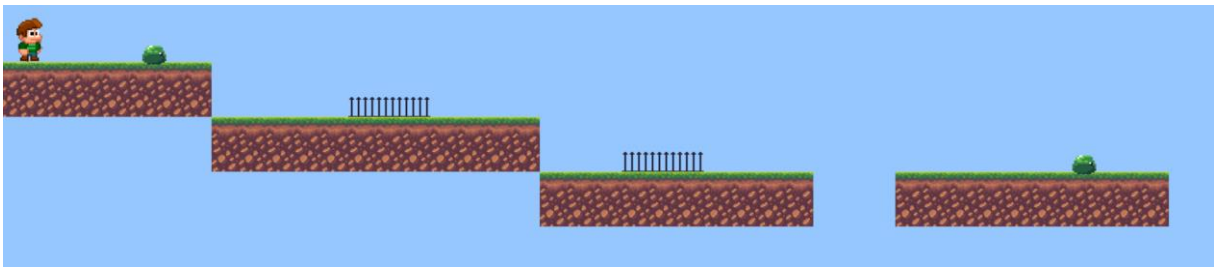
Algoritmus se mi podařilo správně napasovat na potřeby hry a na přidání bodáků a nepřátel jsem použil náhodu, takže každá vygenerovaná mapa vypadá vždy opravdu jinak.

Hra by šla rozšířit přidáním více nepřátel, nebo přidáním nějakých předmětů, které by hráči snižovali výsledný čas, nebo by se mohl udělat žebříček nejlepších časů a hráč by tak viděl, které časy by měl překonat.

[1] Aschenbach, N. (2014). 2D Fractal Terrain Generation. Retrieved January 3, 2024, from https://nick-aschenbach.github.io/blog/2014/07/06/2d-fractal-terrain/

Chtěl bych říct, že mé cíle se mi podařilo naplnit a jsem rád, že jsem si mohl vytvořit takový projekt na fakultě jako semestrální práci.

Obrázek 2 - použitý algoritmus přímo v programu hry



Obrázek 3 – ukázka hry, kde je vygenerovaná mapa odpovídající začátku předešlému obrázku

Použité zdroje pro algoritmus

1. Aschenbach, N. (2014). 2D Fractal Terrain Generation. Retrieved January 3, 2024, from https://nick-aschenbach.github.io/blog/2014/07/06/2d-fractal-terrain/

Použité zdroje při vytváření hry

2. https://www.youtube.com/watch?v=xxRhvyZXd8I&list=PLX5fBCkxJmm1fPSqgn9gyR3qih8yYLvMj&index=2
3. https://www.youtube.com/watch?v=d06aVDzOfV8&list=PLjcN1EyupaQmZw8C-q6a4Zekidxf8SUj3&index=1
4. https://www.youtube.com/playlist?list=PLjcN1EyupaQnHM1I9SmiXfbT6aG4ezUvu