VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

POČÍTAČOVÁ HRA ŽÁNRU EXPLORE/ADVENTURE V UNITY

COMPUTER GAME WITH THE EXPLORE/ADVENTURE GENRE IN UNITY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **MAREK PECHÁŇ**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. MICHAL VLNAS**
SUPERVISOR

BRNO 2025

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

160239

Ústav:	Ústav počítačové grafiky a multimédií (UPGM)
Student:	**Pecháň Marek**
Program:	Informační technologie
Název:	**Počítačová hra žánru explore/adventure v Unity**
Kategorie:	Počítačová grafika
Akademický rok:	2024/25

Zadání:

1. Nastudujte techniky herního vývoje v prostředí Unity.
2. Navrhněte hru žánru adventura/explore s vhodnými herními mechanikami.
3. Implementujte navrženou hru a demonstrujte ji na několika vhodných úrovních.
4. Zhodnoťte dosažené výsledky a též možnosti publikace hry.
5. Vytvořte demonstrační video.

Literatura:

- Gregory, Jason. *Game engine architecture*. crc Press, 2018. ISBN 1351974289, 9781351974288
- Bishop, Lars, et al. "Designing a PC game engine." IEEE Computer Graphics and Applications 18.1 (1998): 46-53.
- Adams, Ernest, and Joris Dormans. Game mechanics: advanced game design. New Riders, 2012. ISBN 0321820274, 9780321820273
- Koster, Raph. Theory of fun for game design. O'Reilly Media, Inc., 2013.
- Schell, Jesse. The Art of Game Design: A book of lenses. CRC press, 2008.
- Unity Learn. Unity, https://learn.unity.com/.

Při obhajobě semestrální části projektu je požadováno:
Body 1 a 2 a rozpracovaný bod 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce:	**Vlnas Michal, Ing.**
Vedoucí ústavu:	Černocký Jan, prof. Dr. Ing.
Datum zadání:	1.11.2024
Termín pro odevzdání:	14.5.2025
Datum schválení:	12.11.2024

Abstrakt

V tejto práci je navrhnutá a vytvorená video hra žánrov adventure a exploration v hernom engine Unity. Prvky hry sú implementované pomocou techník procedurálneho generovania a implementáciou hernej umelej inteligencie. Najdôležitejšou časťou hry je algoritmus kolaps vlnovej funkcie, ktorý procedurálne generuje úrovne za pomoci určenia sady políčok a obmedzení. Bezkontextové gramatiky určujú objekty, ktoré sa v miestnostiach úrovne nachádzajú a herná umelá inteligencia, ktorej interakcie s hráčom a prostredím je realizovaná pomocou stavového automatu. Výsledkom práce je video hra, ktorá využíva tieto techniky, aby hráčovi poskytla unikátne a dostatočne obtiažne úrovne.

Abstract

The goal of this thesis is to design and create a video game in the genres adventure and exploration within the game engine Unity. The games elements are implemented with techniques of procedural generation and utilization of video game artificial intelligence. The most important part of the game is the algorithm wave function collapse, which procedurally generates levels using sets of tiles and restrictions. Context-free grammars determine the objects, which fill the rooms of levels and video game artifical intelligence which interacts with the player and the environment. The result of this thesis is a video game, which provides the player with unique and sufficiently difficult levels.

Kľúčové slová

Vývoj video hier, procedurálne generovanie, herná umelá inteligencia

Keywords

Video game development, procedural generation, video game artificial intelligence

Citácia

PECHÁŇ, Marek. *Počítačová hra žánru explore/adventure v Unity.* Brno, 2025. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Michal Vlnas

Počítačová hra žánru explore/adventure v Unity

Prehlásenie

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Michala Vlnasa. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Marek Pecháň
12. mája 2025

Poďakovanie

Chcem srdečne poďakovať svojmu vedúcemu Ing. Michalovi Vlnasovi za jeho veľkú trpezlivosť pri práci so mnou a jeho skvelé rady, ktoré ma vždy nasmerovali správnym smerom k nájdení riešenia.

Obsah

Kapitola 1

Úvod

Vývoj video hier je oblasť obsahujúca veľké množstvo disciplín, čo obmedzuje možnosti malých tímov alebo samostatných vývojárov tvoriť zábavné video hry. Vďaka pokrokom v nástrojoch využitých pre vývoj, spôsobom realizácie procedurálneho generovania [5], hernej umelej inteligencii [15] a zvyšujúcemu sa záujmu o hry vytvorené nezávislými vývojármi, je táto náročná úloha omnoho dostupnejšia pre nezávislých vývojárov.

V tejto práci je vykonaný proces vývoja hry, od návrhu po realizáciu hry. Návrh upresňuje štruktúru úrovní a podmienky, ktoré musí úroveň a jej obsah spĺňať. Hra sa odohráva v 3D priestore a úrovne musia obsahovať objekty a predmety, s ktorými môže hráč aj nepriateľské prvky interagovať. Pri realizácií návrhu hry sú uplatňované techniky procedurálneho generovania. Techniky procedurálneho generovania umožňujú herným vývojárom ušetriť čas pomocou procedurálneho generovania obsahu hry. V tejto práci je predstavený upravený algoritmus kolaps vlnovej funkcie, ktorý je obvykle použitý v obmedzenej 2D mriežke. Upravený algoritmus je schopný generovať úrovne v 3D priestore pri teoreticky nekonečnej mriežke. Obmedzenia a limity sú definované s cieľom určiť ktorý upravený algoritmus musí nasledovať, aby sa predišlo nekonečnému behu algoritmu alebo generovaniu nelogických úrovní. Definovaná je aj bezkontextová gramatika, ktorá určuje typy miestností a akými objektmi sú plnené.

V návrhu hry sú upresnené aj obtiažnostné krivky, ktoré určujú obtiažnosť hry. Obtiažnosť je reprezentovaná pomocou postupne sa zvyšujúceho počtu nepriateľských prvkov. Hra obsahuje dané nepriateľské prvky a v jej návrhu je definovaná ich interakcia s hráčom a prostredím. Stavový automat je využitý ako jednoduchá implementácia správania nepriateľských prvkov, bez ktorých by hráč hru považoval za nudnú a hru by ukončil. Stavový automat určuje pohyb po navigačnej sieti a interakcie s hráčom a s prostredím. Pohyb nepriateľských prvkov je sprostredkovaný navigačnou sieťou určujúcou plochu, po ktorej sa nepriateľské prvky môžu pohybovať, aby sa vyhli prekážkam.

Kombináciou týchto prvkov je vytvorený logicky súvislý zážitok zodpovedajúci stanovenému návrhu hry v podobe rôznych úrovní, ktoré môže hráč prechádzať a zbierať z nich predmety. Popritom sa musí vyhýbať rôznym nepriateľským prvkom a pasciam.

Kapitola 2

Techniky herného vývoja

Cieľom tejto kapitoly je vysvetliť problematiku vývoja hier a techniky, ktoré sú využívané hernými vývojármi. Sú tu popísané herné enginy [14] a ich úloha pri vývoji hier. Ďalej sú vysvetlené rôzne metódy procedurálneho generovania [5] úrovní hier a spôsoby implementácie hernej umelej inteligencie. Nakoniec sú analyzované podobné riešenia, hlavne ich herné systémy a spôsob motivácie hráčov.

2.1 Úvod do vývoja hier

Herný vývoj je komplexnejší, ako sa môže na prvý pohľad zdať. Je to kombinácia viacerých odlišných disciplín: softvérové inžinierstvo, skladba hudby, 3D a 2D modelovanie, tvorba textúr, herný dizajn, testovanie a mnoho ďalších. Pri vývoji je nutné venovať pozornosť správnej kompozícií modelov, programových skriptov, textúr, vizuálnych a zvukových efektov, s cieľom umožniť hráčovi ponoriť sa do hry bez prerušenia. Vývoj hry je často členený na nasledujúce fázy, ktoré sú definované v závislosti od herného vývojového štúdia a dĺžka jednotlivých fáz je závislá od samotného návrhu hry:

- **Predprodukcia** – Dizajn hry, plánovaný žáner, herné systémy, úrovne a pod. sú tvorené zároveň s vykonávaným prieskumom trhu a konkurencie.
- **Prototypovanie (Pre-Alfa)** – Takzvaný prototyp, čo je skorá verzia navrhnutej hry je vytvorená za veľmi krátky čas a má demonštrovať herné systémy. Prototyp slúži na validáciu návrhu. Ak je validácia neúspešná, znova sa vykonáva predprodukcia. Pri úspešnej validácií sa definujú špecifikácie projektu a zostaví sa sa tím.
- **Alfa** – Všetky herné systémy sú implementované ale nie otestované. Hre v tejto fáze chýba obsah. Hra sa považuje za minimálny životaschopný produkt.
- **Beta** – Všetky herné systémy sú implementované, ale vyžadujú ladenie. Hra má všetok plánovaný obsah. Hra sa môže aj predčasne vydať na otestovanie verejnosťou.
- **Vydanie** – Hra je dokončená a pripravená na publikovanie. Pred publikovaním sa pre hru vedie reklamná kampaň aby bola verejnosť oboznámená o detailoch predaja, dátume vydania a herných systémoch.
- **Po vydaní** – Podľa typu hry môže ale nemusí byť podporovaná aj po vydaní novým obsahom vo forme stiahnuteľného obsahu, v angličtine volaného downloadable content a ďalej skracovaného na „DLC“, rozšírením alebo dodatočným ladením vo forme patchov.

2.2 Herné enginy

Definícia herného enginu je nasledovná: „*Herný engine je softvérový framework vytvorený pre prototypovanie a vývoj video hier. Vývojári ich používajú na tvorbu hier pre konzoly, mobilné zariadenia a osobné počítače. Herné enginy často zahŕňajú podporné programy, knižnice, interpretovaný jazyk a pod., ktoré pomáhajú vyvinúť a prepojiť rôzne komponenty hry.*“ Táto definícia bola parafrázovaná z [14].

Jedna z najdôležitejších častí je voľba alebo implementácia herného enginu [14], ktorý najlepšie vyhovuje návrhu vyvíjanej hry. Vývojári majú na výber s herných enginov [14], ako napríklad:

- **Unity engine** je cross platformový herný engine, ktorý pre scripting využíva programátorský jazyk C#.
- **Godot** je open source herný engine ktorý pre scriptovanie používa vlastný programovací jazyk GdScript, ktorý je syntakticky podobný programovaciemu jazyku Python.
- **Unreal engine** je herný engine, ktorý pre scripting využíva programovací jazyk C++ zameraný na vývoj 3D hier.
- **Game maker** je séria cross-platformových herných enginov, ktoré sú zamerané na začiatočníkov.
- **RPG maker** je 2D herný engine, ktorý je zameraný na začiatočníkov, ktorí chcú vytvoriť hry žánru RPG.

Pre simuláciu fyzikálnych zákonov je v herných enginoch využitý engine fyziky [8]. Tento engine môže byť buď vytvorený špecificky pre herný engine alebo vytvorený treťou stranou.

„*Engine fyziky je všeobecná kalkulačka fyzikálnych výpočtov, teda jeho implementácia nezodpovedá špecifickým situáciám, ktoré simuluje. Engine fyziky implementuje fyzikálne rovnice a zákony čo najbližšie k realite a herný engine ich využíva na účel vykonania simulácie fyziky v tvorenej hre.*“ Táto definícia bola parafrázovaná z [8]. Najdôležitejšia problematika, ktorú v kontexte hier enginy fyziky riešia sú kolízie objektov a ich správanie pri kolíziách. Príklady enginov fyziky sú: Bullet, PhysX, Box2D, Havoc, Vortex a pod.

Požiadavky na engine fyziky závisia na špecifikáciách tvorenej hry. Pre hru je realistická simulácia fyziky vyžadovaná len ojedinele. Často je realistická simulácia fyziky nežiadúca z dôvodov nárokov na hardvér, zníženie hrateľnosti alebo ochudobnenie zábavnosti ako dôsledok zriedenia herného zážitku implementáciou fyziky v prehnanom detaile.

2.2.1 Unity engine

Unity engine je flexibilný herný engine, ktorý zahŕňa široký sortiment nástrojov mierených na bezproblémové spojenie všetkých aspektov herného vývoja. Je založený na princípe objektovo orientovaného programovania. „*Objekt v objektovo orientovaním programovaním je dátová štruktúra obsahujúca dáta (stav) a metódy (správanie).*“ Táto definícia bola parafrázovaná z [3]. Kompatibilita s programami Maya a Blender zaručuje prepojenie 3D modelovania a textúrovania. Pre hudbu a zvukové efekty je možné použiť podporované súborové formáty .mp3, .ogg, .wav, .flac, .aiff/.aif, .mod, .it, .s3m a .xm. Na skriptovanie je použitý jazyk C#, pre ktorý poskytuje knižnicu s dokumentáciou. Pomocou danej knižnice je možné pristupovať k objektom, využívať ich metódy, vytvárať nové inštancie objektov a vystavovať dáta objektov na manipuláciu vývojárom a inými skriptami. Výhody unity enginu sú:

- **Multiplatformová tvorba** – Hra môže byť zostavená pre viacero platforiem ako je web, desktop alebo herné konzole bez potreby hru alebo skripty upravovať pre dodatočné platformy.
- **Unity asset store** – Poskytovaný je najpopulárnejší obchod s hernými assetmi ako zvuky, hudba, textúry, modely, skripty, úrovne a pod. Tento obchod je integrovaný s unity enginom tak že zakúpené assety môžu byť priamo stiahnuté pomocou unity engine.

Hra v engine [14] Unity obsahuje jednu alebo viacej scén. Scény obsahujú rôzne logicky rozdelené časti hry ako menu, samotné úrovne, prestávky medzi úrovňami a pod. Scéna je vždy koreňový objekt hierarchie [3] a všetky objekty, ktoré do nej patria, sú herným enginom [14] Unity definované ako jej deti [3]. V tomto zmysle dieťa objektu je objekt, ktorý je pod rodiča zaradený v hierarchii. Dieťa od svojho rodiča nededí [10] nič okrem pozície a rotácie, ktorá je k rodičovi relatívna. Ak je rodič de-aktivovaný alebo vymazaný, všetky jeho deti sú tiež de-aktivované alebo vymazané. Následujúce objekty poskytuje engine [14] Unity, tento zoznam nie je úplný a objekty pochádzajú z :

- **Cinemachine** – Vývojárom je k dispozícií zabudovaná kamera, s ktorou je možné pokročilo manipulovať bez písania kódu.
- **Osvetlenie** – Vývojárom je poskytnuté upraviteľné osvetlenie. Upravné môžu byť farba a intenzita.
- **3D objekty** – Vývojárom sú poskytnuté rôzne jednoduché 3D tvary ako kocky a gule. Môžu byť upravené zmenou veľkosti, rotácie, materiálu a pod.
- **Prvky užívateľského rozhrania** – Vývojárom sú poskytnuté tlačidlá, text a objekty obsahujúce obrázky alebo textúry, ktoré sú vždy v popredí.

Vzhľad a správanie objektov [3] môže byť upravené pomocou takzvaných komponentov. Komponenty môžu byť napríklad:

- **Rigidbody** – Objekt obsahujúci tento komponent, je ovplyvňovaný fyzikálnymi zákonmi.
- **Collider** – Komponentom sú umožňované kolízie pričom Collider je možné nastaviť či ovplyvňuje objekty s komponentom Rigidbody alebo nie.
- **Skript C#** – Každý vytvorený skript môže upraviť správanie ľubovolných objektov. Priebeh skriptu nie je zdieľaný medzi objektmi, ktoré skript obsahujú ako komponent.

2.3 Procedurálne generovanie

„*Procedurálne generovanie v kontexte herného vývoja je pseudonáhodné algoritmické generovanie obsahu hry. Vytváranie obsahu pomocou procedurálneho generovania je z vývojárskeho pohľadu časovo veľmi úsporné pri tvorení úrovní, predmetov, nepriateľov, a pod.*“ Táto definícia je parafrázovaný z [5]. Ak je využité správne, hráčov zážitok z hry je obohatený a čas strávený hráčom v hre sa tým predlžuje. Nesprávne využité procedurálne generovanie spôsobí opak. V tejto podkapitole sú vysvetlené niektoré spôsoby procedurálneho generovania úrovní:

- **Generátory náhodných čísel** [6] – Generátory náhodných čísel sú metódy, ktoré počítače využívajú na simuláciu náhodnosti.
- **Gramatiky** [7] – Gramatiky [7] sú súbor pravidiel, ktorých výstup je reťazec znakov.
- **Kolaps vlnovej funkcie** [1] – Kolaps vlnovej funkcie je algoritmus pre generovanie úrovní pomocou mriežky. Je potrebné definovať políčka a obmedzenia, ktoré určujú výsledok algoritmu.

Metódy procedurálneho generovania nemusia byť využité samostatne. Je možné kombinovať viacero metód v jednej úrovni, ak je to žiadúce.

2.3.1 Generátory náhodných čísel

Generátory náhodných čísel [6] je možné rozdeliť na dva typy:

- „*Pseudonáhodné generátory náhodných čísel využívajú hodnotu semienka, aby vyprodukovali nepravidelné ale stále deterministické sekvencie čísel. Pseudonáhodný generátor je možné upraviť tak, aby generoval nedeterministické sekvencie čísel pri uplatnení prirodzenej udalosti ako hodnoty semienka.*“ Definícia je parafrázovaná z [6].
- „*Skutočné generátory náhodných čísel závisia na prirodzených udalostiach ako napríklad rozpad rádioaktívneho žiarenia, termálny hluk, nízko-úrovňová hardvérová aktivita a pod. aby vygeneroval skutočne nedeterministické náhodné čisla.*“ Definícia je parafrázovaná z [6].

Náhodné generátory čísel sú dôležitou časťou procedurálneho generovania [5] v hrách. Sekvencie náhodných čísel, ktoré sú vygenerované generátormi náhodných čísel [6] určujú výsledky mnohých metód procedurálneho generovania [5]. Pre využitie v hrách je vhodnejšie využiť pseudonáhodný generátor čísel [6] oproti skutočnému generátoru náhodných čísel [6]. Pseudonáhodný generátor čísel sa pri ladení hier, v ktorých je použitý, využíva jednoduchšie, v porovnaní so skutočným generátorom náhodných čísel [6]. Zabudovaný pseudonáhodný generátor čísel je zahrnutý v hernom engine Unity. Ak semienko nie je nastavené, zabudovaný pseudonáhodný generátor čísel semienko určí podľa systémového času zariadenia, na ktorom je spustený.

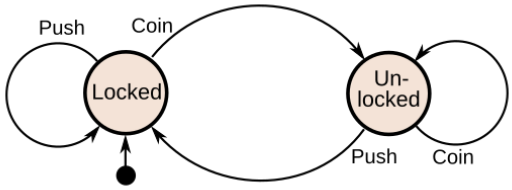
2.3.2 Gramatiky

Gramatiky [7] sú množiny, ktoré určujú postupnosť znakov. „*Všeobecná definícia gramatík [7] je nasledovná:*

$$G = (N, T, P, S)$$

V ktorej

- *N je množina neterminálnych znakov.*
- *T je množina terminálnych znakov.*
- *P je konečná relácia, jej vzťahy sú tiež nazývané produkcie.*
- *S je počiatočný neterminálny znak.*



Obr. 2.1: Grafické znázornenie stavového automatu [2], reprezentujúci stavy turniketu. Prevzaté z: https://en.wikipedia.org/wiki/Finite-state_machine

Produkcie v množine P sú značené $(A, x) \in P$ a často zapisované následovne:

$$A \to x$$

“ Táto definícia bola parafrázovaná z [7]. Produkcie iteračným spôsobom menia neterminálne znaky na ľavej strane zápisu na terminálne alebo neterminálne znaky na pravej strane zápisu, pričom cieľom je dosiahnuť výsledný reťazec obsahujúci výlučne terminálne znaky.

Gramatiky nie sú výhradne určené na využitie pri procedurálnom generovaní [5] hier, to ale neznamená že nie sú silným nástrojom pre tento účel. Hlavne môžu byť využité následovné gramatiky [7]:

- **Bezkontextová gramatika** [7] – Gramatika [7], v ktorej na ľavej strane produkcie [7] sa môže nachádzať len jeden neterminálny znak a na pravej strane produkcie môžu byť terminálne aj neterminálne znaky.
- **Neobmedzená gramatika** [7] – Gramatika [7], v ktorej na ľavej strane produkcie [7] sa môže nachádzať ľubovolná kombinácia terminálnych a neterminálnych symbolov a na pravej strane produkcie môžu byť terminálne aj neterminálne znaky.

Gramatiky [7] môžu obsahovať viac ako jednu produkciu s rovnakou ľavou stranou. V tomto prípade je použitá produkcia vybraná náhodne a výsledná kolekcia znakov je iná pri každom použití gramatiky [7].

2.3.3 Kolaps vlnovej funkcie

„*Kolaps vlnovej funkcie je algoritmus pre procedurálne generovanie textúr, objektov, máp a pod. pomocou mriežky. Na využitie kolapsu vlnovej funkcie musia byť definované sady políčok a obmedzení. Pomocou obmedzení sa predchádza tomu, aby sa určité políčka nachádzali vedľa iných políčok.*“ Táto definícia je parafrázovaná z [1]. Algoritmus kolaps vlnovej funkcie [1] je ďalej nazývaný WFC. Jednoduchá implementácia algoritmu WFC [1] je nasledovná: „

1. *Inicializácie začiatočného políčka a tabuliek obmedzení.*
2. *Výber susediacej bunky pre kolaps.*
3. *Výber políčka (často náhodne), do ktorého sa bunka zrúti.*
4. *Vykonanie kolapsu a prenos obmedzenia na susediace políčka.*
5. *Opakovanie od kroku 2. dovtedy kým všetky bunky nie sú skolabované.*

“ Tento algoritmus bol parafrázovaný z [1].

WFC [1] je najlepšie využité pri generovaní štruktúrovaných máp. Líšiaci sa od iných metód procedurálneho generovania, je pri WFC [1] predchádzané situáciám, kde sú vytvorené nežiadúce výsledky.

2.4 Nepriateľská umelá inteligencia

Pojem umelá inteligencia, ďalej AI, má v kontexte hier odlišný význam ako vo všeobecnosti. AI [15] v hrách predstavuje systém na ovládanie správania nehrateľných postáv, ďalej nazývaných NPC. Pod správaním NPC sa rozumie pathfinding, rozhodovanie a pod. Herné AI [15] môže byť použité aj pre režírovanie obtiažnosti hry. V tejto podkapitole sú stručne popísané stavové automaty [2] a ich využitie v kontexte hier. Ďalej sú stručne definované navigačné siete [4] a ich použitie v hernom engine [14] Unity.

2.4.1 Stavové automaty

„*Stavové automaty sú konečné systémy zostavené z viacerých stavov a vstupov alebo kondícií. Má počiatočný stav a v danom okamihu je práve v jednom stave. Stavový automat vníma vstupy diskrétne a synchronizovane s hodinami. Popisujú sa grafom, ktorý sa skladá z uzlov a hrán. Uzly reprezentujú stavy a hrany reprezentujú vstupy, ktoré spôsobia zmenu stavu*“. Táto definícia je parafrázovaná z [2]. Obrázok 2.1 je ukážka stavového automatu.

Využitie stavových automatov [2] pre hernú AI [15] má výhody aj nevýhody. Jednou z výhod je jednoduchosť návrhu a implementácie pri jednoduchších AI [15]. Naopak pri požiadavke na postupnú rozšíriteľnosť hernej AI [15] do zložitejšej podoby sú stavové automaty [2] nežiaduce. Pri rozširovaní herného AI [15] vo forme stavových automatov sa musí dbať na všetky nové stavy a vstupy, ktoré spôsobia stavový prechod z a do daného stavu. V hernom engine [14] Unity musí byť stavový automat [2] realizovaný manuálne, pretože nie je zahrnutý do sady nástrojov v Unity.

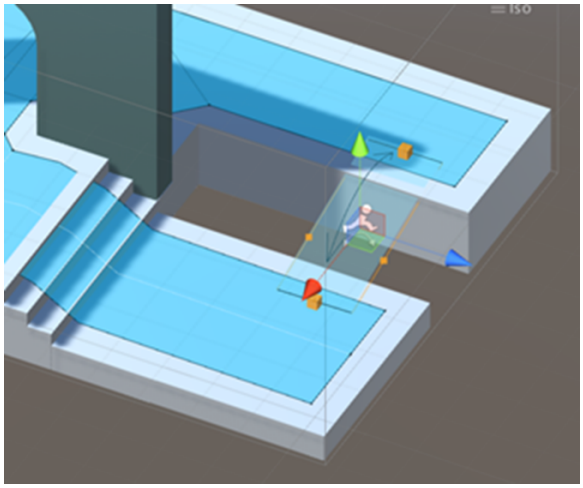
2.4.2 Navigačné siete

„*Navigačné siete sú reprezentáciou povrchov herného sveta, na ktorých je možné chodiť. Polygóny v navigačnej sieti sú spojené takzvanými prepojeniami, ktoré určujú možnosť agentov chodiť, skákať, urobiť krok po schodoch a pod.*“ Táto definícia bola parafrázovaná z [4].

Unity poskytuje navigačné siete [4] ako komponenty alebo ako súčasť editoru. K navigačným sieťam [4] je poskytnutý aj AI [15] agent, ktorý je schopný sieť prechádzať. Pre sieť musia byť definované objekty [3], ktoré má navigačná sieť považovať za nepriechodné. Navigačná sieť v Unity je zobrazená na obrázku 2.2, kde je možné vidieť ako sa agent môže po úrovni hýbať. Rôzne rozloženie procedurálne generovaných [5] úrovní robí navigačné siete [4] ideálnou voľbou pre realizáciu navigácie v hrách, ktoré procedurálne generujú [5] svoje úrovne.

2.5 Významné podobné riešenia

Existuje nespočetne veľa hier, ktoré patria do žánrov adventure [11] a exploration [11]. V tejto podkapitole sú popísané významné príklady hier vzhľadom na ich herné systémy, ktoré ich do týchto žánrov zaraďujú. Nakoniec sú podrobnejšie analyzované dve hry patriace



Obr. 2.2: Vizuálne zobrazená navigačná sieť [4] v prostredí Unity. Modrá oblasť je prechodná. Panáčik reprezentuje skokové prepojenie. Prevzaté z: https://docs.unity3d.com/560/Documentation/Manual/class-NavMeshLink.html

do žánrov adventure [11] a exploration [11] z hľadiska ich herných systémov. Žánre sú definované v podkapitole 3.1.

2.5.1 Významné hry

Medzi významné hry zapadajúce do žánrov adventure/exploration patria v neurčitom poradí:

- **Baldur's gate 3** – Obsahuje viacero hrateľných postáv s rôznzymi schopnosťami, ktoré hráčom nielen umožňujú poraziť nepriateľské NPC ale aj kreatívny pohyb po teréne. Takéto kreatívne myslenie je odmenené skrytými predmetmi, ktoré hráčovi poskytujú dodatočné schopnosti.
- **Crypt of the necrodancer** – Obsahuje procedurálne generovanie úrovní, predmetov a nepriateľských NPC. Schopnosti a predmety sú dôležité pre upravovanie a ničenie stien úrovní, za ktorými sa často skrývajú predmety.
- **Dark hours** – Obsahuje procedurálne generovanie úrovní pomocou viazania miestností a chodieb na seba. Každá miestnosť obsahuje cenné a kľúčové predmety, ale nie všetky miestnosti sú prístupné. Hráči sú podporovaní v prieskume miestností, aby našli čo najviac cenných predmetov.

2.5.2 Podobné riešenia

Ako podobné riešenie sa rozumie hra, ktorá nielen zapadá do žánrov adventure/exploration ale sú to hlavné žánre, v rámci ktorých je navhrnutá väčšina ich herných mechaník. Ukážky hier sú na obrázku 2.3.

Sea of thieves

Sea of thieves je hra o pirátoch, ktorá sa viac nakláňa k žánru exploration ako adventure. Cieľ hry je určený hráčmi, môžu sem patriť ciele ako získať zlato buď cez takzvané plavby

alebo ukradnutie pokladov od ostatných hráčov. Hra sa odohráva na statickej, ručne vyrobenej mape. Plavby sú procedurálne vygenerované úlohy, ktoré vedú hráčov k pokladom, nepriateľom a pod. Hráči sú motivovaní plavbami, aby preskúmavali svet a hra skonči len vtedy, keď je hráč spokojný s pokrokom k jeho vlastným definovaným cieľom.

Forewarned

Forewarned je hra, ktorá sa svojimi hernými systémami viac prikláňa k žánru adventure, v porovnaní s hrou Sea of Thieves. Ciele hráčov sú jasne definované. V tomto prípade cieľom je dedukovať aká príšera v úrovni pôsobí a ukradnúť jej poklad. Prieskum mapy slúži na splnenie týchto cieľov a získanie dodatočných pokladov. Keď je cieľ splnený a hráči útoky monštra prežili, hráči sú motivovaní mapu opustiť z dôvodu značne zvýšenej obtiažnosti.

Obr. 2.3: Ukážka hier Sea of Thieves a Forewarned resp. Prevzaté z https://rarethief.com/how-to-bury-treasure-create-treasure-stash-maps-and-steal-maps-in-sea-of-thieves/ a https://store.steampowered.com/app/1562420/FOREWARNED/

Kapitola 3

Koncept hry

Účelom kapitoly je detailný popis návrhu riešenia. Začína popisom herných žánrov adventure a exploration. Pokračuje popisom žánrov a podmienkami, ktoré musí hra spĺňať, aby sa zaradila do oboch žánrov.

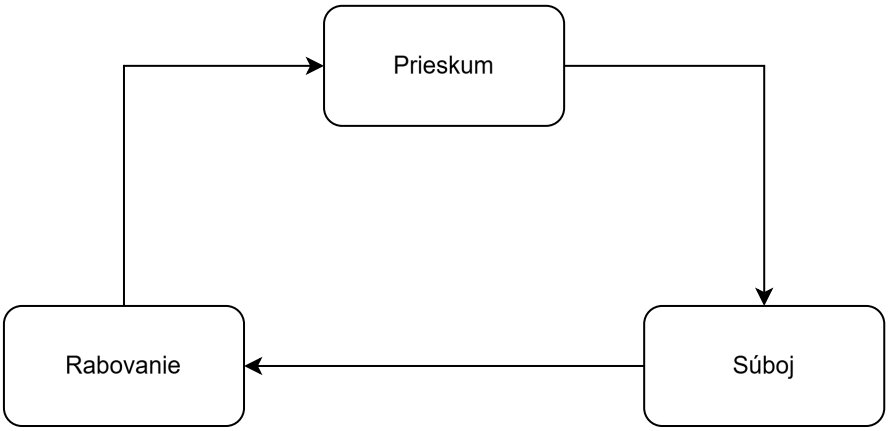
Pre všetky prvky, z ktorých sa hra skladá, je uvedené vysvetlenie ako sa prvky ovplyvňujú navzájom, ako prvky ovplyvňujú zážitok hráčov, odôvodnenie ich inklúzie v návrhu a vysvetlenie riešenia problémov.

3.1 Žáner a zaradenie hry

Keďže očakávania hráčov od herných žánrov sa neustále menia, presná definícia žánrov je veľmi obtiažna. Za premenou definície herných žánrov stojí aj neustála inovácia herných systémov. Hráči používajú pre klasifikáciu hier predošlé príklady hier, ktoré v danom žánri už hrali, v kombinácii s ich vlastnou interpretáciou daného žánra. To znamená, že ak je vydaná nová „žáner definujúca hra“ s inovovanými hernými systémami a bude použitá veľkým počtom hráčov, je vhodným kandidátom na porovnanie pri rozhodnutí, či iná hra patrí do daného žánra. Žánre adventure, exploration a action sú najčastejšie definované nasledovne:

- **Adventure** [11] – „*Žáner je definovaný zdieľanými prvkami ako sú: prieskum, zbieranie predmetov, manipulácia objektov, riešenie hádaniek a znížené zameranie na súboj a konflikt.*“ Táto definícia je parafrázovaná z [11].
- **Exploration** [11] – „*Je žáner, ktorý je pridelený hrám, ktoré kladú dôraz najmä na prieskum úrovní alebo herného sveta.*“ Táto definícia je parafrázovaná z [11].
- **Action** [11] –„*Je veľmi rozšírený žáner, ktorý je definovaný skúškou zručností v koordinácii rúk a očí a času reakcie pod tlakom. Herné systémy sú často jednoduchšie z dôvodu zlepšenia daných zručností pri nižšom príjme informácií.*“ Táto definícia je parafrázovaná z [11].

Kombináciou Adventure/Exploration je teda žáner, kde v danej hre existuje väčšina zdieľaných prvkov zo žánru Adventure s hlavným dôrazom na prieskum úrovní alebo sveta. Hra definovaná následujúcim návrhom je teda hra, ktorá tieto podmienky spĺňa, čím do daných žánrov zapadá. Dodatočne zapadá do žánra action, čo je odôvodnené jej inklúziou nepriateľských NPC, ktoré s hráčom bojujú v reálnom čase. Hra je teda zaradená do žánrov Action, Adventure a Exploration.



Obr. 3.1: Jadrový cyklus [13] hry Atlas of Fortune.

3.2 Návrh hry

V tejto podkapitole je celkovo popísaný návrh hry, jej individuálnych prvkov, ich interakcie medzi sebou a ako dané prvky prispievajú k zážitku hráča. Hra je zameraná na platformu PC, presnejšie na vydanie na službe Steam. Steam je služba poskytujúca digitálnu distribúciu a predaj hier, vyvinutá spoločnosťou Valve. Pomocou služby Steam je vývojárom odobraná zodpovednosť za doručenie hry na osobné zariadenia. Za poskytnutie služby vývojárom je stanovený poplatok a spoločnosť Valve si zoberie podiel 30%.

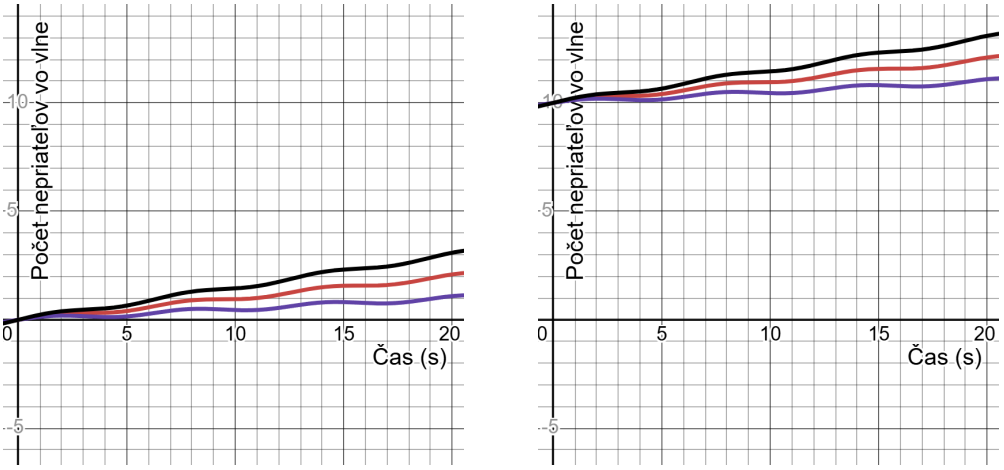
Názov hry je „Atlas of Fortune“. Hra je pomenovaná po predmete, vďaka ktorému sa hlavná ovládateľná postava ocitla v žalári zamorenom príšerami. Atlas of Fortune je hra založená na prieskume procedurálne vygenerovaného žalára plného nebezpečných príšer. Hráči sa vžijú do roly mladého draka, ktorý hľadá poklady do svojej zbierky. Hráčom sú poskytnuté schopnosti draka, pomocou ktorých vie nájsť poklady, vyhýbať sa pasciam a zabíjať príšery. Hráči sú v časovej tiesni a musia urobiť rozhodnutie, či budú pokračovať prieskumom pre získanie ďalších pokladov alebo útekom zo žalára. Hra je primárne zameraná na mladšie publikum a hráčov, ktorí majú radšej zložitejšie akčné hry a taktické rozhodnutia. Hra obsahuje nasledujúce jedinečné prvky:

- Nepriatelia, ktorí interagujú s prostredím.
- Časová tieseň vo forme zvyšujúcej sa obtiažnosti, nie časovača.
- Generovanie žalárov so semienkom, ktoré môžu hráči zdielať a súperiť o vyššie skóre.

Návrh sa skladá z takzvaných herných cyklov [13], ktoré sú definované ako systémy vzájomne sa ovplyvňujúcich častí. Tieto časti sa ovplyvňujú v cykle a sú hlavným nástrojom vytvárania interaktívneho zážitku.

Základ herného návrhu je takzvaný jadrový cyklus [13]. „*Jadrový cyklus definuje akými činnosťami sa hráč najčastejšie zaneprázdňuje. Účel je sformulovaný hráčom, ktorý potom na základe účelu vykoná činnosť. Systém je ovplyvnený činnosťou a reaguje poskytnutím spätnej väzby hráčovi, ktorá ho informuje o jeho vplyve.*“ Táto definícia herného cyklu bola parafrázovaná z [13] Na obrázku 3.1 je zjednodušený následujúci cyklus:

- Hráč začína s účelom **preskúmať** žalár.
- Pasce a nepriatelia sú nájdené hráčom, ktorý s nimi musí **bojovať** alebo ich obísť.



Obr. 3.2: Krivka obtiažnosti hry Atlas of Fortune (naľavo $c = 0$ a napravo $c = 100000$). Čierna krivka má $d = 1,5$. Červená krivka má $d = 1,0$. Modrá krivka má $d = 0,5$.

- Po súboji sú nájdené a **zbierané** poklady a predmety, ktoré hráčovi umožňujú obnoviť svoje suroviny alebo zvyšovať svoje skóre.

3.2.1 Úrovne a ich obtiažnosť

Obtiažnosť je jeden z najdôležitejších prvkov zaručujúcich zábavnosť hry pre hráča. „*Zábava je charakterizovaná pocitom spokojnosti a rovnováhou, podielania sa na aktivitách intenzitou medzi stresujúcimi a nudnými.*“ Táto definícia bola parafrázovaná z [13]. „*Obtiažnosť je definovaná pomocou krivky obtiažnosti, ktorá vyjadruje ako sa obtiažnosť mení počas priebehu úrovne. Krivka by mala byť prispôsobená schopnostiam hráčov. Čím je skúsenejší hráč, tým vyššiu obtiažnosť by hra mala mať. Toto spôsobuje že čím dlhšie hráč hru hrá, tým sa krivky obtiažnosti zvyšujú.*“ Táto definícia bola parafrázovaná z [12]. Najjednoduchšie krivky obtiažnosti sú lineárne, ale tvorba krivky vyžaduje experimentovanie, znalosť vstupných veličín a ako ich hráč bude ovplyvňovať. Atlas of Fortune definuje svoju krivku obtiažnosti nasledovne:

$$e = \frac{\sin t + t \cdot d + \frac{c}{1000}}{10}$$

V danej krivke sú premenné definované takto:

- **e** – počet nepriateľov, ktorí sú pridaní do mapy po určitom časovom intervale.
- **t** – čas v sekundách, ktoré prebehli od začiatku úrovne.

$$t \in \langle 0, \inf \rangle$$

- **d** – číselné vyjadrenie výberu troch možných obtiažností: Easy, Normal, Hard. Resp. sú vyjadrené číselne ako 0,5; 1,0; 1,5. Strmosť krivky je určená týmto výberom.
- **c** – počet mincí, ktoré hráč doposiaľ z úrovne získal. Každých 10000 mincí natrvalo pridá nepriateľa do každej „vlny“ nepriateľov.

Krivka je graficky znázornená na obrázku 3.2. Atlas of Fortune upravuje obtiažnosť pridávaním dodatočných nepriateľov do mapy. Z tohto dôvodu je v krivke vyjadrená premenná e a nie premenná d.

3.2.2 Procedurálne generovanie úrovne

Atlas of Fortune obsahuje procedurálne generovanie [5] nielen úrovne, ale aj predmetov, pokladov, osvetlenia, nábytku, nepriateľov a pod. Procedurálny generátor musí dodržiavať určené pravidlá, ktoré sú v tejto podkapitole rozvinuté.

Zámer procedurálneho generovania je poskytnúť hráčom úplne novú úroveň pri každej hre a zároveň poskytuje hráčom možnosť ovplyvnenia úrovní pomocou semienka a obtiažnosti *d*. Základ procedurálneho generátora je algoritmus riešenia obmedzení WFC [1] alebo „kolaps vlnovej funkcie“, ktorý je doplnený náhodným generátorom čísel. Náhodný generátor [6] je v tomto prípade kategorizovaný ako pseudonáhodný generátor [6]. Toto správanie je pre hru Atlas of Fortune žiadúce, prináša viacero výhod, ako napríklad jednoduchšie ladenie a poskytnutie väčšej miery kontroly nad hrou pre hráčov z pohľadu generovania.

Mapa úrovne

Mapa úrovne je generovaná pomocou algoritmu WFC [1]. Algoritmus WFC [1] je popísaný v sekcií 2.3.3. Pre využitie WFC [1] musia byť splnené následujúce predpoklady:

- **Základnou štruktúrou mapy je mriežka** – Bez mriežky nie je algoritmus WFC [1] schopný korektne distribuovať dané obmedzenia na susedné bunky mriežky a nie je schopný dané bunky skolabovať na konkrétne políčka.
- **Definovaná sada políčok** – Je vyžadovaná z dôvodu výberu políčka počas behu algoritmu a je integrálna pre samotnú definíciu obmedzení, napr. ktoré políčka môžu a nemôžu susediť.
- **Definovaná sada obmedzení** – Sú vyžadované pre udržiavanie konzistencie, úplnosti a logiky vygenerovaných úrovní. Problémy ako východy von z mapy, do stien, možné nelogické generovanie susedov políčok a nikdy nekončiaci beh algoritmu pri nekonečnej mriežke sú spôsobené nesprávnou definíciou obmedzení.

Základom mapy je teoreticky nekonečná štvorcová mriežka a obmedzenia sú distribuované všetkým susedným bunkám, ktoré nie sú skolabované. Štvorcová mriežka je žiadúca aj pre generovanie žalára, pretože každé políčko je štvorec, ktorý je dokonale vhodný na generovanie štruktúr vytvorených človekom. Sada políčok je definovaná nasledovne:

- **Malé miestnosti** – Zmestia sa na jedno políčko, môžu mať $0-3$ východy, nepočítajúc vchod do miestnosti.
- **Chodby** – Šírka chodby je 1 bunka a dĺžka chodby sú 3 políčka, to znamená, že ak sú vybrané algoritmom WFC [1], musí sa stať kolaps viacerých buniek v jednej iterácii. Môžu obsahovať od $0-7$ východov pričom vchod do chodby do rozsahu nepatrí.
- **Veľké miestnosti** – Dĺžka aj šírka sú 2 bunky a podobne ako pre chodby, ak sú vybrané algoritmom WFC [1], musí sa stať kolaps viacerých buniek v jednej iterácii. Podobne ako chodby, môžu obsahovať $0-7$ východov, do ktorých sa nepočíta vchod do miestnosti.
- **Typy miestností** – Rozlišujú sa miestnosti piatich typov: sklad, zbrojnica, pokladnica, začínajúca miestnosť a únikový východ. Zároveň musia mať jeden z vyššie uvedených veľkostí. Hráč začína v počiatočnej miestnosti a ak je nájdený únikový východ, tak môže hru ukončiť.

Obmedzenia sú definované nasledovne:

- **Limit generovania** – Keďže štvorcová mriežka je rozmerovo nekonečná, algoritmus WFC [1] by bez obmedzenia nikdy neskončil. Daný limit počtu generovaných políčok je mäkký, algoritmu WFC [1] je dovolené pokračovať aj po prekročení limitu so zmenenými váhami určovania políčok. Keď je limit dosiahnutý, garantuje existenciu cesty od začiatku úrovne do konca. Limit je určený hráčom vybranou obtiažnosťou, podľa následujúcej jednoduchej rovnice:

$$l = 10 \cdot d$$

- **Obmedzenie typov miestnosti** – Hneď po sebe skolabované miestnosti nesmú byť toho istého typu. Diverzita vygenerovaného nábytku a predmetov je zaistená vďaka tomuto obmedzeniu.

- **Východy musia byť vyplnené** – Všetky východy existujúcich políčok musia mať na konci behu algoritmu WFC susediacu miestnosť s nadväzujúcim vchodom. Úplnosť vygenerovanej úrovne je garantovaná týmto obmedzením. V tomto prípade je úplnosť definovaná ako úroveň bez chýbajúcich políčok pri východoch z miestností.

S danými definovanými sadami políčok a obmedzení a pri využití teoreticky nekonečnej štvorcovej mriežky, generátor spĺňa podmienky pre generovanie úplných, logických a dokončiteľných úrovní. Generovanie miestností pomocou WFC obmedzení vytvára mapy, ktoré motivujú hráča mapu preskúmavať aj mimo cesty k úniku zo žalára. Týmto sa hra začlení do žánru „exploration“, ako je definovaný v sekcií 3.1.

Osvetlenie

Hra využíva nasledujúcu nerovnicu pre determinovanie či je daná miestnosť osvetlená alebo nie. Miestnosť je osvetlená ak je nerovnosť splnená, inak miestnosť nie je osvetlená:

$$r < 10 \cdot (N - n_l) - 10 \cdot (i)$$

Premenné v nerovnici nadobúdajú tieto hodnoty:

- **r** – náhodná hodnota vygenerovaná náhodným generátorom čísel [6].

$$r \in \langle 0, 100 \rangle$$

- **N** – počet existujúcich miestností, vygenerovaných WFC algoritmom [1].
- **nl** – počet osvetlených miestností, zvyšuje sa vždy keď je miestnosť osvetlená.
- **i** – počet pokusov o osvetlenie miestností.

Podľa vyhodnotenej nerovnice sa determinuje či je vybraná miestnosť osvetlená alebo nie. Osvetlenia majú dodatočný účel, všetky dvere, ktoré vedú k východu zo žalára, sú osvetlené modro. Bez modrého osvetlenia sa hráč ľahko stráca kvôli bludiskovému charakteru žalára. Modré aj bežné svetlá tak slúžia aj ako orientačné body pre hráča. Táto funkcia svetla bola inšpirovaná hrou Skyrim, kde sa objavuje pomocná schopnosť, ktorá hráčovi nakreslí cestu k jeho cieľu.

Nábytok, predmety a poklady

Nábytok slúži na spestrenie prostredia žalára a na umiestnenie užitočných predmetov pre hráča. Nábytok v miestnosti závisí na type danej miestnosti:

- **Skladisko** – Obsahuje sudy a elixíry.
- **Pokladnica** – Obsahuje truhlice a poklady.
- **Zbrojnica** – Obsahuje poličky a stojany na zbrane, ktoré držia elixíry a kľúče.

Nábytok slúži pre skúsených hráčov aj ako identifikátor predmetov, ktoré v danej miestnosti nájde. Ak sa hráč ocitne pod tlakom, môže sa takto rýchlo rozhodnúť či riziko a zisk z rabovania miestnosti sú dostatočne vyvážené pokrokom v úrovni. Hráč môže zbierať predmety a tie mu pri ich použití poskytujú rôzne výhody. Výskyt predmetov a nábytku je ovplyvnený obtiažnosťou podľa nasledujúcich rovníc, ktoré vyjadrujú pravdepodobnosť generovania predmetu alebo nábytku:

$$P_n = 1 - (0.3 \cdot d)$$

$$P_p = 1 - (0.5 \cdot d)$$

Pravdepodobnosť vygenerovania kusu nábytku je vyjadrená ako Pn a pravdepodobnosť vygenerovania predmetu je vyjadrená ako Pp. Obtiažnosť je vyjadrená premennou d. Existenciu každého predmetu určuje generátor náhodných čísel [6].

Predmety, ktoré môžu byť objavené hráčom a ich funkcie sú nasledovné:

- **Červený elixír** – Môže byť využitý na doplnenie životov hráča.
- **Modrý elixír** – Môže byť využitý na doplnenie suroviny „mana“ hráča.
- **Zelený elixír** – Môže byť využitý na doplnenie suroviny výdrže hráča.
- **Kľúč** – Môže byť využitý na otvorenie zamknutej truhlice, ktorá je garantovane plná pokladov.

Suroviny a schopnosti hráča sú definované v sekcií 3.4.2. Cieľom hráča sú v neposlednom rade poklady. Hráč je motivovaný zozbierať čo najviac mincí, aby mali vyššie skóre. Čím viac mincí hráč drží, tým viac nepriateľských kreatúr sa vloží do úrovne v pravidelných intervaloch. Mince sú nasledovné, líšia sa len v hodnote, ktorú hráč získa:

- **Medená minca** – Hodnota mince sa rovná 1.
- **Strieborná minca** – Hodnota mince sa rovná 10.
- **Zlatá minca** – Hodnota mince sa rovná 100.
- **Kôpka medených mincí** – Hodnota mince je náhodne vygenerovaná, náhodným číselným generátorom [6] od 1-15.
- **Kôpka strieborných mincí** – Hodnota mince je náhodne vygenerovaná náhodným číselným generátorom [6] od 1-15 a vynásobená 10.
- **Kôpka zlatých mincí** – Hodnota mince je náhodne vygenerovaná náhodným číselným generátorom [6] od 1-15 a vynásobená 100.

Hra Atlas of Fortune je zaraďovaná do žánru Adventure ako je definovaný v sekcií 3.1, vďaka začleneniu interaktívnych predmetov. Predmety a ich využitie boli inšpirované hrou Skyrim.

3.3 Nepriateľské prvky

Konflikt a obtiažnosť do hry uvádzajú nepriateľské prvky, ktoré hráčovi viacerými spôsobmi ubližujú. Neúspešný koniec hry je zapríčinený nepriateľskými prvkami.

V tejto sekcií sú definované a vysvetlené nepriateľské prvky, ktoré pôsobia v hre Atlas of Fortune. Medzi nepriateľské prvky patria:

- **Pasce** – Pasce sú statické nebezpečenstvá, ktoré je hráč schopný dočasne odzbrojiť.
- **Kreatúry** – Hráč je nimi prenasledovaný a napadnutý.

Hráčove suroviny sú vyčerpané nepriateľskými prvkami, čo vytvára u hráča motiváciu hlbšie úroveň preskúmať, aby suroviny doplnil. Nepriateľské prvky chce hráč buď obísť, ujsť im alebo ich poraziť.

3.3.1 Pasce

Pasce sú statické nebezpečenstvá, ktoré môže hráč objaviť v náhodných miestnostiach okrem štartujúcej miestnosti a únikového východu. Každá pasca sa od ostatných líši poškodením, ktoré vie spôsobiť, ich vzhľadom a funkciou. Patria sem:

- **Hroty** – Sú vždy v strede miestnosti, kde sú pravidelne vysúvané a stiahnuté. Je ich obtiažne preskočiť, a preto je lepšie ich preletieť alebo počkať kým sa stiahnu pred prechodom.
- **Sekera** – Je to zo stropu mávajúca sekera, ktorú nie je obtiažne obísť vo veľkých miestnostiach, ale v bežných miestnostiach spôsobí hráčovi škodu. Je vždy orientovaná na vstupné dvere, čo môže hráča prekvapiť.
- **Šípy** – Sú veľmi nenápadné ak ich hráč nehľadá. Diery, z ktorých šípy strieľajú, sa nachádzajú na strope. Jediná identifikácia, ktorú hráč môže využiť bez vedomosti o pasci, je zbrojná páka.
- **Prítlačná doska** – Je vždy v strede miestnosti, kde čaká, kým na ňu hráč stúpi, potom vybuchne a natrvalo prestane fungovať.

Nepriateľské kreatúry sú tiež ovplyvnené pascami. Každá miestnosť okrem počiatočnej miestnosti a únikového východu môže obsahovať pascu. Generátor náhodných čísel [6] determinuje, či a ktorá pasca je do miestnosti umiestnená. Pravdepodobnosť výskytu pasce v miestnosti je určená následujúcou rovnicou:

$$P_t = 0.25 \cdot d$$

V ktorej Pt reprezentuje pravdepodobnosť výskytu pasce v miestnosti a d reprezentuje hráčom vybranú obtiažnosť. Inklúzia pascí hru Atlas of Fortune začleňuje do žánru Adventure. Pasce a ich inklúzia boli inšpirované hrami Minecraft a Skyrim.

3.3.2 Kreatúry

Nepriateľské kreatúry sú kreatúry, ktoré hráčovu postavu naháňajú a napádajú. Sú ovládané nepriateľskou AI [15], konkrétne stavovými automatmi [2]. Stavový automat [2] je popísaný v sekcií 3.3.3.

Nepriateľské kreatúry vyvíjajú na hráča tlak tým, že ho naháňajú. Hráč je nútený kreatúram buď utiecť alebo s nimi bojovať. Kreatúry v hre sú nasledovné:

- **Sliz** – Je veľmi slabý. Hráč je schopný sliz poraziť rýchlo. V každej úrovni sa vyskytuje častejšie v porovnaní s druhou kreatúrou.
- **Zlatý sliz** – Je veľmi odolný. Hráč nie je schopný sliz poraziť rýchlo, to znamená, že sliz ho môže zahnať do kútov úrovne. Keď je sliz porazený, hráč je odmenený mincami.

Hra Atlas of Fortune vďaka slizom obsahuje aktívne nebezpečenstvo, ktoré hráč nemôže pomaly a jednoducho obísť. Slizy, ktoré hráča naháňajú, na hráča vyvíjajú tlak, ktorý hráčovi môže spôsobiť padnutie do pasce. Slizy sú schopné ničiť dvere. Nepriateľské kreatúry hru začleňujú do žánru Action. Slizy sú inšpirované hrou Minecraft.

3.3.3 Stavy

Kreatúry ako sú popísané v sekcií 3.3.2, sú ovládané nepriateľskou AI [15]. Táto AI má formu stavového automatu [2]. Stavový automat [2] je znázornený na obrázku 3.3. Stavy daného stavového automatu sú popísané nasledovne:

- **Nečinnosť** – Počiatočný stav automatu.
- **Hliadka** – Je vykonávaná ovládanou kreatúrou, ak nemá žiadny iný podnet.
- **Útok** – Ovládaná kreatúra sa nekompromisne približuje k hráčovi. Ak hráča nevidí, prepne sa do stavu vyhľadávanie.
- **Vyhľadávanie** – Hráč je určitý čas hľadaný ovládanou kreatúrou.
- **Prepnúť** – Páka v neaktívnom stave je prepnutá ovládanou kreatúrou.

Férovosť nepriateľských kreatúr je zaistená týmto stavovým automatom [2]. Hráč sa môže ovládaných kreatúr striasť. Hráči, ktorí systematicky vypínajú pasce však zistia, že kreatúry sú schopné ich znovu zapnúť, čo môže hráča motivovať k ničeniu kreatúr.

3.3.4 Interaktívne objekty

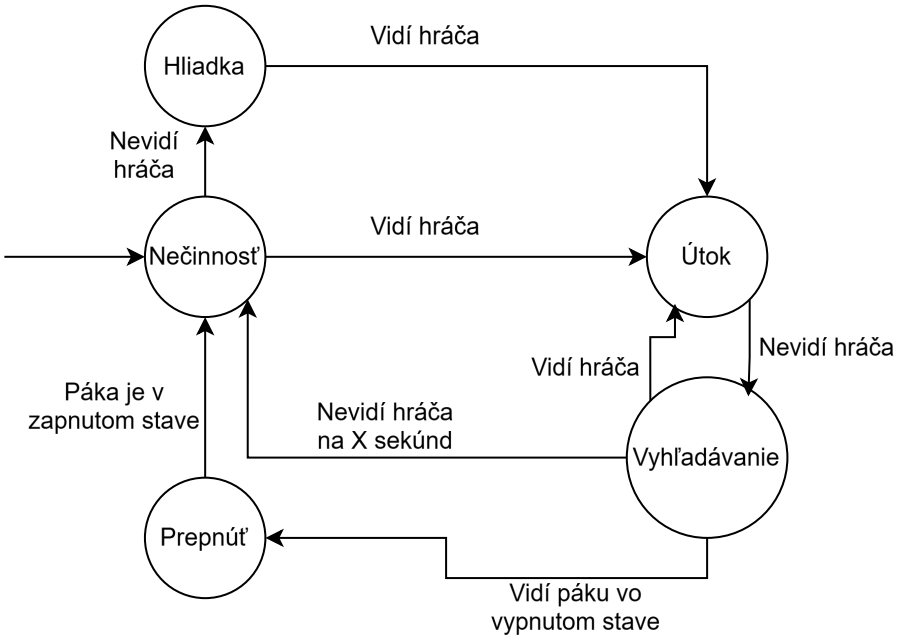
V hre Atlas of Fortune existujú tri interaktívne objekty:

- **Páka** – Pasce sú ovládané korešpondujúcimi pákami, ktoré keď sú prepnuté, pasca prestane fungovať. Jediná pasca, ktorá nemá páku je prítlačná doska.
- **Dvere** – Každá miestnosť obsahuje aspoň jedny dvere, je možné ich otvoriť a zatvoriť. Nepriateľské kreatúry nie sú schopné vidieť cez dvere ale sú schopné dvere zničiť.
- **Rebrík** – Je umiestnený v miestnosti únikového východu, interakcia s ním úroveň úspešne ukončí a hráč hru vyhráva.

Hráčovi rastú možnosti vďaka interaktívnym objektom. Hra je zaraďovaná do žánru Adventure vďaka interaktívnym objektom.

3.4 Hráč

V hre Atlas of Fortune sa hráč vžije do úlohy mladého draka. V tejto sekcii je popísané, čo to znamená, čiže ako sa drak môže hýbať, aké má schopnosti, aké má hráč suroviny, ako ich využíva a aké má hráč ciele.



Obr. 3.3: Stavový automat [2] hernej AI [15] v hre Atlas of Fortune.

Hráč má prístup k trom surovinám [13]. „*V tomto prípade je surovina definovaná ako objekt alebo informácia, ktorá je pri behu hry manipulovaná buď hráčom alebo hernými systémami a ovplyvňuje beh herných systémov a herných cyklov.*“ Parafrázovaná definícia suroviny pochádza z [13].

- **Zdravie** – Je to štandardná surovina [13], ktorú hráč stráca pri stretnutí s kreatúrami alebo pascami. Ak sa dostane na 0, tak sa hra končí prehrou.
- **Výdrž** – Táto surovina [13] sa míňa, keď hráč šprintuje alebo lieta. Ak sa minie, hráč nemôže šprintovať alebo lietať. Túto surovinu hráč získava naspäť, ak ju nestráca.
- **Mana** – Mana je surovina [13], ktorá je čerpaná hráčom na využitie špeciálnych schopností a útokov. Túto surovinu hráč pomaly získava naspäť, ak ju nestráca.

Suroviny si musí strážiť hráč. Ak sa minú, tak niektoré schopnosti a spôsoby pohybu sa stanú nedostupnými. Suroviny sú ovplyvnené všetkými hernými systémami a hráč sa ich musí naučiť využívať, aby v hre uspel.

3.4.1 Pohyb

Hráč má niekoľko spôsobov pohybu. Pohyb je jeden z najdôležitejších spôsobov ako hráč s hrou a jej cyklami [13] interaguje. Hráčovi prístupné spôsoby pohybu sú nasledovné:

- **Chôdza** – Chôdza je základný spôsob pohybu hráča, je pomalý, ale hráč môže tento spôsob pohybu využiť vždy. Hráč, ktorý využíva len tento spôsob pohybu, nie je schopný ujsť kreatúram, ktoré ho prenasledujú.
- **Skok** – Hráč má skok prístupný vždy a je schopný ho využiť na zbieranie predmetov položených vyššie, bez potreby strácania suroviny Výdrže. Skok môže byť využitý na vyhýbanie sa pasciam alebo kreatúram.

- **Šprint** – Šprint je spôsob pohybu hráča, v ktorom sa hráč pohybuje rýchlejšie ako v chôdzi. Hráčom je umožnené pomocou šprintu útek nepriateľským kreatúram a rýchlejší prieskum úrovne. Pri šprinte sa stráca surovina [13] Výdrž, pričom nie je možné šprintovať ak sa hráčovi surovina Výdrž minie.

- **Let** – Let je najrýchlejší spôsob pohybu hráča. Hráčovi umožňuje rýchlo utiecť prenasledujúcim kreatúram. Má aj značné nevýhody, vchody a východy miestností sú vždy pri zemi a je potrebné míňať surovinu [13] Výdrž. Ak sa hráčovi surovina Výdrž minie, nemôže Let využívať a ak práve lieta tak je let zrušený a hráč spadne na zem.

Surovina Výdrž je neoddeliteľnou časťou efektívneho využívania spôsobov pohybu pre hráča. Hráč musí surovinu Výdrž využívať múdro, alebo si ju v kritických momentoch doplniť využitím zeleného elixíru. Plytvanie surovinou Výdrž môže hráčovi spôsobiť prehru, keď kreatúry hráča zaženú do slepej uličky.

3.4.2 Schopnosti

Hráč má schopnosti, ktoré mu pomáhajú bojovať s kreatúrami a pri prieskume úrovne. Hráč má k dispozícii následujúce schopnosti:

- **Ohnivá guľa** – Ohnivá guľa je útok na diaľku. Hráčovi je umožnené pomocou nej s kreatúrami bojovať bez rizika straty suroviny Zdravia, pričom sa ale stráca surovina Mana. Ak je surovina Mana úplne vyčerpaná, táto schopnosť sa stáva nedostupnou.

- **Ohnivý dych** – Ohnivý dych je útok na blízko. Hráčovi umožňuje bojovať s nakopenými prenasledujúcimi kreatúrami, pričom riziko straty suroviny Zdravia je vyššie z dôvodu nízkeho dosahu tejto schopnosti. Schopnosť míňa surovinu Mana, ak je surovina úplne vyčerpaná, schopnosť sa stáva nedostupnou.

- **Vízia pokladov** – Vízia pokladov je nebojová schopnosť. Hráčovi umožňuje vidieť všetky poklady v úrovni. Schopnosť míňa surovinu Mana, ak je surovina polovične vyčerpaná.

- **Fyzický útok** – Fyzický útok je útok na blízko. Hráčovi umožňuje brániť sa aj pri úplnom vyčerpaní suroviny Mana. Surovina [13] Mana sa nestráca pri využití tejto schopnosti. Využitie schopnosti je však veľmi riskantné, jej dosah je veľmi malý, preto hráč riskuje stratenie suroviny [13] Zdravie pri pokusoch o využitie tejto schopnosti.

Surovina [13] Mana je dôležitá súčasť využívania schopností. Jej vyčerpanie znamená pre hráča odobratie silnejších schopností pre boj a prieskum. Ak hráč potrebuje surovinu [13] Mana núdzovo doplniť, má príležitosť využiť modrý elixír.

3.4.3 Cieľ

Cieľ hráča je definovaný ako séria podmienok a metrík, ktoré je hráč motivovaný splniť, aby dosiahol úspešné ukončenie úrovne. Ciele sa delia na primárne a sekundárne, pričom primárne ciele musia byť splnené pre úspešné ukončenie úrovne a pre sekundárne ciele. Závisí na hre, či sú ciele pevne definované, alebo si hráč musí určiť vlastné. Hra Atlas of Fortune určuje hráčove ciele nasledovne:

- **Útek zo žalára** – Útek zo žalára je primárny cieľ hráča. Ak na konci hry nie je splnený, hráč hru prehráva. Tento stav nastáva len v prípade vyčerpania suroviny

[13] Zdravia. Aby bol cieľ splnený, hráč musí objaviť únikovú miestnosť a vykonať interakciu s interaktívnym predmetom rebrík, ktorý sa v nej nachádza.

- **Zber pokladov** – Zber pokladov je sekundárny cieľ hráča. Tento cieľ nie je považovaný za dokončený alebo nedokončený, skôr je využívaný ako merítko výkonu hráča. Čím viac pokladov hráč nazbieral, tým vyššie má skóre a teda je považovaný za úspešnejšieho ako pokus s nižším nazbieraným pokladom.

Kapitola 4

Realizácia herných systémov

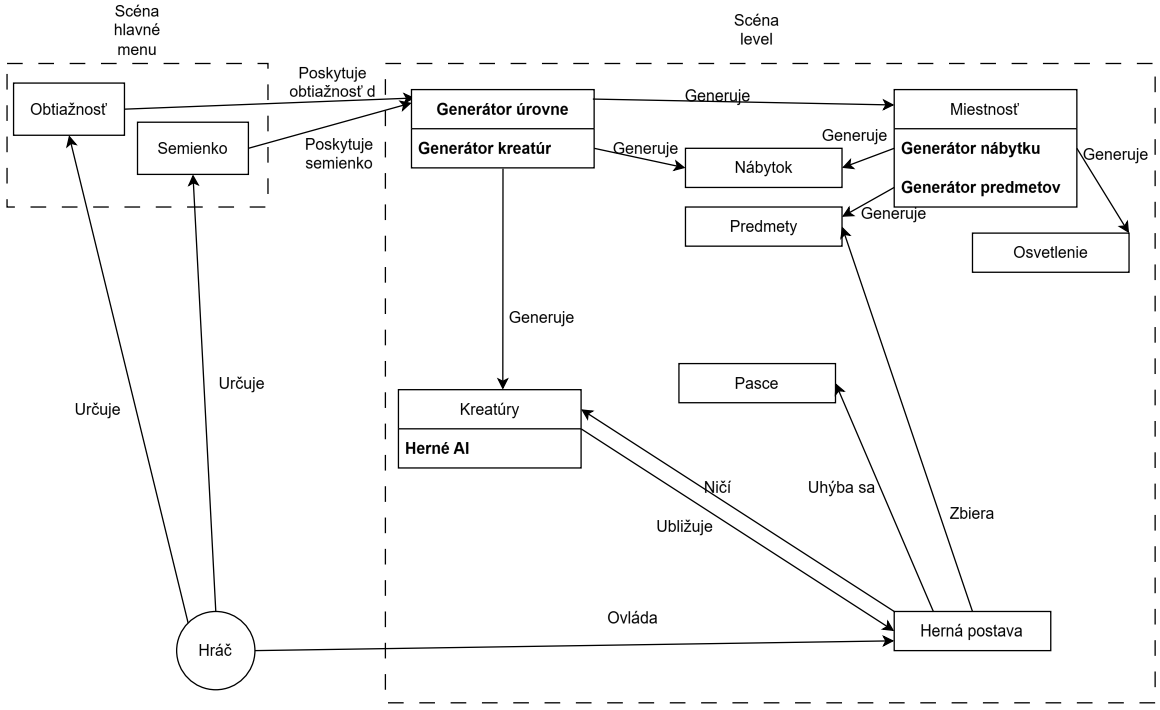
V tejto kapitole sú rozvinuté a vysvetlené detaily realizácie herných systémov podľa návrhu hry, ktorý je rozvinutý v kapitole 3. Začína prehľadom herných systémov, kde je vysvetlená schéma herných komponentov a pokračuje vysvetlením generátorov úrovní, kde je rozvinutá problematika a riešenie generovania miestností, pascí a nepriateľských kreatúr. Je detailne rozvinuté generovanie objektov a predmetov a ako je dokončená úroveň naplnená. Naposledy je rozvinuté herné AI [15] nepriateľov a ako herné systémy rozhodujú o napĺňaní úrovní nepriateľmi. Nakoniec je zhodnotená možnosť publikovania hry na rôznych herných obchodoch a zhodnotená pravdepodobnosť úspechu hry, aj ďalšie kroky, ktoré môžu byť uskutočnené

4.1 Prehľad herných systémov

V návrhu systémov v kapitole 3 sú definované všetky herné systémy, ktoré sa majú v hre nachádzať, aby boli zaradené do žánrov exploration [11], adventure [11] a action [11]. Otázka je nasledovná: Ako zmeniť návrh na realizovaný projekt? Projekt je realizovaný v hernom engine Unity, ktorý využíva objekty [3]. Systémy z návrhu teda musia byť obsiahnuté v daných herných objektoch [3]. Objekty [3] obsahujú dáta a metódy, ktoré sú daným objektom aj inými manipulované. Navrhnuté systémy teda musia byť mapované na herné objekty [3]. Herné objekty v engine [14] sú zoskupené do scén. Objekty [3] v rôznych scénach sú schopné si vymieňať len jednoduché dáta. Všeobecný postup mapovania je nasledovný:

1. Herný systém v návrhu je identifikovaný a je dôležité rozlíšiť medzi systémami, objektmi [3] a surovinami [13].
2. Herný systém vytvoriť ako metódu objektu alebo ako objekt samotný.
3. Prepojiť výsledné objekty aby bola možná komunikácia medzi hernými objektmi.
4. Zoskupiť objekty do scén tak, aby mohli dostatočne navzájom komunikovať.
5. Definovať, s ktorými systémami môže hráč komunikovať.

Na obrázku 4.1 je definovaná výsledná objektová schéma tohto procesu. Je v nej zobrazený hráč a všetky herné systémy ako objekty, ktoré medzi sebou komunikujú, alebo objekty vytvárajú. Hráč samotný má vlastnú reprezentáciu. V schéme sú následujúce systémy:



Obr. 4.1: Objektová [3] schéma hry Atlas of Fortune. Herné objekty a hráč sú reprezentované ako bloky v schéme, na ktoré sú mapované herné systémy (v hrubom písme). Výmena dát a tvorenie nových objektov sú reprezentovaná šípkami.

- **Generátor úrovne** – Herný objekt je zdieľaný generátorom úrovne a generátorom nepriateľov zároveň, čo uľahčuje výmenu informácií medzi nimi. Vytvára herné objekty, na ktoré sú mapované generátory nábytku a predmetov.
- **Generátor kreatúr** – Herný systém je obsiahnutý v tom istom objekte ako generátor. Tieto systémy musia vykonať výmenu informácií o obtiažnosti a informácií o miestnostiach.
- **Generátor nábytku a predmetov** – Generátory nábytku a predmetov sú obsiahnuté v objektoch miestnosti. Predmety vyžadujú na generovanie nábytok.
- **Herné AI** – Herná AI je obsiahnutá v rámci objektu každej vygenerovanej kreatúry. Činnosti kreatúry sú určené herným AI.

4.2 Generátor úrovní

Generátor [5] úrovní má podľa návrhu v podkapitole 3.2.2, využívať algoritmus WFC [1]. Tento algoritmus umožňuje generovanie vyžadovaného vzhľadu úrovne. Aby algoritmus bolo možné realizovať, je potrebné splniť jeho požiadavky. Pre algoritmus musia byť definované políčka, obmedzenia a musí byť vytvorená teoreticky nekonečná mriežka, ktorú algoritmus WFC [1] vypĺňa. Unity neobsahuje mriežku, ktorá by tento účel spĺňala. V nasledujúcich podkapitolách je vysvetlené, ako sú vyžadované podmienky splnené.

4.2.1 Miestnosti

Pri riešení prvej podmienky algoritmu WFC [1] a zároveň dodržiavaní návrhu z kapitoly 3 sa pridružujú ďalšie problémy. Sada políčok musí byť ľahko prístupná generátoru úrovní. Každé políčko musí byť správne identifikované, aby mohlo byť vyradené obmedzeniami.

Unity engine obsahuje riešenie prvého problému vo forme „Prefabov“. Prefab v hernom engine [14] Unity, je definovaný ako objekt [3], ktorý bol vývojárom uložený vrátane svojich dcérskych objektov. Prefaby sú v engine [14] Unity využívané na tvorenie identických kópií. V tomto prípade je každé políčko miestnosť, ktorá má tvar kocky s rozmermi 5x5x5. Vzdialenosti nie sú presne definované v reálnych jednotkách vzdialenosti (napríklad v metroch), sú teda bez označenia. Políčka sú takto vytvorené v Unity engine a uložené ako Prefaby. Druhý problém je riešený tvorbou príznakov, ktoré môžu byť porovnané s obmedzeniami.

Tvorba teoreticky nekonečnej mriežky v hernom engine [14] obsahuje prekážky. Definícia buniek mriežky a určenie ich polohy sú dve hlavné prekážky. Definícia bunky je určená podľa políčok, mriežka musí byť definovaná v 2D priestore a tak rozmer každej bunky je 5x5. Stred každej bunky je teda od stredu ostatných buniek vzdialený o 10 jednotiek vzdialenosti. Tým je vytvorená teoreticky nekonečná mriežka. Pozícia a rotácia políčka je odvodená od stredu bunky.

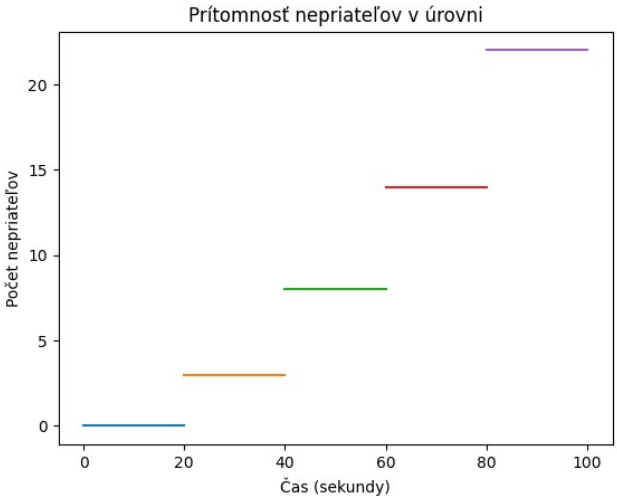
Sada obmedzení je posledná podmienka, ktorá musí byť splnená, aby mohol byť využitý algoritmus WFC [1]. Sada obmedzení je realizovaná nasledovne: Každé políčko obsahuje východy, ktoré ak ústia do steny alebo prázdnej bunky, obmedzenie je porušené. Preto môžu byť vygenerované len tie políčka, ktorých východy sú smerované na prázdne políčka, do ktorých neústi žiadny iný východ. Limitom je druhé obmedzenie, ktoré je nutné realizovať, že musí existovať nezablokovaná cesta od začiatku úrovne do jej konca.

Všetky podmienky pre využitie algoritmu WFC [1] sú splnené. Algoritmus WFC [1], tak, ako je definovaný v podkapitole 2.3.3, však nie je vhodný pre realizáciu, pretože by pokračoval do nekonečna. Musí byť upravený nasledovne:

1. Inicializácia začiatočného políčka a tabuliek obmedzení.
2. Náhodný výber bunky, do ktorej ústi východ pre kolaps.
3. Políčko, do ktorého bunka skolabuje, je vybrané náhodne.
4. Ak políčko nevyhovuje obmedzeniu, tak algoritmus je vrátený do kroku 3.
5. Vybraná bunka je skolabovaná do vybraného políčka.
6. Opakovanie od kroku 2 dovtedy, kým všetky bunky, do ktorých ústi východ, prejdú kolapsom.

Táto realizácia algoritmu WFC [1] je prvý krok k prevencii nekonečne dlhého vykonávania algoritmu WFC [1]. Ak sú náhodne vybrané políčka, ktoré obsahujú veľa východov, mohlo by sa stať, že aj daná realizácia algoritmu WFC [1] by nikdy neskončila. Druhý a posledný krok sú správne nastavené váhy políčok pri kolapse bunky. Váha je v tomto prípade definovaná ako pravdepodobnosť výberu daného políčka pre kolaps. Krok je vykonaný realizáciou druhého obmedzenia a zároveň úpravy váh.

Druhé obmedzenie je prakticky realizované obmedzením kolapsu bunky vybranej v iterácií l, kde hodnota l je definovaná v podkapitole 3.2.2 na políčko únikového východu. Váhy políčok sú zmenené, aby boli preferované políčka s nižším počtom východov.



Obr. 4.2: Postupnosť zvyšovania počtu nepriateľských kreatúr pri obtiažnosti d=1, s predpokladom žiadnych porazených nepriateľov hráčom.

Posledné nezrealizované obmedzenie, teda obmedzenie typov miestností, je jednoducho realizovateľné. Je implementované pomocou bezkontextovej gramatiky [7], definovanej nasledovne:

$$G = (N, T, P, S)$$

Jednotlivé množiny sú definované nasledovne:

$$N = \{S, A, B, C\}$$

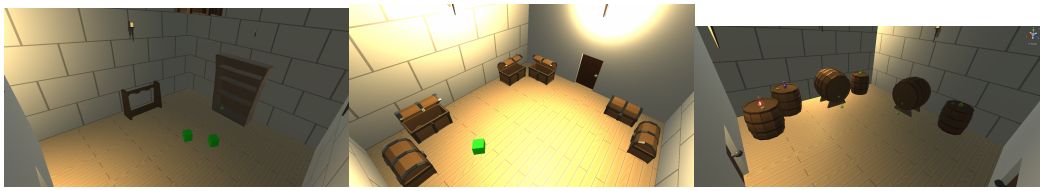
$$T = \{a, b, c\}$$

$$P = \{1 : S \to aA|bB|cC, 2 : A \to bB|cC|\epsilon, 3 : B \to aA|cC|\epsilon, 4 : C \to aA|bB|\epsilon\}$$

Typy miestností sú určené výstupom z gramatiky [7] v poradí kolapsu buniek. Miestnosti zodpovedajúce terminálnemu symbolu a sa stávajú zbrojnicami, terminálnemu symbolu b sa stávajú skladiskami a terminálnemu symbolu c sa stávajú pokladnicami. Typy miestností sú zobrazené v obrázku 4.3.

4.2.2 Nepriateľské kreatúry

Keď sú všetky miestnosti vygenerované, aktivuje sa generátor nepriateľských kreatúr. Každá miestnosť obsahuje niekoľko prázdnych herných objektov. Prázdne objekty sú objekty, ktoré neobsahujú žiadny model. Prázdne herné objekty nie sú viditeľné hráčovi, čím sa stávajú užitočnými pre účely vloženia herných systémov bez rizika objavenia hráčom. Prázdne herné objekty sú vhodné aj pri vytváraní kópií objektov, v tomto prípade nepriateľských kreatúr. Tieto objekty sa nazývajú „spawner“ a ide o názov inšpirovaný hrou Minecraft. Súradnice nových kópií objektov sú relatívne k ich rodičovi a tieto súradnice sa nazývajú lokálne súradnice. Relatívnosť k rodičovi znamená že súradnice sú sčítané od objektu ku koreňovému objektu aby boli determinované takzvané svetové súradnice, súradnice relatívne k bodu $(0, 0, 0)$ v scéne. Svetové súradnice môžu byť vypočítané následujúcim spôsobom, kde n je



Obr. 4.3: Ukážka miestností Zbrojnica, Pokladnica a Skladisko resp.

počet objektov, ktoré sú hľadanému objektu nadriadené v hierarchií scény:

$$(x, y, z) = (\sum_{i=1}^{n} x_i, \sum_{i=1}^{n} y_i, \sum_{i=1}^{n} z_i)$$

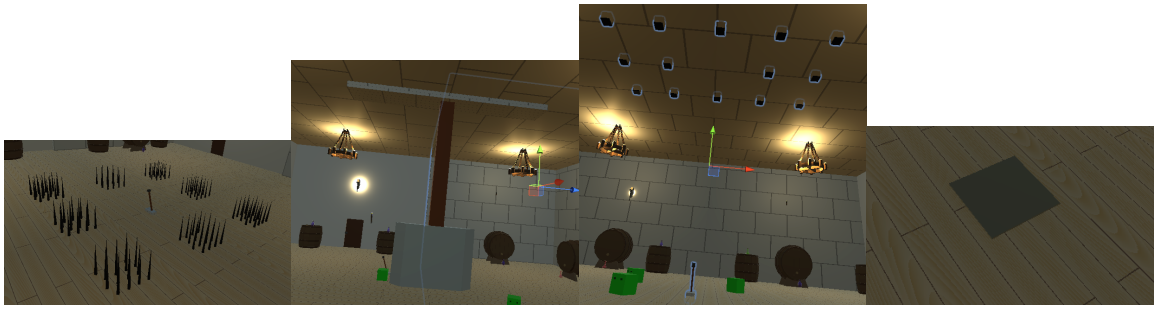
Tento spôsob je využitý aj pri určení počiatočnej rotácie kreatúry. Kreatúry nesmú byť vytvárané príliš rýchlo z dôvodu vyčerpania zdrojov počítača a zároveň, ak by počítač mal neobmedzené zdroje, hráč by bol rýchlo ohromený a porazený. Riešením tohto problému je generovanie kreatúr v takzvaných vlnách. Vlny sú definované ako časový interval, ktorý keď uplynie, tak sa pomocou spawnerov vygeneruje určitý počet kreatúr. Počet vygenerovaných kreatúr je určený obtiažnostnou krivkou [12] definovanou v podkapitole 3.2.1. Počet nepriateľov je zobrazený na obrázku 4.2.

4.3 Generovanie objektov a predmetov

Generovanie objektov a predmetov je vykonané ihneď po dokončení generovania miestností. Pod objektmi sa v tomto prípade rozumie rôzny nábytok, ktorý miestnosti môže naplniť a pod predmetmi sa rozumejú interaktívne predmety ako elixíry, kľúče a mince, teda predmety, ktoré hráč môže zbierať a využívať. Cieľom generovania objektov a predmetov je, aby každá miestnosť obsahovala predmety, ktoré hráč chce zbierať. Problémy pri generovaní predmetov a objektov sú podobné problémom pri generovaní nepriateľských kreatúr. Umiestnenie nábytku závisí na miestnosti. V každej miestnosti sú umiestnené prázdne objekty [3], ktoré sú dcérskymi objektmi miestnosti a sú využité na určenie presných súradníc a rotáciu nábytku spôsobom, ktorý je popísaný v podkapitole 4.2.2. V tejto podkapitole je popísaná aj pravdepodobnosť vygenerovania nábytku. Konkrétny umiestnený nábytok a to, či konkrétne potenciálne miesto nábytku zaplní, závisí na type miestnosti, ktorý je určený pri generovaní miestností a špecifikovaný v podkapitole 3.2.2. Umiestnenie predmetov je určené prázdnymi objektmi, ktoré sú dcérskymi objektmi vygenerovaného nábytku. Podobne ako pri nábytku, typy predmetov, ktoré sa vygenerujú, závisia na type miestnosti, v ktorej sa nachádzajú. Predmety a nábytok sa generujú len jeden krát. Predmetov je teda v jednej úrovni obmedzený počet. Ak ich hráč nájde a využije všetky, nebude môcť nájsť žiadne iné. Takáto situácia je ale veľmi nepravdepodobná, pretože pri obvyklom správaní hráč určite objaví únikový východ skôr, ako objaví všetky predmety v úrovni. Situácia teda môže nastať len v prípade, že sa hráč rozhodne využiť kľúče na odomknutie truhlíc v predchádzajúcich miestnostiach. Objekty a predmety teda vyplnia všetky miestnosti podľa ich typu. Pravdepodobnosť miestnosti, ktorá je vygenerovaná bez predmetov a nábytku, je nasledovná:

$$P = (1 - P_n)^x$$

Pn je definované v podkapitole 3.2.2, pričom je vybraná najvyššia obtiažnosť $d = 1.5$ a najmenší počet prázdnych objektov v políčku, pod ktorými môže byť vygenerovaný kus



Obr. 4.4: Ukážka pascí zľava doprava: Hroty, Sekera, Šípy, Prítlačná doska

nábytku, je $x = 8$. Začínajúca miestnosť nikdy neobsahuje predmety alebo nábytok.

$$P_n = \frac{0.7}{1.5} \doteq 0.46$$

P_n je zaokrúhlené nadol.

$$P = (1 - 0.46)^8 \doteq 0.00723$$

Pravdepodobnosť je nižšia ako 1%. Za každých sto vygenerovaných miestností, menej ako jedna bude vygenerovaná bez nábytku alebo predmetov. Úroveň neobsahuje viac ako sto miestností. Takto nízka pravdepodobnosť je pre hru prijateľná.

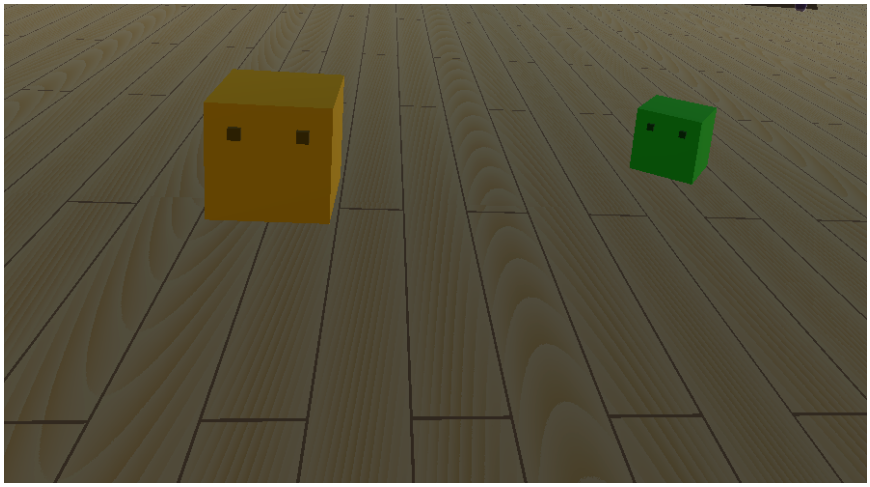
4.3.1 Pasce

Generovanie pascí je ďalším krokom generovania objektov a jeho účelom je predstaviť dodatočnú obtiažnosť pre hráča. Pasce sú obmedzené vplyvom na miestnosť, v ktorej sú vygenerované. Hráč musí byť schopný pascu obísť a niektoré pasce môžu byť hráčom deaktivované. Metóda ich generovania je podobná generovaniu predmetov a objektov až na niekoľko detailov. Pasce sa vždy generujú v strede miestnosti, kde majú najväčší vplyv na celú miestnosť. Pasce potrebujú dva komponenty:

- **Spúšťací mechanizmus** – Aktívnosť pasce je určená týmto mechanizmom.
- **Poškodzovací mechanizmus** – Hráča a nepriateľské kreatúry tento mechanizmus poškodzuje a ničí. Môže spôsobiť prehru hráča. Je funkčný len ak je pasca aktívna.

Spúšťací mechanizmus musí byť interaktívny pre hráčov aj pre nepriateľské kreatúry. Musí teda obsahovať spôsob ako zaznamenať hráča a kreatúry, ktoré môžu s mechanizmom komunikovať. Tento spôsob je uskutočnený pomocou takzvaných „Colliderov“. Collidery zaznamenávajú strety medzi sebou a inými collidermi. Táto udalosť sa dá odchytiť a využiť na prepnutie spúšťacieho mechanizmu na opačný stav či hráčom alebo aj nepriateľskou kreatúrou.

Na obrázku 4.4 sú ukázané pasce, ich škodlivé mechanizmy a spúšťacie mechanizmy. Hráčovi je umožnené pasce obísť alebo deaktivovať pomocou spúšťacieho mechanizmu. Jediná pasca, ktorá nemôže byť prepnutá do vypnutého stavu bez spôsobenia škody hráčovi, je prítlačná doska. Prítlačná doska bola aktivovaná nepriateľskými kreatúrami predtým, ako bola prítlačná doska objavená hráčom. Prítlačná doska je už schopná reagovať len pri kolízii s hráčom.



Obr. 4.5: Nepriateľské kreatúry, ktoré sa vyskytujú v hre. Napravo normálny sliz a naľavo zlatý sliz.

4.4 AI nepriateľov

Herné AI [15] musí byť schopné kompetentne ovládať nepriateľské kreatúry, nesmie byť ale neférové. Cieľom tejto podkapitoly je rozvinúť realizáciu dvoch typov kreatúr, ktoré sa v hre vyskytujú, odôvodniť ich prítomnosť a vysvetliť ako herné AI [15] určuje cestu k svojmu cieľovému objektu [3]. Nakoniec sú popísané stavy, v ktorých sa stavový automat [2] herného AI [15] môže nachádzať, ako sa herné AI [15] správa v jednotlivých stavoch a ako herné AI [15] určuje, kedy sú splnené podmienky pre zmenu stavu. Férovosť hernej AI [15] závisí na nasledujúcich pravidlách, vytvorených pre hru Atlas of Fortune:

1. Informácie, ktoré nepriateľské kreatúry nezískali pomocou „zmyslov“, nesmú byť využité proti hráčovi. Jediný zmysel, ktorý je hrou Atlas of Fortune braný do úvahy je zrak.

2. Počet a obtiažnosť nepriateľských kreatúr musia byť nepriamo úmerné.

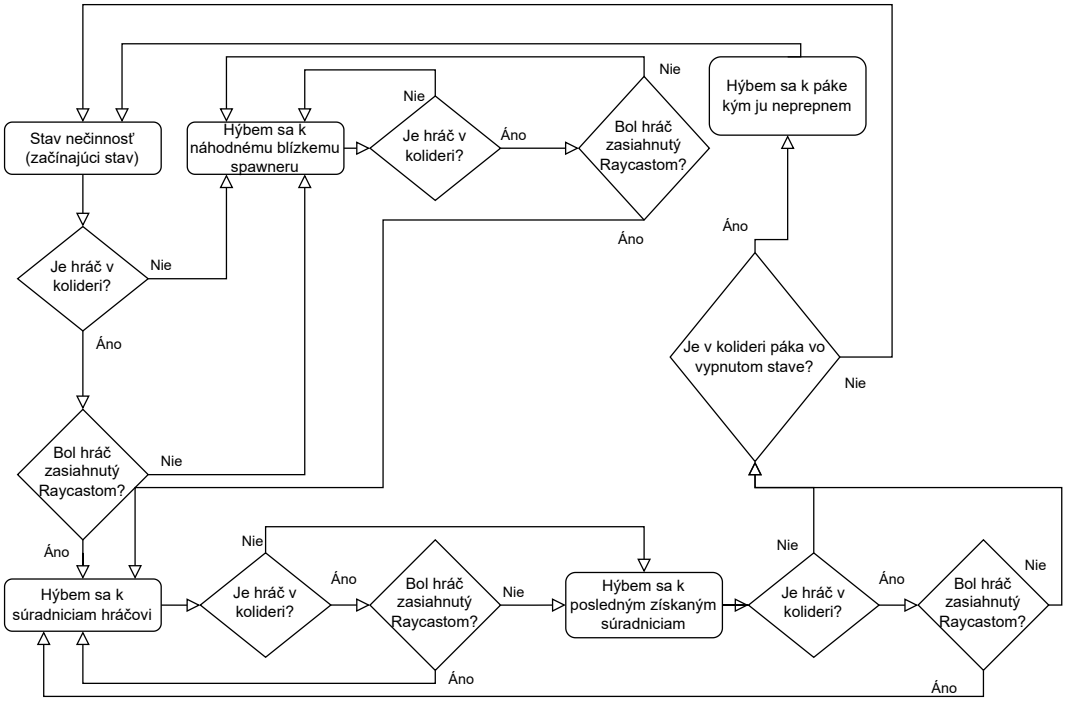
3. Obtiažnosť porazenia herného AI [15] musí korešpondovať s odmenou za porazenie herného AI [15] a so vzhľadom nepriateľskej kreatúry.

4. Nepriateľské kreatúry musia dodržovať podobné pravidlá pohybu ako hráč. Čiže nesmie nastať situácia, kde nepriateľské kreatúry sú schopné chodiť cez steny alebo nábytok.

Porušenie týchto pravidiel herným AI môže viesť k frustrácií a predčasnému zakončeniu hry hráčom. Tento výsledok je neprijateľný. Následne je teda popísané aj to, ako realizácia herného AI predchádza porušeniu týchto pravidiel.

4.4.1 Typy

Dva typy nepriateľských kreatúr v hre Atlas of Fortune sú popísané v podkapitole 4.2.2. Ich realizácia musí vyhovovať pravidlám stanovených v podkapitole 4.4. Obtiažnosť nepriateľských kreatúr je definovaná ako úsilie, ktoré musí byť vynaložené hráčom, aby bola nepriateľská kreatúra porazená. To znamená, že zlatý sliz musí byť obtiažnejší na porazenie



Obr. 4.6: Diagramom je popísaný algoritmus chovania nepriateľských kreatúr

a tiež mať menšiu pravdepodobnosť vygenerovania ako normálny sliz. Vyššia obtiažnosť porazenia zlatého slizu je dosiahnutá pridaním suroviny Životov, ktorú hráč pri útoku nepriateľským kreatúram odoberá. Normálny sliz pri vygenerovaní obsahuje 10 životov a zlatý sliz obsahuje 30 životov. Zlatý sliz je teda tri krát obtiažnejší na porazenie a musí byť aj tri krát vzácnejší ako normálny sliz. Pravdepodobnosť vygenerovania zlatého slizu teda je:

$$P = \frac{1}{3}$$

Zlatý sliz musí byť od normálnych slizov odlíšený aj vzhľadovo, hráča by inak mohli kreatúry zmiasť. Vzhľad kreatúr je možné vidieť na obrázku 4.5. Typovanie nepriateľských kreatúr napĺňa tretie pravidlo férovej hernej AI [15].

4.4.2 Navigácia

AI nepriateľov využíva Navigation mesh, ďalej nazývaný NavMesh pre hľadanie cesty. Herný engine [14] Unity poskytuje zabudovaný NavMesh ako nástroj pre vývojárov. NavMesh je vytvorený po vygenerovaní miestností a nábytku. Herné AI [15] využíva NavMesh, spolu so súradnicami spawnerov definovaných v podkapitole 4.2.2, na vytvorenie cesty, po ktorej sa nepriateľské kreatúry pohybujú. Ak kreatúry hráča naháňajú, sú využívané svetové súradnice hráča. Svetové súradnice sú definované v podkapitole 4.2.2. Správne generovanie a využitie NavMeshu predchádza situáciám, kde sa nepriateľské kreatúry hýbu nelogickým spôsobom ako napr. cez steny alebo nábytok. Týmto realizácia pomocou NavMeshu spĺňa štvrté pravidlo férovosti herného AI.

4.4.3 Stavy

V tejto podkapitole sú popísané stavy stavového automatu [2] hernej AI [15] a ako sú nepriateľské kreatúry ovládané hernou AI [15]. Algoritmus, ktorý herné AI [15] využíva, je graficky znázornený na obrázku 4.6. Prvé pravidlo férovosti hernej AI [15] je riešené daným stavovým automatom [2] pomocou nasledujúcich nástrojov:

- **Collider** – zaznamenáva kolízie s objektmi [3], ktoré sú schopné ovplyvňovať stav automatu [2].
- **RayCast** – zaznamenáva, či sú objekty v priamom dohľade kreatúry. Ak je zasiahnutý iný objekt ako bol cieľ RayCast, objekt nie je kreatúrou videný a teda kreatúra nesmie reagovať na jeho prítomnosť.

Počiatočný stav stavového automatu [2] je **Nečinnosť**. V tomto stave sa kreatúra „rozhliadne“. Rozhliadnutie je v tomto prípade definované ako zaznamenanie dôležitých objektov pomocou Collideru a kontrola či sú dané objekty v dohľade pomocou RayCastu. Ďalší stav je určený rozhliadnutím sa a videním určitých objektov. Následne sú popísané podmienky a zmeny stavov:

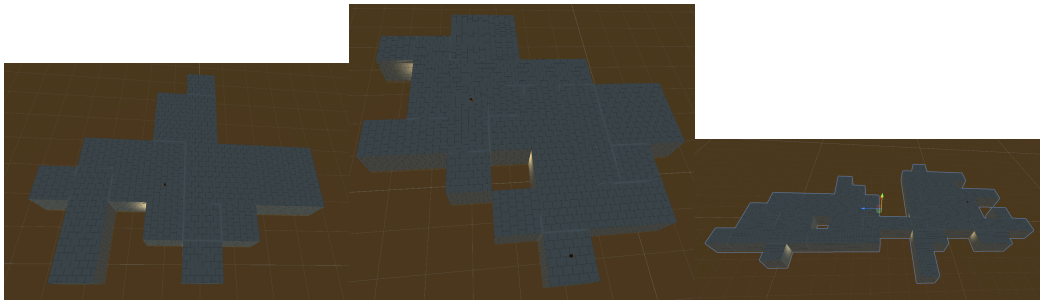
- **Nečinnosť** – Nepriateľská kreatúra sa rozhliadne a ak je zaznamenaný hráč, tak stav je zmenený na **útok**. Ak nie je zaznamenaný hráč, stav je zmenený na **hliadka**.
- **Útok** – Súradnice hráča sú využívané na určenie pohybu kreatúry. Rozhliadnutie sa a zaznamenávanie hráča prebieha dovtedy, kým zaznamenanie hráča nezlyhá a stav je vtedy zmenený na **vyhľadávanie**.
- **Vyhľadávanie** – Posledné súradnice hráča, ktoré boli zaznamenané pri rozhľade počas stavu **útok** sú využité na pohyb kreatúry. Ak je hráč pri rozhľade znova zaznamenaný, stav je prepnutý na **útok**. Ak je pri rozhľade zaznamenaná páka vo vypnutom stave, stav sa zmení na **prepnúť**.
- **Prepnúť** – Súradnice vypnutej páky sú použité ako destinácia kreatúry. Páka je kreatúrou prepnutá a stav sa mení na **nečinnosť**.
- **Hliadka** – Nepriateľská kreatúra sa rozhliadne a je vybraný náhodný spawner, ktorý sa stane destináciou pohybu kreatúry. Ak je zaznamenaný hráč, stav je zmenený na **útok**.

Stav **útok** je stavovým automatom uprednostňovaný oproti ostatným stavom. Týmto stavovým automatom je zaručované prvé pravidlo férovosti hernej AI [15].

4.5 Využitie cudzích zdrojových kódov, modelov a animácií

V hre Atlas of Fortune sú využité nasledujúce cudzie zdroje:

- 3D model, animácie, schopnosti, zvuky a ovládanie hráča boli získané z Unity asset store: Little Dragons: Tiger a Animal Controller (Malbers Character Controller) od autora MalberS Animations.
- 3D Modely nábytku, predmetov a osvetlenia boli získané z Unity asset store: Ultimate Low Poly Dungeon od autora Broken Vector.



Obr. 4.7: Výsledok generovania úrovní pri obtiažnosti resp. 0,5; 1,0; 1,5.

- 3D modely všetkých mincií boli získané z Unity asset store: Gold Coins od autora Devtoid.
- Komponenty navigačnej siete boli získané z:

  https://github.com/Unity-Technologies/NavMeshComponents od autora Unity technologies.
- Shader graf dreva, ktorý využíva Perlinov šum [9], obsahuje interpretáciu následovného skriptu ako časť shader grafu:

  https://thebookofshaders.com/edit.php#11/wood.frag od autora @patriciogv – 2015 http://patriciogonzalezvivo.com

4.6 Testovanie hry

Výsledok realizácie hry je otestovaný nasledovnými spôsobmi:

- Generovanie úplných úrovní pri rôznych semienkach.
- Generovanie správnych typov a počtu objektov, pascí a predmetov v úrovni.
- Generovanie správneho počtu nepriateľov podľa obtiažnostnej krivky.
- Testovanie správneho správania nepriateľských kreatúr.

Prvý test je vykonaný pre generovanie úrovní a ako vstupy sú použité semienka ako vstup náhodného generátora čísel 202452236, 397546757, 823917974. Obtiažnosť je respektívne 0,5; 1,0; 1,5. Test je považovaný za úspešný, ak všetky vstupy vytvoria úplnú úroveň, čiže úroveň, ktorá vyhovuje limitu generovania a obmedzeniu plných východov definovaných v podkapitole 3.2.2. Výsledky testu sú znázornené na obrázku 4.7.

Druhý test je vykonaný pre generovanie objektov a predmetov. Používa semienka ako vstup náhodného generátora čísel 202452236, 397546757, 82391797 a nasledovné obtiažnosti d : 0,5; 1,0; 1,5. V experimente sa dbá na počet vygenerovaných objektov a či náležia korektnej pravdepodobnosti definovanej v podkapitole 3.2.2. Ďalej sa kontroluje či objekty a predmety náležia typu miestnosti, v ktorej sa nachádzajú. Experiment je považovaný za úspech, ak podiel vygenerovaných objektov a maxima vygenerovaných objektov sa rovná svojej pravdepodobnosti s toleranciou 0,05. Výsledky experimentu sú nasledovné:

Druhý test prvý experiment			
	$d = 0,5$	$d = 1,0$	$d = 1,5$
Počet vygenerovaných kusov nábytku	97	152	292
Maximálny počet vygenerovaných kusov nábytku	123	221	543
Podiel	0.78861	0.68778	0.5377517
Počet vygenerovaných predmetov	94	82	99
Maximálny počet vygenerovaných predmetov	124	202	362
Podiel	0.78861	0.66129	0.27348

Pravdepodobnosti *Pn* pre jednotlivé obtiažnosti sú 0,85; 0,7; 0,65. Pravdepodobnosti *Pp* pre jednotlivé obtiažnosti sú 0,75; 0,5; 0,25. Experiment parciálne zlyhal. V tomto prípade by bolo potrebné podstúpiť viacero experimentov s rôznymi semienkami a aby sa podiely priemerovali medzi experimentami. Iba to určí definitívne úspech alebo zlyhanie testu.

Tretí test je vykonaný pre generovanie nepriateľských kreatúr. Používa semienka ako vstup náhodného generátora čísel 202452236, 397546757, 82391797 a nasledovné obtiažnosti d: 0,5; 1,0; 1,5. V experimente sa dbá na počet vygenerovaných nepriateľských kreatúr a či náležia obtiažnostnej krivke zobrazenej na obrázku 3.2.

Tretí test prvý experiment			
Čas (t) v sekundách	$d = 0,5$	$d = 1,0$	$d = 1,5$
$t = 0$	0	0	0
$t = 20$	2	3	4
$t = 40$	5	8	11
$t = 60$	8	14	20
$t = 80$	12	22	32
$t = 100$	17	32	47

Test bol úspešný, výsledné počty nepriateľov korešpondujú s obtiažnostnou krivkou na obrázku: 3.2.

Testovanie správneho správania herného AI bolo vykonané počas tvroby demonštračného videa, ktoré ukazuje správne správanie pri všetkých stavoch herného AI.

Zhodnotenie možnosti publikovania hry je možné vykonať pomocou nasledujúcich bodov:

- Výber platformy publikovania.
- Porovnanie hry s podobnými riešeniami s totožnými žánrami.
- Klasifikácia hry, teda jej zaradenie do fázy vývoja.

Podľa návrhu je hra zameraná na platformu Steam, ale existuje aj jej konkurent EpicStore. Obidve platformy obsahujú hry všetkých žánrov a kategórií, publikovaná hra však musí dodržiavať štandardy platformy a musí byť v hrateľnom stave. Obidve platformy požadujú

poplatok pri publikovaní. Výsledná realizácia hry Atlas of Fortune je síce hrateľná, ale bez dodatočného ladenia a reklamy na platformách hra nedosiahne úspech. Platforma itch.io je určená pre hry realizované menšími skupinami vývojárov alebo samostatnými vývojármi. Hry publikované na platforme itch.io majú menší dosah. Publikovanie na platforme itch.io nevyžaduje poplatok a slúži ako spôsob získania spätnej väzby od hráčov počas vývoja. Platforma itch.io nemá štandardy pre dokončenosť hry.

Pred publikovaním hry na platformách Steam alebo EpicStore musí byť hra schopná konkurencie s podobnými riešeniami. Hra je porovnaná s podobným riešením **Forewarned**, stručne popísaným v podkapitole 2.5. Hra Atlas of Fortune má generovanie úrovní podobnej kvality ako Forewarned. Z pohľadu grafiky, modelov a zvukových efektov je jej kvalita nižšia. Hra Forewarned obsahuje aj viacero herných systémov, ktoré zážitok hráča posilňujú. Pokračovanie vo vývoji hry má teda nasledujúce ciele:

- Realizácia herného systému, ktorý hráčovi povoľuje postavu vylepšovať a mince získané počas úrovní uschovávať a využívať.
- Zavedenie systému ukladania hry.
- Realizácia bonusovej úrovne, ktorú je možno preskúmavať pri výbornom výkone hráča v úrovni.
- Získanie alebo vytvorenie grafických a zvukových zdrojov a ich následovné začlenenie do hry.

Hra je pevne začlenená do fázy **Alfa**, fáza je popísaná v podkapitole 2.1. Hru teda je možné publikovať na platforme itch.io pre účely pokračovania vývoja hry a pravidelného získavania spätnej väzby od hráčov. Dosiahnutie cieľov znamená pripravenosť hry na publikovanie platformami Steam alebo EpicGames.

Kapitola 5

Záver

Úlohou tejto práce bolo preskúmať spomenuté techniky vývoja hier, následne vytvoriť návrh hry, ktorá vyhovuje žánrom adventure a exploration, návrh realizovať pomocou preskúmaných techník vývoja hier a nakoniec výsledok demonštrovať videom.

Spôsoby procedurálneho generovania sú vysvetlené pomocou využitia gramatík, algoritmu kolaps vlnovej funkcie a generátorov náhodných čísel. Ďalej je popísaný spôsob implementácie herného AI v podobe statického automatu a spôsob pohybu herného AI pomocou navigačnej mriežky.

Spôsob návrhu herných systémov je riešený vo forme herných cyklov a tvorbou interakcie medzi nimi a hráčom, čo formuje záujem, motiváciu a zábavu. Návrh obtiažnostnej krivky rieši réžiu intenzity hry, ktorá musí zapadať do určitej rovnováhy, aby hráča nenudila alebo nefrustrovala. Spôsob plnenia úrovne je predstavený tak, aby v hráčovi vzbudzoval túžbu úroveň preskúmať.

Nakoniec sú prezentované detaily realizácie návrhu s vysvetlením spôsobu prispôsobenia algoritmu kolaps vlnovej funkcie na využitie v 3D prostredí, v hernom engine Unity a pre teoreticky nekonečnú mriežku, v ktorej by obvyklý algoritmus bežal donekonečna. Kombináciou upraveného algoritmu a ostatných spôsobov procedurálneho generovania je dosiahnutá kompletná úroveň.

Výsledok práce je realizovaná hra, obsahujúca procedurálne generované úrovne, ktoré úspešne demonštrujú úplnosť a logickosť a sú plnené objektmi, predmetmi, pascami a nepriateľskými kreatúrami. Úrovne obsahujú interaktívne predmety a objekty, ktoré hráč využíva. Počet vygenerovaných objektov nekorešponduje s pravdepodobnosťou uvedenej v návrhu, sú ale stále generované v počte, ktorý hráčovi umožňuje úroveň dokončiť. Pasce sa v úrovniach nachádzajú v správnom počte, je s nimi možné vykonávať interakcie aj nepriateľskými prvkami aj hráčom. Úrovne obsahujú nepriateľské prvky v počte, ktorý sa zhoduje s obtiažnostnými krivkami, hra má teda správne regulovanú obtiažnosť. Nepriateľské kreatúry realizované stavovými automatmi sú schopné interagovať s predmetmi v úrovni aj s hráčom v správnej postupnosti. Hru nie je vhodné publikovať na platformách Steam alebo EpicStore. Je však vhodná pre publikovanie na platforme itch.io kde môže byť získaná spätná väzba používateľov. Výsledok je možné v budúcnosti rozširovať o viacero herných systémov ako sú hádanky, rozšírenie typov nepriateľov používajúcich pokročilejšie spôsoby realizácie hernej umelej inteligencie, pridanie perzistentných prvkov medzi úrovňami ako vylepšenia a pod.

Literatúra

[1] Alaka, S. a Bidarra, R. Hierarchical Semantic Wave Function Collapse. In: *Proceedings of the 18th International Conference on the Foundations of Digital Games*. New York, NY, USA: Association for Computing Machinery, 2023, s. 1–2. FDG '23. ISBN 9781450398558. Dostupné z: https://doi.org/10.1145/3582437.3587209.

[2] Allen Kent, J. G. W. *Encyclopedia of Computer Science and Technology*. CRC Press, október 1991. ISBN 978-0-8247-2275-3.

[3] Blaschek, G. *Object-Oriented Programming: with Prototypes*. Springer Berlin Heidelberg, 2012. ISBN 9783642780776. Dostupné z: https://books.google.sk/books?id=akKqCAAAQBAJ. Citovaná strana 11.

[4] Golodetz, S. Automatic Navigation Mesh Generation in Configuration Space. *Overload Journal*. ACCU, October 2017, zv. 117. Dostupné z: https://members.accu.org/index.php/articles/1838.

[5] Linden, R. van der; Lopes, R. a Bidarra, R. Procedural Generation of Dungeons. *IEEE Transactions on Computational Intelligence and AI in Games*, 2014, zv. 6, č. 1, s. 78–89.

[6] Lugrin, T. Random Number Generator. In: Mulder, V.; Mermoud, A.; Lenders, V. a Tellenbach, B., ed. *Trends in Data Protection and Encryption Technologies*. Cham: Springer Nature Switzerland, 2023, s. 31–34. ISBN 978-3-031-33386-6. Dostupné z: https://doi.org/10.1007/978-3-031-33386-6_7.

[7] Meduna, A. *Automata and Languages: Theory and Applications*. 1. vyd. Božetechova 2, Brno 61266, Czech republic: Springer London, July 2000. 707-740, 269-357, 145 s. ISBN 978-1-85233-074-3.

[8] Millington, I. *Game Physics Engine Development*. First. Springer Berlin Heidelberg, 2007. ISBN 9780429178559. Dostupné z: https://www.r-5.org/files/books/computers/algo-list/realtime-3d/Ian_Millington-Game_Physics_Engine_Development-EN.pdf. Citovaná strana 3.

[9] Perlin, K. An image synthesizer. In: *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*. New York, NY, USA: Association for Computing Machinery, 1985, s. 287–296. SIGGRAPH '85. ISBN 0897911660. Dostupné z: https://doi.org/10.1145/325334.325247.

[10] Ralph E. Johnson, B. F. Designing Reusable Classes. In: *Journal of Object-Oriented Programming*. University of Illinois, August 1991. Dostupné z: https://www.cse.msu.edu/~cse870/Input/SS2002/MiniProject/Sources/DRC.pdf.

[11] Rollings, A. a Adams, E. *Andrew Rollings and Ernest Adams on Game Design*. New Riders Games, 2003. ISBN 1592730019.

[12] Sarkar, A. a Cooper, S. Transforming Game Difficulty Curves using Function Composition. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2019, kap. 1. CHI '19. ISBN 9781450359702. Dostupné z: https://doi.org/10.1145/3290605.3300781.

[13] Sellers, M. *Advanced Game Design: A Systems Approach*. Addison-Wesley, 2018. 51,53,156,164,165 s. Game Design Series. ISBN 9780134667607. Dostupné z: https://books.google.sk/books?id=KcooswEACAAJ.

[14] Valencia García, R.; Lagos Ortiz, K.; Alcaraz Mármol, G.; Del Cioppo, J.; Vera Lucio, N. et al. *Technologies and Innovation*. Springer, 2016. ISBN 978-3-319-48024-4.

[15] Yannakakis, G. N. Game AI revisited. In: *Proceedings of the 9th Conference on Computing Frontiers*. New York, NY, USA: Association for Computing Machinery, 2012. CF '12. ISBN 9781450312158. Dostupné z: https://doi.org/10.1145/2212908.2212954.

Príloha A

Obsah zdielaného súboru NextCloud

Prílohy poskytnuté pomocou NextCloud je možné nájsť v následujúcich súboroch:

- Priečinok **Atlas of Fortune** obsahuje všetky zdrojové kódy potrebné pre kompiláciu projektu pomocou editoru Unity.
- Priečinok **Build** obsahuje skompilovanú a spustiteľnú verziu hry Atlas of Fortune.
- Priečinok **Docs** obsahuje manuála používateľskú príručku.
- Priečinok **Demo video** obsahuje demonštrančné video.
- **README.md** obsahuje všeobecný rozbor štruktúry súboru **Atlas of Fortune** a stručný popisok zdrojových kódov.
- Priečinok **Overleaf source** obsahuje všetky zdrojové súbory písomnej časti, ktorá bola vytvorené prostredníctvom stránky Overleaf.