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

FAKTOR STRACHU U HERNÍ UMĚLÉ INTELIGENCE

FEAR FACTOR OF GAMING ARTIFICIAL INTELLIGENCE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **MATEJ MIŠTÍK**
AUTHOR

VEDOUCÍ PRÁCE **Ing. TOMÁŠ CHLUBNA**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2021/2022

Zadání bakalářské práce

Student: **Mištík Matej**
Program: Informační technologie
Název: **Faktor strachu u herní umělé inteligence**
Fear Factor of Gaming Artificial Intelligence
Kategorie: Počítačová grafika

Zadání:

1. Seznamte se s vývojovým prostředím herních enginů (Unity, Unreal Engine...).
2. Vyberte a nastudujte herní mechaniky potřebné k demonstraci projevů strachu u herních nehratelných postav.
3. Navrhněte aplikaci demonstrující vybrané mechaniky.
4. Demo aplikaci implementujte.
5. Proveďte měření, případně uživatelskou studii hodnotící implementované výsledky.
6. Vytvořte video reprezentující výsledky vaší práce.

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

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3, experimenty vedoucí k bodu 4.

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

Vedoucí práce: **Chlubna Tomáš, Ing.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 11. května 2022
Datum schválení: 1. listopadu 2021

Abstrakt

Cieľom tejto práce je prezentovať faktor strachu pomocou hernej umelej inteligencie. Práca sa zameriava na interakciu hráča s umelou inteligenciou, ktorej faktor strachu je riešený pomocou vyhodnocovania komplexných podmienok a následného výberu stavu chovania. Vytvorený systém funguje pre boj a útek umelej inteligencie. Prínosom tejto práce je zavedenie ľudskej emócie, a to strachu, hernej AI pre herný engine Unity.

Abstract

The goal of this thesis is to present the fear factor by gaming artificial intelligence. The work focuses on the player's interaction with artificial intelligence, whose fear factor is addressed by evaluating complex conditions and the subsequent selection of the state of behaviour. The created system works for combat and escape of artificial intelligence. The outcome of this thesis is the implementation of human emotion, mainly the fear for gaming artificial intelligence in the enviroment of Unity engine.

Kľúčové slová

Simulácia, Hra, Unity, C#, FPS, Umelá Inteligencia, UI, Faktor Strachu

Keywords

Simulation, Game, Unity, C#, FPS, Artificial Inteligence, AI, Fear Factor

Citácia

MIŠTÍK, Matej. *Faktor strachu u herní umělé inteligence*. Brno, 2022. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Tomáš Chlubna

Faktor strachu u herní umělé inteligence

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Tomáše Chlubny. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Matej Mištík
17. mája 2022

Poďakovanie

Chcel by som poďakovať vedúcemu práce Ing. Tomášovi Chlubnovi za čas a rady, ktoré mi vždy poskytol. Pomohol mi usmerniť cestu práce a vždy mal pre mňa profesionálnu kritiku toho, čo je nutné zlepšiť. Takisto chcem poďakovať všetkým respondentom štúdie za vyplnenie dotazníkov. A v neposlednom rade hercovi, ktorý stvárnil Rockyho Balbou a motivoval ma nezastaviť sa celý semester, Sylvesterovi Stallonovi.

Obsah

Kapitola 1

Úvod

Táto práca sa zameriava na riešenie problematiky prejavov emócie strachu umelej inteligencie. Riešenie tejto problematiky je implementované za pomoci prechodov medzi stavmi chovania danej umelej inteligencie. Stavy musia byť samostatné pracujúce celky, ktoré určia jasné chovanie v konkrétnej situácii podľa vyhodnotení určitých podmienok. Následne môže umelá inteligencia podľa stavov vykonať špecifické užívateľom zadané úlohy. Hráč aktivuje ich chovanie v špecificky pripravených herných scénach so zoznamom úloh, ktoré je treba vykonať pre zobrazenie.

Strach ako emócia je v práci vyobrazená práve cez chovanie umelej inteligencie, cez vyobrazenie ukazovateľov, ktoré jasne naznačujú na akej úrovni sa strach práve nachádza, a krátke textové správy. Ale strach je komplexná emócia. Je to zložitá úloha, ktorá núti skúmať človeka a jeho správanie. Čo sa deje v tele a ako je to možné vyobraziť dokonalo pre obyčajného pozorovateľa, aby dokázal pozitívne potvrdiť, že subjekt pociťuje umelo vyobrazený strach.

Problémom je vyobrazenie emócii a na to sú potrebné vysoko úrovňové emócie tváre s dokonalou grafikou a dnešné technológie využívajú nástroje, ktoré zachytávajú mimiku tváre pomocou pripravených bodoch na tvárach hercov, ktoré sú prevedené ako animácia do počítačového programu. Strach, takisto, znamená, že telo reaguje aj inými spôsobmi a tie môžu byť vyobrazené za pomoci úpravy herných prvkov nehrateľných postáv. Ako napríklad znížená vízia alebo roztiahnutie svalov, čo môže znamenať zrýchlenie atribútu rýchlosti takejto postavy.

Výsledkom práce je herné demo v Unity, ktoré demonštruje využitie strachu u počítačom ovládaných postáv na špecificky navrhnutých scénach, ktoré demonštrujú určitý scenár. Sledovanie chovania nehrateľných postáv je v deme z pohľadu hráča prvej osoby[1] a vyžaduje určitú interakciu na základe slovne zadanej úlohy na začiatku každej scény.

Každopádne, táto práca rozoberá práve tieto programátorské úlohy a začína **teóriou strachu a ľudských reakcii** naň v kapitole 2, následne poskytne náhľad do **teórie mechaník** v kapitole 3 využitých v práci. Práca poskytuje takisto **návrh** 4 simulácie a konceptu hry. A v neposlednom rade zobrazí princípy implementovania najdôležitejších časti simulácie kapitole **implementácia** 5. Nakoniec poskytne prehľad **experimentov** v kapitole 6 z herných scén na základe užívateľom nastaviteľných konštánt, ktoré ovplyvňujú tvorbu strachu a chovanie UI.

[1] Kamera je vložená na úrovni hračovych očí

Kapitola 2

Emócie strachu a jej reakcie

Práca sa zaoberá pojmom strach a mieri na jeho prejavy u umelej inteligencii. Pre implementáciu tejto emócie je nutné vedieť zodpovedať nasledovné:

- Čím sa vyvolá strach?
- Aké sú reakcie naň?
- A čo strach spôsobí?

Je potrebné vedieť odpovede na všetky z týchto otázok, aby bolo možné vytvoriť čo najreálnejšie zobrazenie strachu u umelej inteligencie. Hráč musí poznať, že ide o strach a zároveň ho musí vedieť použiť vo svoj prospech.

2.1 Strach

Strach ovplyvňuje správanie a rozhodovanie sa mnohých zvierat a rôznych živočíšnych druhov. Strach u ľudí je vyvolaný z rôznych faktorov, ale práca sa zaoberá hlavne strachom spôsobeným nebezpečím a jeho odpovednú obrannú mechaniku a nie napríklad strach spôsobený nervozitou. Dôležité je poznať, ako človek zareaguje pri konkrétnych situáciách pre zachovanie čo najväčšej výpovednej hodnoty. Z konkrétnej štúdie o strachu [5] pri 12 rôznych scenároch uskutočňovaných na dospelých mužoch, tabuľka 2.1, sa využije poradie reakcií podľa priority na implementáciu ich stromu chovania. Reakcie, a ich podrobnejšie vysvetlenie, sú zoradené podľa najčastejšieho výberu mužov zo štúdie [5] od hora dole:

1. Útek - nie je pod držaním nepriateľa, ale rozhodne sa utiecť
2. Pokus o únik - ak sa nepodarilo utiecť, nasleduje situácia, kde je subjekt pod kontrolou niekoho iného a snaží sa o útek alebo oslobodenie sa
3. Skrytie sa - vyžaduje nájdenie úkrytu s dostatočným priestorom na krytie
4. Kontrola situácie - skontrolovanie, čo sa deje okolo subjektu na celých 360 stupňov
5. Prieskum/priblíženie - priblíženie sa k nebezpečiu a zároveň v niektorých scenároch využitie komunikácie na objasnenie, čo stojí za nebezpečím

Scenár/prostredie	reakcia
Krovie	Útok alebo uvoľnenie sa z držania nepriateľom
Výťah	Útok alebo uvoľnenie sa
Semafor	Útek, pokus o únik, skrytie sa
Prenasledovanie	Útek, pokus o únik, skrytie sa
Roh budovy	Kontrola situácie, priblíženie alebo prieskum
Známosť	Kontrola situácie, priblíženie alebo prieskum
Park	Útek, pokus o únik, skrytie sa
Uchopenie niekým	Kontrola situácie, priblíženie alebo prieskum
Neznámy zvuk	Kontrola situácie, priblíženie alebo prieskum
Zvuk telefónu	Nájdenie niečoho, čo sa použije ako zbraň
Bomba	Útek, pokus o únik, skrytie sa
Šepot	Kontrola situácie, priblíženie sa alebo prieskum

Tabuľka 2.1: Poradie reakcii na situácie, kde rozhoduje strach u mužov zo štúdie [5]

2.2 Ako vyvolať strach

Strach v riešení vyvolá hráč, a preto je potrebné mať nástroje, ktorými to dokáže. Scenáre v tabuľke 2.1 sú vhodnými prostriedkami pre demonštráciu. Strach môže byť vyvolaný násilím, zahnaním do kúta, naháňaním alebo prenasledovaním. Takisto mimo scenárov môže byť využitých niekoľko akcii, ktoré to narušia. Príkladom môže byť zníženie viditeľnosti, neznámy zvuk. Z hľadiska konfrontácie útočníka ovplyvnia tvorbu strachu jeho parametre, ako napríklad výška, svalnatosť, postoj, oblečenie. Takisto je obmedzená tvorba strachu u človeka na základe jeho predošlých skúsenosti so situáciou, jeho všeobecných paranojí alebo fóbii, psychikou z hľadiska odvahy a v neposlednom rade jeho fyzickým výzorom a presvedčením o jeho sile. To koľko strachu vyvolá u ľudí iný jedinec, je reakcia hlboko inštinktívna, ktorá funguje u mäkkýšov už 350 miliónov rokov [1] z ich hierarchie dominancie. Ich prenos zdravia a sociálneho rebríčka síce prenášajú hormonálne pred súbojom, ale práve tieto informácie rozhodnú či boj vôbec začne. A tým pádom aj u ľudí, je strach vyvolávaný už na základe výzoru a správania druhého človeka. Neočakávaná reakcia vyvoláva najväčšie pochybnosti a väčšinu z týchto faktorov je potrebné zahrnúť do simulácie.

2.3 Účinky strachu a reakcie na situáciu

Pri pocítení strachu sa všeobecne mozog riadi stratégiou „bojuj alebo uteč“, pričom ak nie je schopný sa rozhodnúť, ľudské telo takzvane zamrzne a nevie sa pohnúť. Následné body sú najčastejšie reakcie podľa štúdie [3]:

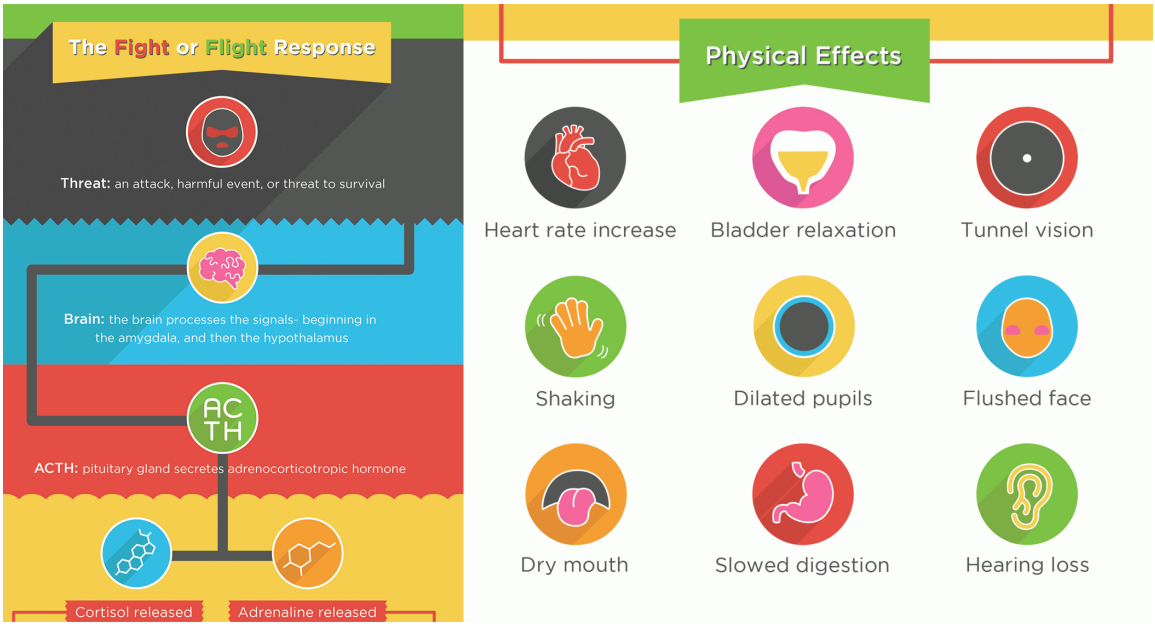
1. Boj - eliminovať cieľ alebo vytvoriť cestu na útek
2. Útek - opustiť čo najrýchlejšie zónu nebezpečia
3. Prosenie o milosť - presvedčenie cieľu s cieľom vyhnúť sa nebezpečenstvu
4. Zamrazenie - znemožnené psychické alebo fyzické akcie s vnímaním situácie
5. Odpadnutie - úplná strata vedomia

2.4 Reakcia tela na strach

Podľa odborného článku Tim Newmana [7], pri vzniku nejakej hrozby reaguje mozog a signalizuje, aby hypofýza [2] sekrétovala do krvi hormón ACTH (adrenokortikotropný hormón). Medzi tým podporný nervový systém, ktorý je zodpovedný za riadenie reakcie bojuj alebo uteč, dá znamenie nadobličke, aby vyplavila hormóny adrenalín a katelochamíny[1], ktoré spôsobia rôzne fyzické efekty na človeku. Vďaka ACTH je v krvi aj zvýšená prítomnosť kortizolu [10], ktorý má vplyv na zvýšenie tlaku krvi, cukru a bielych krviniek. Výhodou je, že cortisol v tejto situácii premieňa tukové kyseliny na rýchlo použiteľnú energiu pre svaly. Katelochamíny takisto zaistia aby boli svaly pripravené na extrémnu kontrakciu, zamedzilo sa ich poškodeniu a využilo ich plný potenciál. Tieto hormóny môžu takisto spôsobiť:

- Zvýšenú aktivitu srdca a pľúc
- Znížený aktivitu žalúdka, ktorá môže predstavovať pocit motýľov v bruchu
- Zastavenie produkcie slín a sĺz, čo spôsobuje pocit suchých úst
- Čiastočnú stratu sluchu

Podľa štúdii [3], strach takisto pôsobí na telo rýchlym tepom srdca a dýchaním, aby získalo, čo najviac kyslíku do svalov pre prípadný útek alebo boj. Zreničky oka sa roztvoria pre získanie, čo najväčšieho obzoru nad situáciou. Človek sa začína potiť a jeho ruky chladnúť, kvôli odčerpanej krvi z koncových častí teľa do torza. Celý proces je viditeľný na obrázku 2.1. Spomenuté reakcie sú vhodné pre demonštráciu a zviditeľnenie strachu hráčovi, napríklad za pomoci špeciálneho umožneného videnia tepu srdca alebo obmedzenia videnia umelej inteligencie.



Obr. 2.1: Proces vykonávaný telom pri pocítení strachu[2]

[1] Hormony produkované pri pocítení emočného alebo psychického nátlaku [8]
[2] Prevzaté z: https://hmn.wiki/sk/Fight-or-flight_response

Kapitola 3

Herné Mechaniky

Nasledujúca kapitola rozoberá najdôležitejšie mechaniky použité v simulácii faktoru strachu. Najdôležitejšia je teória stromov chovania 3.1, ktoré sú hlavným stavebným prvkom pri riadení prechodov medzi stavmi chovania umelej inteligencie. Následne je dôležité vývojové prostredie Unity 3.2.1, kde je nevyhnutné pochopiť, hlavne čo je to herný objekt 3.4, navigačná plocha 3.2.1 pre umelú inteligenciu a herná scéna 3.5. Samostatný vizuálny prvok Animation rigging 3.3 pomáha k reálnemu výzoru hernej postavy pri držaní objektov alebo vierohodnému ohýbaniu končatín u nehrateľných postáv. Nakoniec, kapitola poskytne náhľad na vnímanie sveta pomocou senzorov 3.6, ktoré je nevyhnutné pre registrovanie ostatných herných objektov v scéne pre UI.

3.1 Stromy chovania

Strom chovania je matematický model uskutočňovania akcii a prepínania medzi konečným počtom stavov v uzloch umelej inteligencie usporiadaných v stromovej štruktúre [6]. Jeho hlavnými výhodami sú udržateľnosť, znovu použiteľnosť a reaktívnosť. Inými slovami, jednoduché pridávanie uzlov umožňuje efektívnu zmenu chovania UI agenta. Zároveň je možné znovu použiť kód iného stromu chovania, v novom strome chovania bez špecifikácie akéhokoľvek vzťahu medzi nimi. Dôvod použitia stromu chovania pred konečným automatom je v tom, že prechody stromu chovania sú na rozdiel od konečného automatu, definované štruktúrou a nie podmienkami v jednotlivých stavoch [4]. Tento problém vidno, keď je v konečnom automate jedna z komponent odstránená čo spôsobí, že musí byť každý prechod do a z nej ošetrení. Zatiaľ čo odstránenie uzla v strome chovania spôsobí iba vynechania nejakej časti implementovaného chovania, a to umožňuje jednoduché zmeny. Tento model takisto vytvára čitateľnú štruktúru, kde z jedného veľkého stromu je možné vytvoriť mnoho menších podstromov a zvýši prehľadnosť chovania.

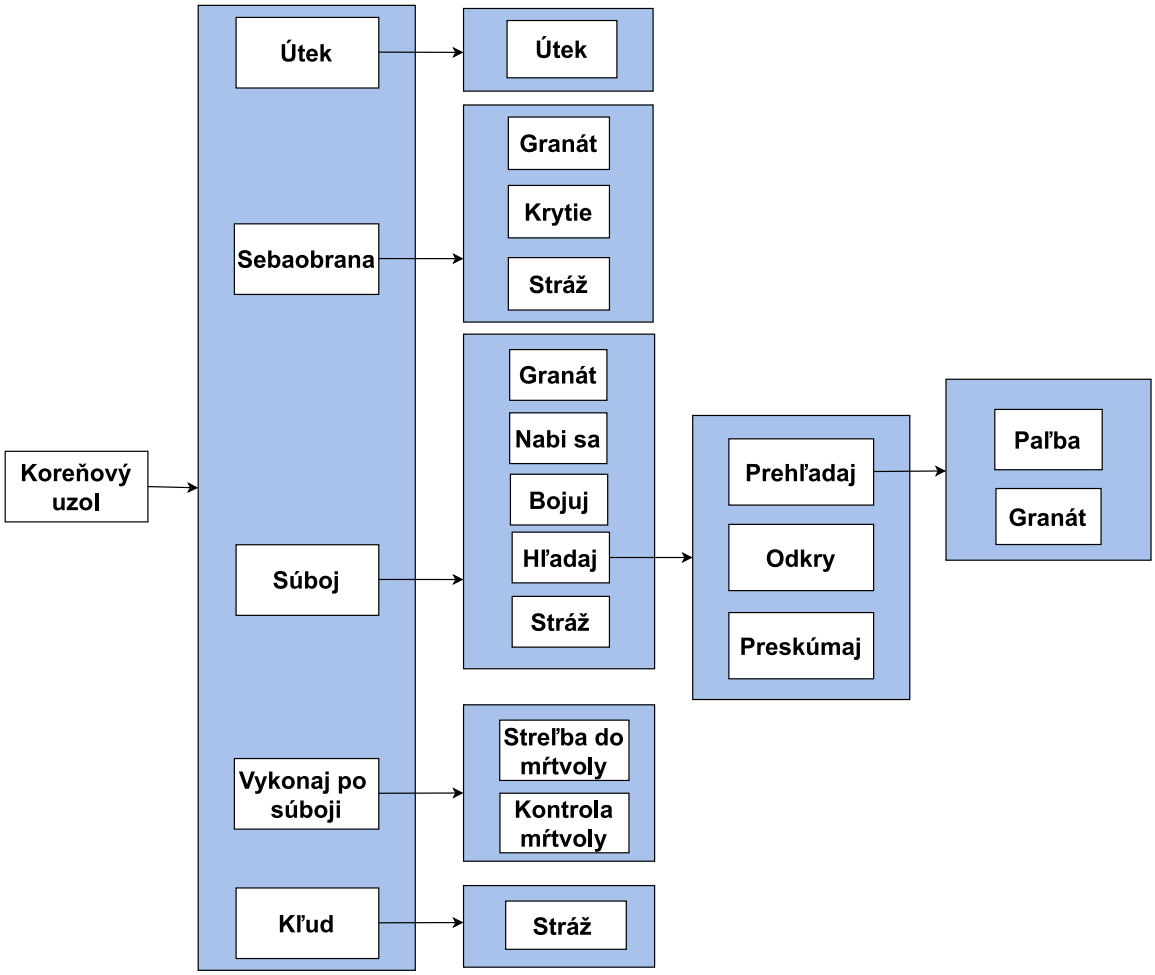
Problémom tohoto modelu je, že vždy začína od koreňového uzlu, čo pri hlbokom strome môže spôsobovať problémy, keďže sa vo vývoji ešte môže rozširovať.

Za vznikom stromu chovania stoja herní vývojári z Bungie Studios, ktorí ich implementovali do hernej série Halo[1]. Ich abstraktný model stromu chovania z hry Halo 2 je vidieť na obrázku 3.1, kde sa úlohy vykonávajú od spod na hor. Tento uľahčujúci plán pre umelú inteligenciu je prenositeľný mimo herný priemysel a podporuje armádne drony a správanie samostatných robotov.

[1] https://www.halowaypoint.com/en

Dôležitá je štruktúra stromu skladajúca sa z uzlov: koreňového, kontrolér toku udalostí a uskutočniteľ úlohy. Strom je spracovaný smerom zľava doprava. Z koreňového uzlu sa podľa aktuálneho stavu odosiela informátor o tom či sa môže uskutočniť úloha uzlu uskutočniteľa úlohy a posiela naspäť stav vykonanej úlohy, a to úspech, beh alebo zlyhanie (success, running, failure).

- Koreňový uzol je vrcholom stromu, nemá rodičovský uzol a má pod sebou 1 až N uzlov, kde N patrí prirodzeným číslam.
- Kontrolér toku udalostí má jeden rodičovský uzol a minimálne jeden detský uzol uskotočnitela udalostí.
- Uskotočniteľ úlohy je listovým uzlom s jediným rodičom.



Obr. 3.1: Strom chovania zobrazujúci štruktúru správania základného nepriateľa hry Halo2.

V klasickej formulácii [6] stromu chovania existujú štyri kategórie uzlov toku udalostí, a to Postupnosť, Výber, Paralelná a Dekoračná. (Sequence, Fallback, Parallel a Decorator) a dve kategórie uzlov uskutočnenia akcia a podmienka (Action and Condition). Strom chovania vykonáva úlohy z koreňového uzlu, ktorý generuje signály, ktoré povoľujú vykonanie

úlohy synovského uzlu Tick s dodanou frekvenciou, ktoré sú poslané jeho synovským uzlom. Synovské uzly sa teda vykonajú, len ak príjmu signál od uzlu Tick.

Sequence node / uzol postupnosti

Uzol postupnosti vyžaduje, aby každý z jeho synovských uzlov vrátil stav success a až vtedy môže sám vrátiť success, v ostatných prípadoch vracia stav failure alebo running. Zápis pseudokódom 3.1 je možný nasledovne:

```
for i <- 1 to N do
    childStatus <- Tick(child(i))
    if childStatus = Running then
        return Running
    else if childStatus = Failure then
        return Failure
return Success
```

Výpis 3.1: Pseudokód uzlu postupnosti.

Selector node / uzol výberu

Uzol výberu vyžaduje, aby aspoň jeden z jeho synovských uzlov vrátil stav success a až vtedy môže sám vrátiť success, v ostatných prípadoch vracia stav failure alebo running. Zápis pseudokódom 3.2 je možný nasledovne:

```
for i <- 1 to N do
    childStatus <- Tick(child(i))
    if childStatus = Running then
        return Running
    else if childStatus = Success then
        return Success
return Failure
```

Výpis 3.2: Pseudokód uzlu výberu.

Parallel node / uzol súbežný

Súbežný uzol vykonáva algoritmus 3.3. Uzol vracia success ak M synovských uzlov vráti success. Vracia hodnotu failure ak $N - M + 1$ synovských uzlov vráti failure. V iných prípadoch vracia hodnotu running, kde N je počet synovských uzlov a M, $M \geq N$, je používateľom nastavená hranica.

```
for i <- 1 to N do
    childStatus(i) <- Tick(child(i))
    if SUM(i):childStatus(i)=Success1 >= M then
        return Success
    else if SUM(i):childStatus(i)=Failure1 > N - M then
        return Failure
return Running
```

Výpis 3.3: Pseudokód uzlu súbežného.

Decorator node / Dekorátor

Dekorátor je uzol s jediným synovským uzlom, ktorý manipuluje s návratovou hodnotou podľa užívateľom nastavených pravidiel a takisto vracia stav podľa nejakého pred definovaného pravidla. Príkladom môže byť algoritmus 3.4 typu Inverter, ktorý vracia opačný stav uzlu, ak sa do času T nezmení status uzlu. Môže teda zabrániť zacykleniu umelej inteligencii pri vykonávaní niektorej z úloh. Jedná sa teda o špeciálny užívateľovom vytvorený uzol, ktorý riadi tok udalostí a chráni pred zacyklením chovania.

```
for i <- 1 to N do
    childStatus <- Tick(child(i))
    if childStatus = Running then
        return Running
    else if childStatus = Success then
        return Failure
return Success
}
```

Výpis 3.4: Pseudokód uzlu dekorátor - typ inverter.

Action node / uzol akcie

Keď sa vstúpi do uzlu akcie, vykoná špecifickú akciu. Vracia success ak je jej úloha splnená alebo failure ak je úloha neumožiteľná. Zatiaľ čo sa akcia prevádza, uzol vracia hodnotu running. Uzol akcie je vyobrazený v pseudokóde 3.5.

```
Function Tick()
    DoAStepOfComputation()
    if action-succeeded then
        return Success
    else if action-failed then
        return Failure
    else
        return Running
```

Výpis 3.5: Pseudokód uzlu akcia.

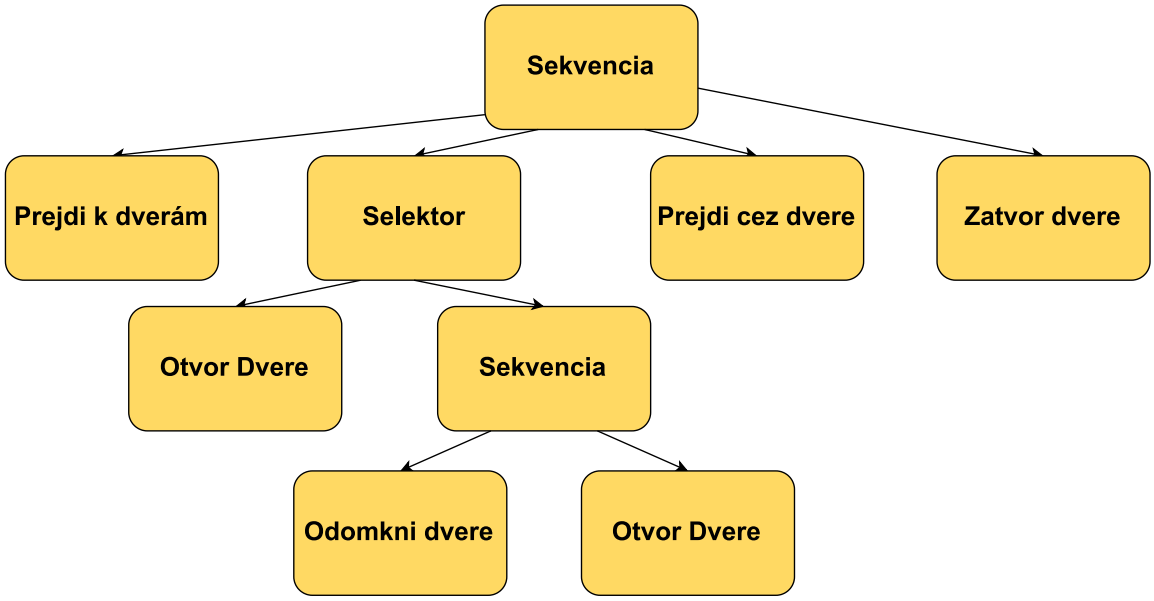
Condition Node / Uzol Podmienky

Uzol podmienky vyhodnocuje podmienku s tým, že vracia iba dva stavy uzlu a to pri úspechu success a pri neúspechu failure. Stav running sa pri podmienkovom uzle nevyužíva, na rozdiel od uzla akcie. Pseudokód 3.6 uzlu podmienky.

```
Function Tick()
    if condition-true then
        return Success
    else
        return Failure
```

Výpis 3.6: Pseudokód uzlu podmienka.

Uzle typu Sekvencia, Selektor a Akcie môžu vytvoriť jednoduchý strom ilustrovaný obrázkom 3.2, kde proces v uzle akcie **Prejdi k dverám beží** a teda odosiela stav running až pokým, nesplní vnútornú podmienku a odošle stav success a vyhodnocovanie môže pokračovať ďalším uzlom v sekvencii.



Obr. 3.2: Strom chovania pre prechod cez dvere. Nasledujúca sekvencia sa vykonáva zľava doprava až pokým uzol Zatvor dvere nevráti stav success a na to sú vyžadované success stavy od všetkých predošlých uzlov tej istej úrovne.

3.2 Herný engine

Herný engine je software slúžiaci k vývoju videohier. Pojem „herný engine“ sa začalo objavovať už v roku 1993 a to s hrou DOOM od id Software. Herný engine sa nazýval id Tech1 a v ňom bola hra veľmi racionálne navrhnutá s tým, že jadro bolo oddelené a tým pádom oddelené časti ako (vykresľovanie 3D grafiky, systém pre detekcie kolízii alebo samostatný audio systém) a herný svet, umelecké prvky alebo pravidla hry. Plný potenciál štúdio využilo, keď spustilo predaj tohoto enginu, kde tým pádom štúdiá, ktoré si ho kúpili, nemuseli stavať od nuly, ba dokonca stačilo niekedy iba pridanie artistických prvkov a nové originálne hry mohli vznikať na viac-menej univerzálnej platforme. Vyobrazenie levelu hry Doom na obrázku 3.3.

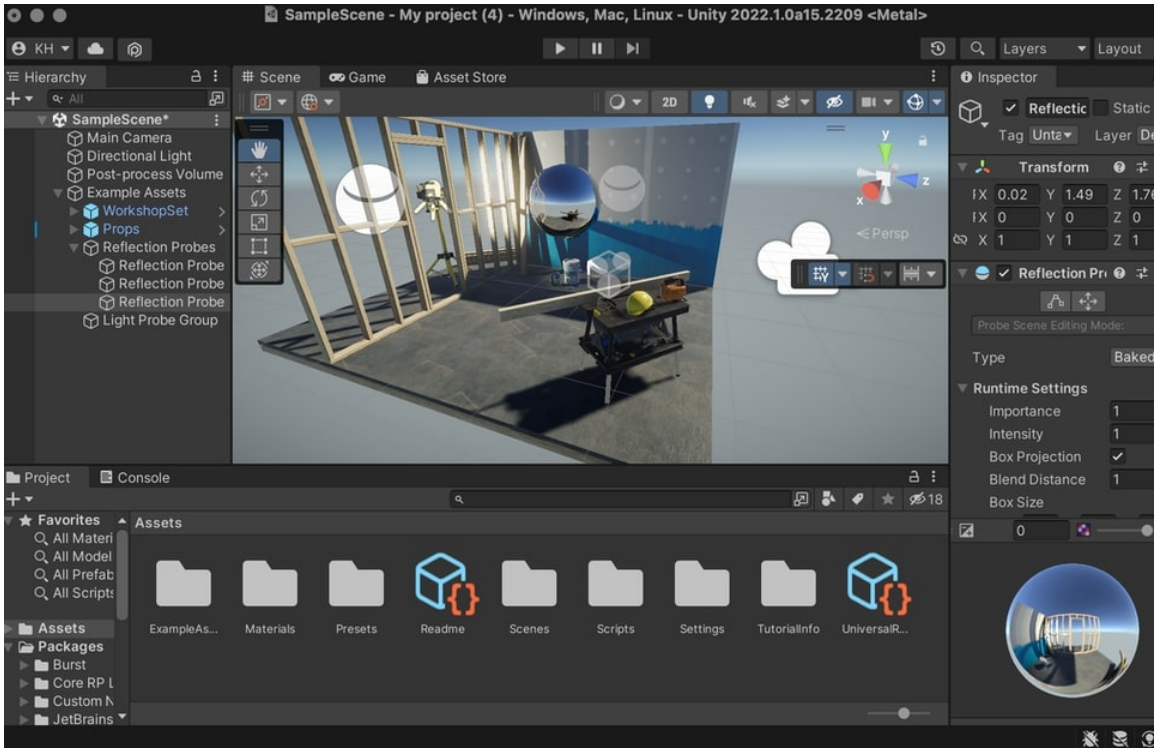
Obr. 3.3: Ukážka hry Doom vytvorenej v id Tech1[2]

3.2.1 Unity

Ide o herný engine, ktorý poskytuje vývoj pre široké spektrum platforiem. Napríklad Windows, mobilné hry, playstation alebo xbox. Univerzálnosť je pri Unity kľúčom a nezanedbateľná je aj jej skúsená a rozmanitá komunita, ktorá vyjadruje obrovskú podporu pri otázkach a odpovediach, ponúka užitočné vývojárske rady a tutoriály a v neposlednom rade je vždy inovatívna.

Engine, ale takisto ponúka veľmi pokročilý editor s hernou scénou a veľkým množstvom assetov určených pre vývoj zdarma. Jeho rýchlosť a grafický vizuál sa síce nevyrovná Unreal Engine, ale ide o jednoduchší engine určený pre vývoj, kvôli možnosti písania kódu v programovacom jazyku C♯ a takisto podporuje vstavané vývojové prostredie Visual Studio 2019. Herný editor enginu Unity vidieť na nasledovnom obrázku 3.4.

[2]Prevzaté z: https://www.imdb.com/title/tt0286598/



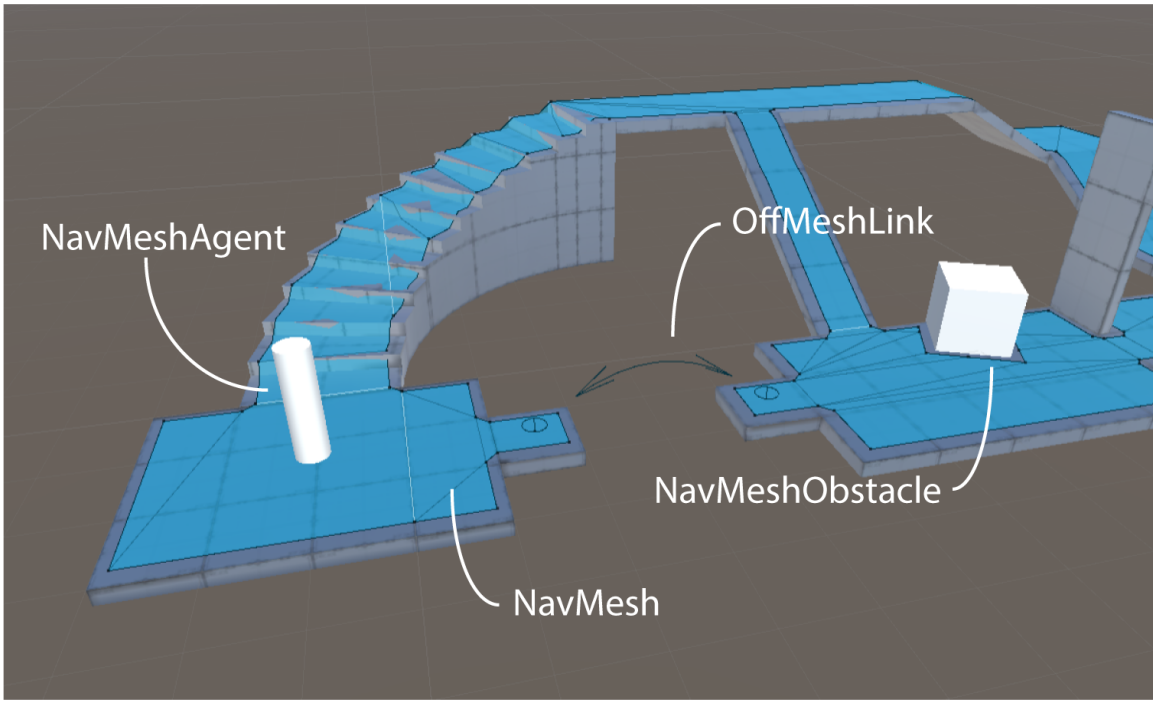
Obr. 3.4: Ukážka editoru enginu Unity

Prefab

Prefab systém, ktorý je súčasťou Unity, umožňuje vývojárovi vytvárať, konfigurovať a ukladať herný objekt (GameObject) 3.4 so všetkými jeho komponentami, vlastnosťami a ďalšími hernými objektami, ktoré sú potomkami tohoto objektu. Prefab funguje ako predloha, ktorá sa dá kopírovať/klonovať do herného sveta ako nový objekt, ktorý je modifikovateľný bez ovplyvnenia daného prefabu. Modifikácia prefabu sa naopak prejaví pri všetkých kópiách v hernom prostredí.

Navigačný systém Unity

Navigačný systém [11] v Unity podporuje vytvorenie navigačnej plochy, agenta, prekážok a mimo plošné spojenie navigačných plôch (**NavMeshAgent, NavMesh, NavMeshObstacle, OffMeshLink**). Hry v Unity majú možnosť vytvorenia navigačnej vrstvy (NavMesh), ktorá zaručuje nájdenie cesty a pohyb UI. **Navmesh** využíva toho, že určitá vrstva určená pre pohyb v hrách je návrhárom dopredu špecifikovaná a je jasné kde sa jednotlivé časti spájajú a kde je možné sa medzi nimi pohybovať, pričom samotná vrstva pre pohyb je tvorená z polygónov, ktoré sa môžu využiť k navigácii. Každý polygón nasledovne predstavuje bod v grafe a pokiaľ mali pôvodné polygony spoločnú hranu, majú aj ich body medzi sebou hranu. Vyhľadávanie cesty sa potom realizuje pomocou algoritmu A*. Príklad navmeshu vidno na obrázku 3.5.

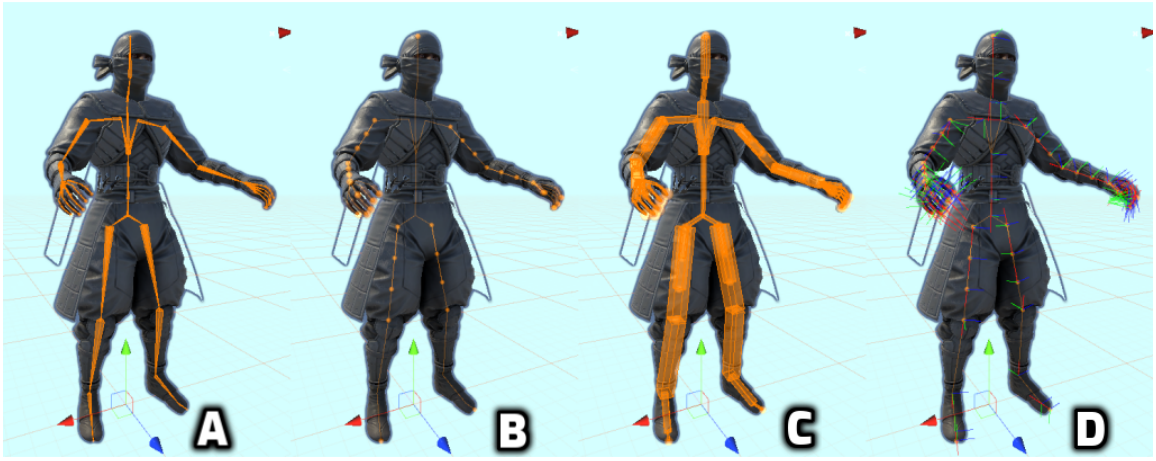


Obr. 3.5: Vygenerovaná plocha **NavMesh** - modrá plocha obopínajúca objekty v leveli umožňujúca pohyb **NavMeshAgenta**, **OffMeshLink** vykresľuje prechod medzi fyzicky nespojenými plochami a **NavMeshObstacle** má okolo seba šedú plochu, ktorá zabraňuje kolíziam s agentom[4]

3.3 Animation rigging

Animation rigging [11] je technológia, ktorá umožňuje vytváranie a organizáciu konštrukcií, ktoré ovládajú jednotlivé komponenty tela objektu. Tieto konštrukcie následne komunikujú s animačným API a umožňujú jednoduché vytváranie animácií s konkrétnymi časťami tela objektu ako s celkom. Využiteľný je napríklad pre animácie držania zbrane pre umelú inteligenciu a hlavne pre humanoidné postavy, keďže ponúka komponentu **Bone Renderer**. **Bone Renderer** umožňuje vykreslenie kosti humanoidného objektu. Hlavná komponenta je **rig builder**, ktorý je hlavným prvkom konštrukcie a následne sa pod neho vkladá zo zoznamu prvkov dostupných. Náhľad technológie pre **Bone Renderer** je zobrazený na obrázku 3.6.

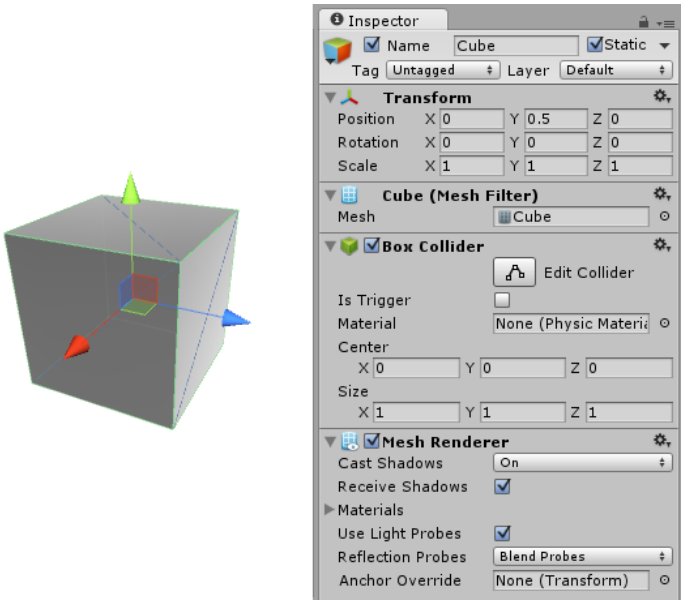
[4]Prevzaté z: https://docs.unity3d.com/Manual/nav-NavigationSystem.html



Obr. 3.6: Vykresľovanie kosti humanoidnej postavy: A značí základný render komponenty. B zobrazuje body ohybu v kostiach. C zobrazuje upravený obrázok A, kde je sa môže zmeniť veľkosť kostí. D zobrazuje pripevňovacie trojnožky na osiach, ktoré sa dajú modifikovať.[6]

3.4 Herné objekty Unity

GameObject [11] v preklade **Herný objekt** je základným a najdôležitejším konceptom editoru Unity. Každý objekt v hre je v Unity **GameObject**, ktorého úloha vyplýva až pri priradení komponentov na daný objekt. **GameObject** môže byť teda pokojne postava, osvetlenie až po kameru a špeciálne efekty. Jediná komponenta, ktorá je neodlučiteľná sa nazýva **Transform**. **Transform** komponenta slúži na určenie polohy, rotácie herného objektu. Vyobrazenie herného objektu v Unity 3.7.

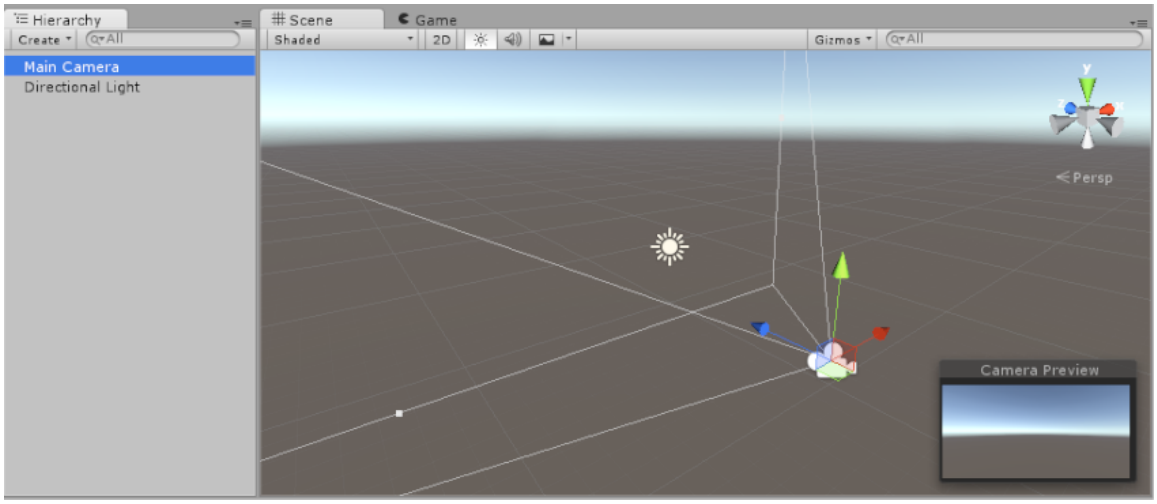


Obr. 3.7: Jednoduchý 3D herný objekt kocka zobrazená v editore[7]

[6] Prevzaté z: https://docs.unity3d.com/Packages/com.unity.animation.rigging\spacefactor\@m{}0.2/manual/index.html

3.5 Herná scéna

Scéna je priestor, do ktorého sa vkladajú všetky herné objekty a vytvárajú aplikáciu [11]. Väčšinou obsahuje celú časť aplikácie. Jedna scéna predstavuje pri komplexnejších hrách jeden level. V tejto práci vykresľuje jedna scéna práve jeden scenár strachu. Scéna obsahuje vrstvy, ktoré umožňujú separovanú interakciu herných objektov a teda ide o veľmi pokročilú a mocnú techniku v hernom svete. Scéna viditeľná na obrázku 3.8 .

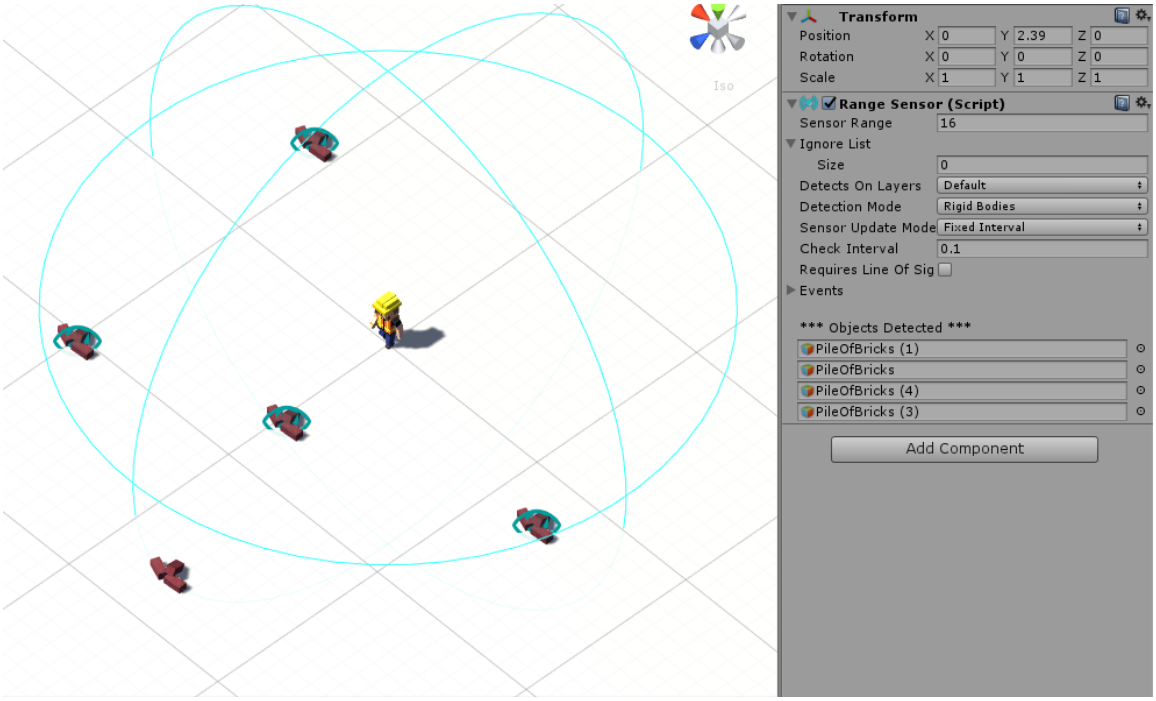


Obr. 3.8: Ukážka scény pri vytvorení nového projektu[8]

3.6 Videnie herného sveta umelou inteligenciou

UI agent musí, pre svoje pôsobenie v hernom svete, vedieť zachytiť rôznymi senzormi svet okolo seba a hlavne hráča. Jedným z hlavných senzorov je ten, ktorý deteguje herné objekty v určenej vzdialenosti. Spôsob detekcie je s využitím Unity metódy z triedy **Physics.OverlapSphereNonAlloc**, ktorá zozbiera všetky herné objekty, ktoré majú komponentu rigidbody a tým pádom vytvoria takzvaný collider, čiže zrážku v priestore tohoto senzoru. Senzor využíva 3D objektu guľa, ktorý má priemer, pozíciu a hernú vrstvu, na ktorej zaznamenané kolízie zapisuje do zásobníku kolízií. Konkrétne zobrazenie použitého senzoru je na obrázku 3.9.

[7]Prevzaté z: https://docs.unity3d.com/uploads/Main/GameObjectCubeExample1.png
[8]Prevzaté z: https://docs.unity3d.com/uploads/Main/NewEmptyScene_01.png



Obr. 3.9: Ukážka herného senzoru s parametrami v editore, ktorý detekuje objekty s komponentou **RigidBody** v dosahu modrých čiar tvaru gule. Všetky detekované objekty sú vypísane v **ObjectsDetected**[10]

[10]Prevzaté z: https://www.micosmo.com/sensortoolkit/images/RangeSensorExample.png

Kapitola 4

Návrh

Zámerom bolo navrhnúť hru/simuláciu z pohľadu prvej osoby v Unity, ktorá zviditeľní reakcie umelej inteligencie (ďalej iba UI), ktoré budú vysoko ovplyvniteľné jej strachom. Chovanie UI bude riadené pomocou viacerých samostatných skriptov, ktoré budú riadiť vždy jednu časť reakcii na chovanie hráča. Avšak, hlavné chovanie UI bude spočívať vo vyhodnocovaní stavov jej stromu chovania závislého od aktuálnej scény. Hra ale takisto závisí na hráčovi, ktorý bude musieť vedieť využiť svoje prostredie a ovplyvniť nepriateľskú UI na toľko, aby ju porazil. Hráč dostáva rozšírený pohyb v prostredí, prostriedky na zničenie nepriateľa a prehľad o jeho stave. UI dostáva prvky na útok či obranu a rozšírenú škálu reakcií na základe zmeny jej vnútorných premenných, ako napríklad strachu, zdravia.

4.1 Žáner a cieľ hry

Ide o simuláciu z prvej osoby, ktorá poskytuje akčné prvky boja s umelou inteligenciou a zároveň sa snaží o čo najpresnejšiu simuláciu ľudského strachu u UI. Nepriateľov je možné poraziť aj inak ako len zásahom ich tela. Hra v aktuálnom štádiu funguje na základe odoberaní bodov života UI, ktorá má zakomponovanú regeneráciu pod určitou hranicou, takisto má regeneráciu aj hráč. Druhý spôsob je možný, kde zvýšenie strachu je na úrovni, že daný nepriateľ upadne do bezvedomia a tým pádom je možné ho brať za eliminovaného.

Cieľ hry / simulácie

Cieľom hry bude eliminovanie nepriateľov s postupným prechodom cez level. U hráča je potrebné, aby využil čo najviac možností, ako využiť strach UI na jej porazenie. Využitie okolia na rozptýlenie UI.

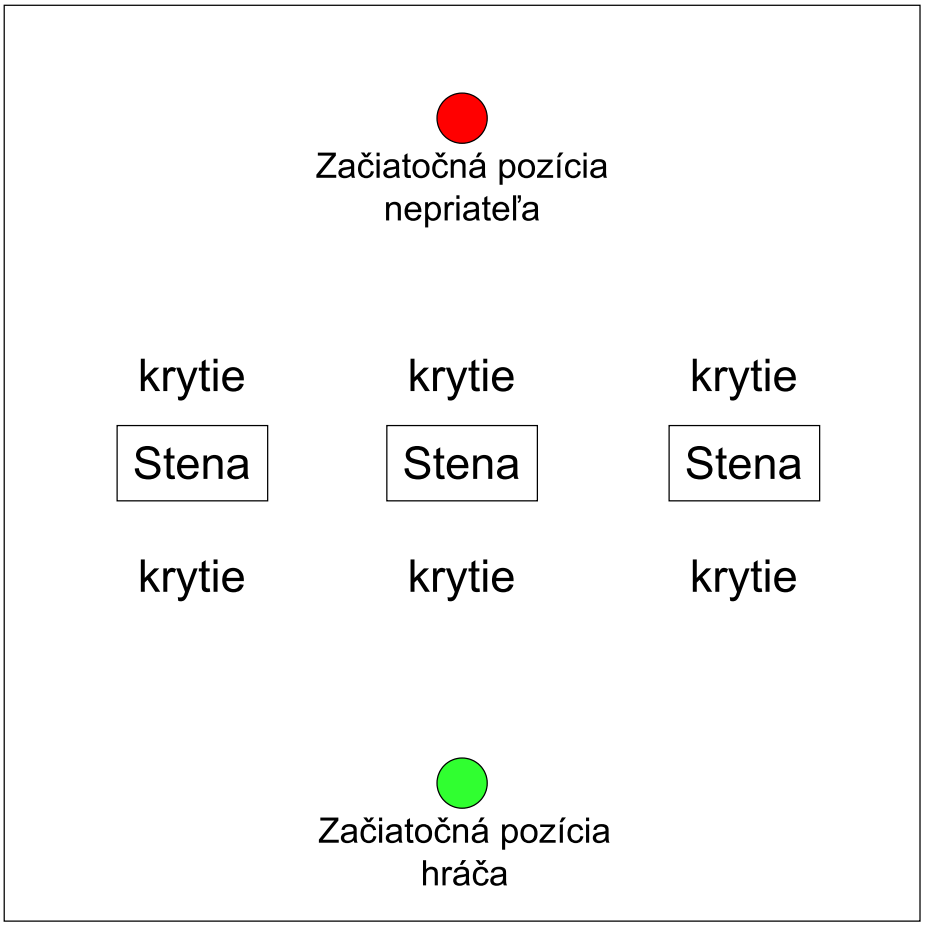
Simulácia určí textovými príkazmi akými krokmi hráč vyobrazí strach u umelej inteligencie. Na základe parametrov nastavených, pred spustením levelu, bude spozorovaný výsledok správania sa UI, keďže budú ovplyvňovať rast strachu, čo sa odrazí na správaní nepriateľov.

4.2 Level design

Scéna musí poskytovať cover zóny (miesta úkrytu) pre UI a hráča. Hráč aj nepriateľ má svoj vlastný spawn point[1]. Platforma je ohraničená, aby zabránila pádu mimo herný svet.

[1] Je to umiestnený herný objekt v scéne, ktorý určuje polohu vloženého objektu do scény

Ak by sa hráč náhodou prepadol, presunie sa na naspäť na svoju začiatočnú pozíciu hráča. 2D pohľad levelu je na obrázku 4.1.



Obr. 4.1: Obrázok zobrazuje štruktúru a rozostavenia základnej simulačnej scény. Miesta začiatočnej pozície a hráča určia ich základnú polohu a steny slúžia ako krytie z jednej alebo druhej strany. Krytie je herným objektom určujúci polohu, kde sa má UI dostaviť, ak hľadá úkryt.

4.3 Hráč

Herná postava bude ovládana s pohľadom z prvej osoby a takisto bude poskytovať ovládanie zbraní v podkapitole 4.3.1 a pohybu postavy v podkapitole 4.3.2.

4.3.1 Útok

Hráč bude môcť útočiť za pomoci zbraní na diaľku buď pomocou automatickej pušky alebo brokovnice, avšak výroba akejkoľvek zbrane by mala byť jednoduchá, pomocou editácie funkčnosti jedného skriptu. Zvýšenie strachu je možné za pomoci poškodenia života nepriateľa na určitú hranicu, akcie v prostredí ako zhasnutie svetiel, rýchle eliminovanie nepriateľov prinúti sa niektoré jednotky automaticky vzdať.

4.3.2 Mechaniky pohybu

Umožnené sú funkcie pohybu do každej strany s klávesmi WSAD. Hráč môže takisto aj vyskočiť do určitej výšky a aktivovať šprint prípadne extra rýchly presun za krátky čas, tzv. dash. Dash bude implementovaný iba vo vertikálnom smere.

4.4 Návrh UI

Ide o UI, ktorá bude schopná byť nepriateľom. UI je schopná patroly po vyznačenej trase. Následne po zahliadnutí nepriateľa môže spozornieť alebo automaticky zaútočiť. Podľa úrovne UI bude jej strach inak ovplyvnený. Slabšie jednotky budú mať strach už pri spozorovaní, niektoré až pri strate vysokého množstva bodov života. Ako náhle je UI v dosahu môže zaútočiť. Vizualizácia postavy s popisom jej častí na obrázku 4.2.

Obr. 4.2: Prvotný návrh nehrateľnej postavy len ako 3D herný objekt kapsula s textom pre stav chovania a dvoma ukazovateľmi. Zelený pre život a oranžový pre strach.

4.5 Pozorovanie strachu

Hráč bude oboznámený so sériou príkazov, ktorých vykonanie vytvorí požadovaný efekt. Strach bude možné pozorovať na základe jeho ukazovateľa nad každou postavou, ktorý sa pohybuje v rozmedzí 0-100. Najväčšie predloženie strachu bude preukazovať aktuálna reakcia postavy. Takisto budú zobrazované textové správy, ktoré budú popisovať aktuálny stav. Napríklad útek. Prvotný koncept úteku z boja ukázaný na obr. 4.3.

Obr. 4.3: Nepriateľ s ukazovateľmi aktuálneho stavu, života a strachu v stave úteku

4.6 Stavy UI

- Idle: V tomto základnom stave nie je obmedzovaná faktorom strachu a riadi sa len základnými funkciami ako je napríklad patrolovanie.
- Boj: V aktuálnom štádiu UI má možnosť, kedy nemá čo stratiť, ktorý predstavuje stav boj - zvýšenie adrenalínu a rozhodnutie pre súboj miesto úteku, dáva výhodu UI a zvyšuje jej parametre pre útok. Pri zvýšenom strachu bude mať nepriateľ síce rýchlejšie krvácanie a horšiu presnosť streľby, ale bude mať zlepšené priestorové videnie, rýchlejšie pohyby a útoky.
- Útek: V tomto stave UI beží na bezpečné miesto, pričom má zvýšenú štandardnú rýchlosť a snaží sa prežiť. Ak je pri UI strach naďalej zvyšovaný môže sa stav prepnúť a zmeniť až na odpadnutie.

4.7 Herné scény

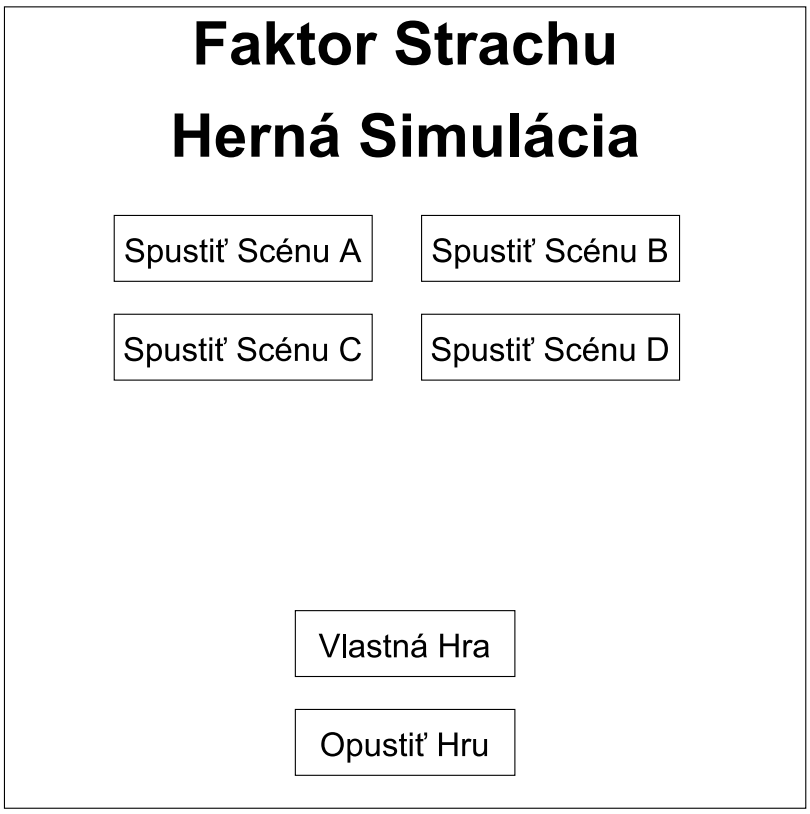
Štruktúra dema je tvorená scénami nasledovnými:

- Scéna Hlavné Menu
- Scéna Konfigurátor

- Scéna Prenasledovanie
- Scéna Konfrontácia so zbraňou
- Scéna Výťah
- Scéna Mŕtve telo
- Voľná Hra

4.7.1 Scéna hlavného menu

Je uvítacia scéna s prvkami užívateľského prostredia, ktorá je ako prvá po spustení aplikácie. Poskytuje výber scény, spustenie simulácie, ktoré sa presunie do ďalšej scény 4.7.2 a opustenie hry. Mockup[2] na obrázku 4.4.

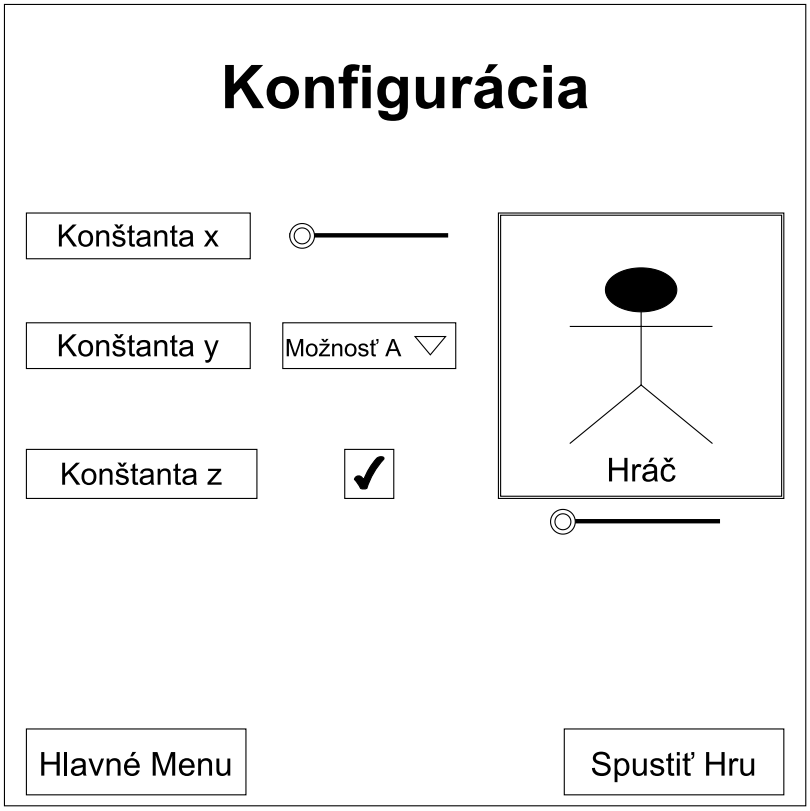


Obr. 4.4: 2D scéna s tlačidlami pre výber prechodov medzi scénami

4.7.2 Konfigurátor

Konfigurátor poskytuje nastavenie konštánt konfigurátora, ktoré ovplyvňujú tvorbu strachu u UI. Takisto poskytuje náhľad na charakter a prechod naspäť do hlavného menu alebo spustenie hry. Mockup na obrázku 4.5.

[2]Prvotný všeobecný náčrt



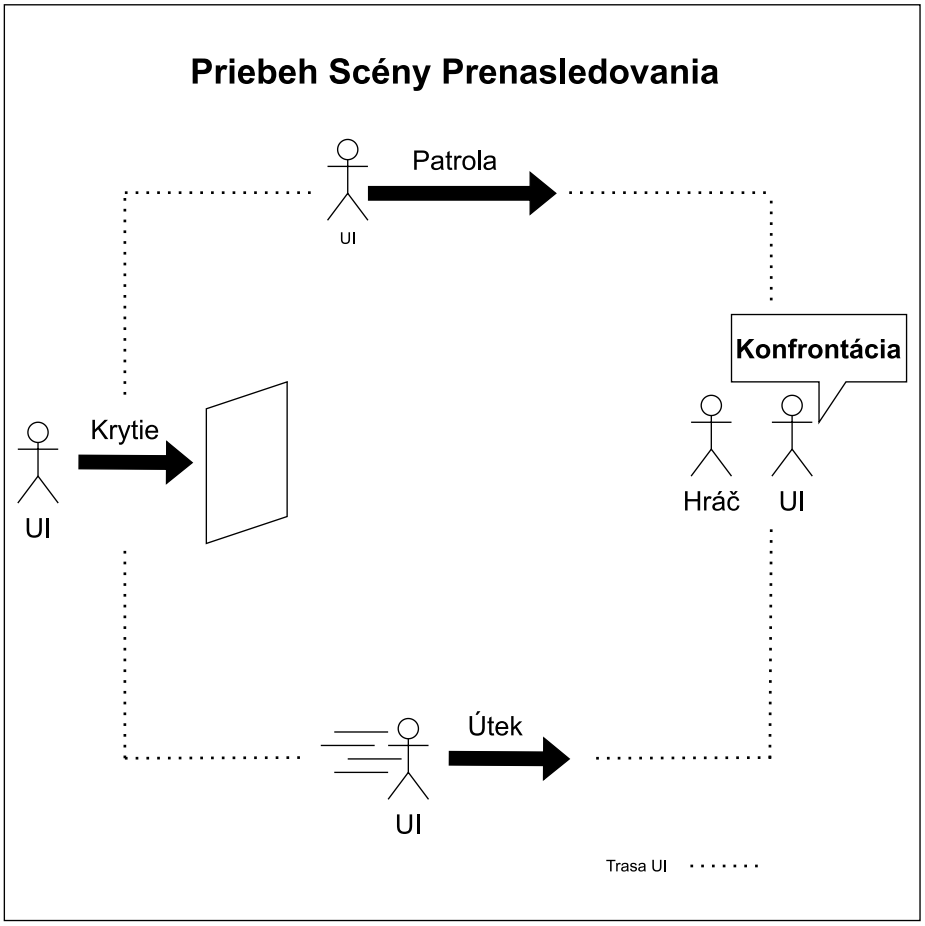
Obr. 4.5: 2D scéna s výberom float hodnôt pomocou polohovateľného ukazovateľa, výberu zo zoznamu pomoci dropdown menu alebo bool hodnôt pre toggle.

Každá nasledujúca scéna bude mať vypísanú úlohu, ktorú musí hráč splniť pre splnenie plánovaného chovania. Úloha bude kedykoľvek dostupná v hre kliknutím na klávesu **I**[3] a zastaví hru, kým si hráč prečíta, ktoré úlohy má splniť.

4.7.3 Prenasledovanie

Chovanie by malo v scéne by malo vyobrazovať obyčajného človeka bez zbrane. Umelá inteligencia bude pôsobiť vo svojej rutine prechádzania sa po prednastavenej trase, kde hráč túto rutinu naruší a agent ho konfrontuje, ak to nezaberie uteká a následne sa snaží pred hráčom skryť. Mockup levelu na obrázku 4.6.

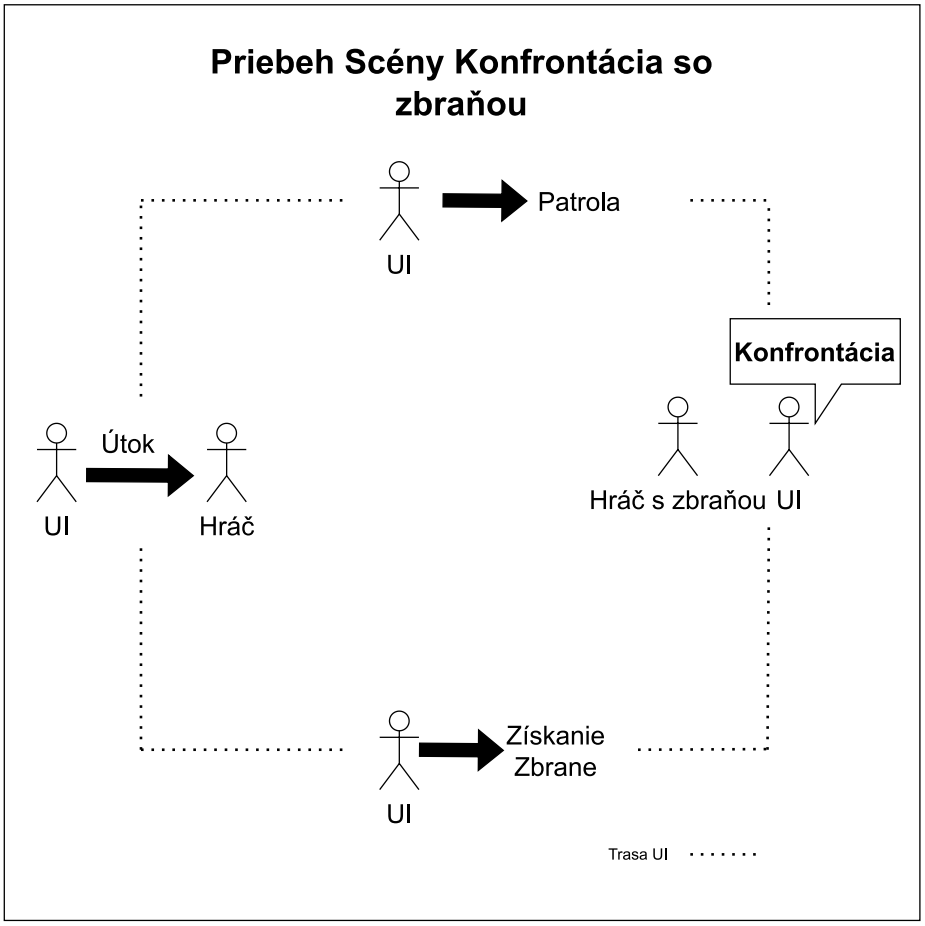
[3] I - klávesa zaužívaná ako info klávesa v herných žánroch



Obr. 4.6: Cielený priebeh levelu pre simululáciu Prenasledovania.

4.7.4 Konfrontácia so zbraňou

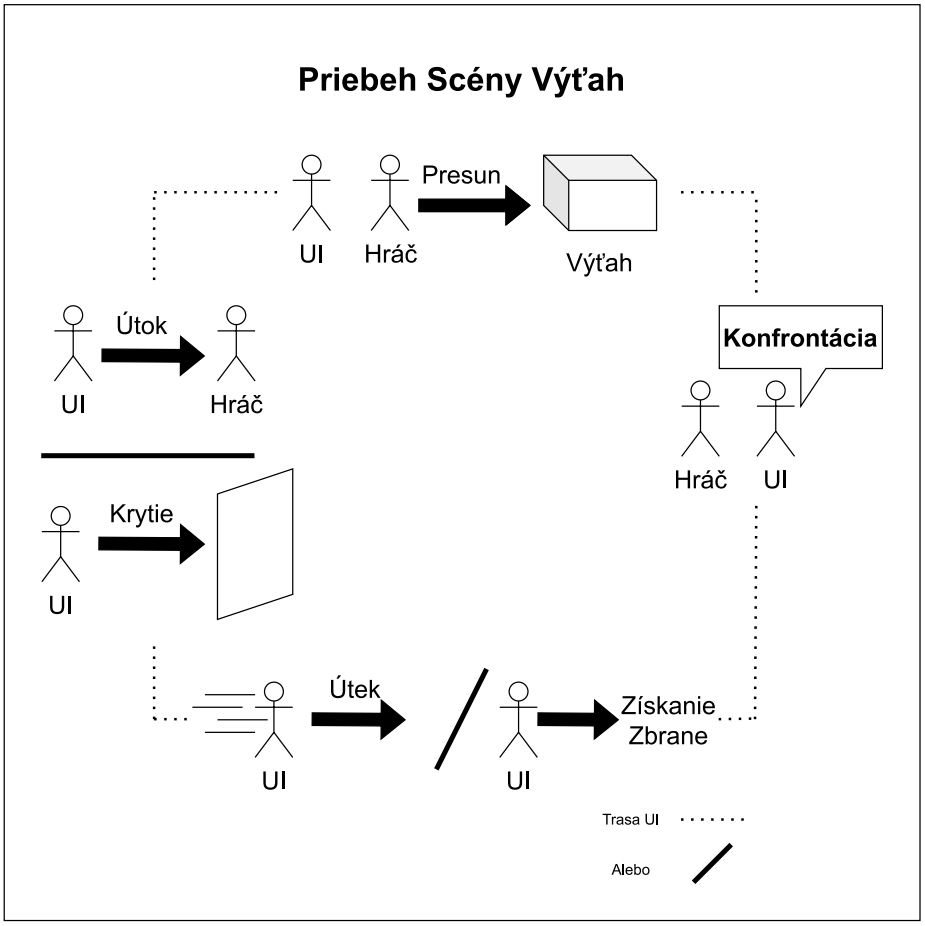
Umelá inteligencia je na svojej patrole, ak ju hráč konfrontuje so zbraňou beží pre svoju zbraň a útočí na hráča, v inom prípade pokračuje v svojej rutine. Priebeh scény na obrázku 4.7



Obr. 4.7: Priebeh scény s postupnými krokmi UI a hráča.

4.7.5 Výťah

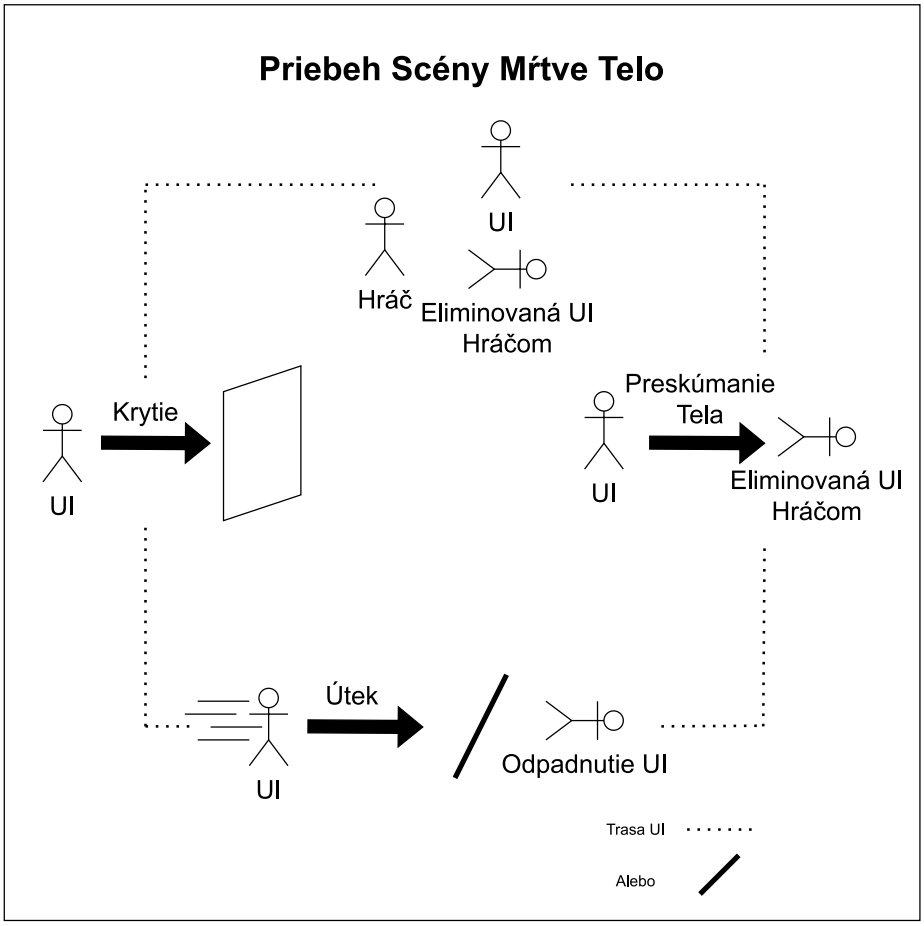
Priebeh scény Výťah, kde UI využíva priestor výťahu na prevoz, kým nie je narušený jej priestor hráčom a rozhodne sa situáciu riešiť na základe strachu buď utiecť alebo zaútočiť. Mockup levelu na obrázku 4.8.



Obr. 4.8: Mockup vyzobrazuje ako sa Agent presunie spoločne aj s hráčom do výťahu, ak hráč naruší jeho priestor vzniká konfrontácia. Agent buď utečie alebo nájde zbraň a ňou následne útočí.

4.7.6 Mŕtve Telo

Skupina agentov bude konfrontovaná hráčom, kde hráč musí zneškodniť jedného z nich. Agenti v okolí reagujú na túto akciu zvýšenou koncentráciou strachu, ktorá vo väčšine prípadov skončí až odpadnutím. Takisto agenti, ktorí nájdu telo, budú v stave, kde sa bude strach zvyšovať rýchlejšie ako v kľudnom stave. Mockup levelu na obrázku 4.9.



Obr. 4.9: Ciélený priebeh levelu pre simululáciu Prenasledovania.

4.7.7 Voľná hra

Bude ponúkať predefinovaný level, v ktorom je možné, aby hráč vyskúšal boja schopnosť UI s jej základným nastavením a implementovaných univerzálnym stromom chovania 5.5 s doplnením o sekvenciu liečby v úkryte.

Kapitola 5

Implementácia

Demo so scénami bolo vytvorené v hernom engine Unity - verzia 2021.3.0f1. Implementácia skriptov prebiehala v jazyku C# vo vývojárskom prostredí Visual Studio 2019. Inšpiráciou pre implementáciu bola kniha pre pokročilé programovanie UI v unity [9] a takisto videá komunitných tvorcov a to konkrétne Brackeys, TheKiwiCoder, CodeMonkey, GameDevChef ktorí vysvetľujú princípy a uvádzajú demonštráciu skriptov.

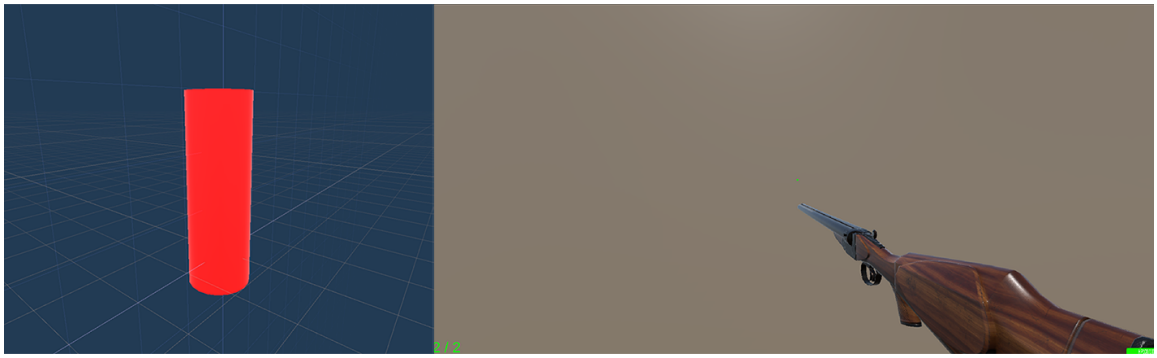
5.1 Herné objekty

Všetky herné objekty, ktoré sú v scénach sú vytvorené ako prefabs. Takže vkladanie objektov do scén sa prevádza pomocou funkcie **Instantiate()**[1]. Táto funkcia vytvorí kópiu daného prefabu a vloží ho do scény.

FirstPersonPlayer / hráč s pohľadom z prvej osoby

Hráč je tvorený základným 3D objektom cylinder, ktorý ma na sebe pripevnený collider, kvôli zaznamenaniu prekrytia vrstiev. Pre pohyb má takisto komponentu Character Controller a jeho pohyb je riadený skriptom **PlayerMovement** zatiaľ čo jeho body života v skripte **PlayerLife**. Hráč má v sebe vloženú kameru a tá je ovládaná pohybom počítačovej myše. Takisto má na sebe pripevnený herný objekt **GroundCheck**, ide o kontrolu či sa hráč dotýka zeme. Posledný objekt, ktorý hráč má je prefab zbrane s pripevneným časticovým efektom (particle effect) pre grafické znázornenie paľby. Pohyb kamery je zaistený skriptom **MouseLook** priradeným ku kamere. Pohyb hráča je možný klávesmi W,S,A,D a zároveň je ovládanie prispôsobené tak, aby fungovalo aj na konzolových ovládačoch cez funkcie **Input.GetAxis**. Skákanie pomocou klávese Space (medzerník), šprint klávesou Shift a pohyb dash pomocou LeftCtrl. Prefab hráča na obrázku 5.1.

[1] https://docs.unity3d.com/ScriptReference/Object.Instantiate.html



Obr. 5.1: 3D model hráča v prostredii Unity a pohľad z prvej osoby v hernom prostredii

WeaponPickup / Zdvihnutie Zbrane

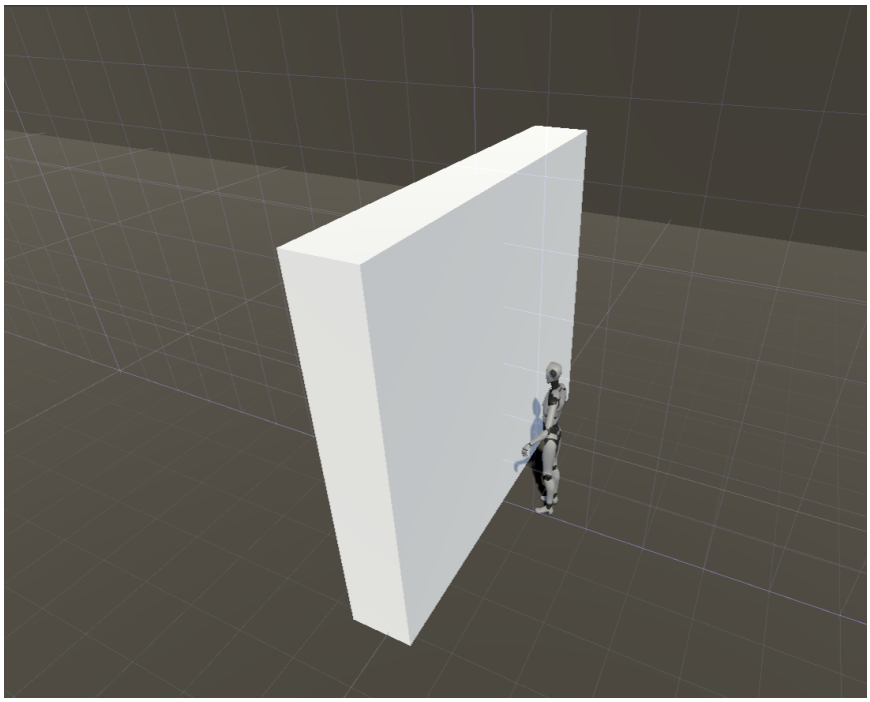
WeaponPickup je herný objekt využitý v scéne Mobil. Skladá sa z herných objektov guľa a zbrane. Collider komponent sa využije na rozpoznanie, kedy UI vbehla do priestoru objektu, pomocou funkcie **OnTriggerEnter(Collider other)** a môže sa aktivovať skript, ktorý pripevní zbraň na prednastavenú pozíciu. Skript takisto musí zaistiť kontrolu, že je zbraň už zdvihnutá a takisto nastaviť jej rotáciu a pozíciu. Následne sa musí ešte predať komponenta skriptu zbrane UI agentovi, aby ju mal možnosť ovládať pomocou skriptu **RayCastWeapon**. Na obrázku 5.2 vidno prefab tohoto objektu.

Obr. 5.2: 3D model Weapon pickup v Unity, kde gula je collider a vnútri je zbraň, ktorá sa nasadí na UI

Cover / Úkryt

Cover je herný objekt využitý pre skrytie sa UI agenta. Skladá sa z herných objektov cube (kocka) a herných objektov coverspot (miesto úkrytu), ktoré určujú polohu, do ktorej sa agent dostaví pri krytí. Zoznam miest úkrytu sa vždy pri vyhodnocovaní stromu chovania

aktualizuje a určí sa poloha najbližšieho pre využitie najrýchlejšieho dostupného úkrytu. Úkryt je znázornený na obrázku 5.3.



Obr. 5.3: Krycia stena a NPC na pozícii herného objektu s polohou krytia

Gun / zbraň

Objekt zbraň je ovládaný skriptom **Gun**, ktorý je určený pre ovládanie zbrane. Zamierenie na cieľ funguje pomocou tzv. RayCasts, v preklade lúče, ktoré určia vzdialenosť a aj aký cieľ zasiahneme. Následne je možné vystreliť paprsok, ktorý prakticky instantne zasiahne cieľ. Modifikáciou tohoto skriptu bolo možné vytvoriť aj streľbu pre UI, s tým, že implementácia zahrnula streľbu projektilu miesto lúčov. Skriptom je univerzálne nastaviteľný pre rôzne typy zbraní. Obrázok 5.4 jednej zo zbraní používajúci skript **Gun**.

Obr. 5.4: Dvojhlavňová Brokovnica poskytnutá z balíčku Double-barrel gun[2]

5.2 Zostavenie nepriateľa

Nepriateľ bol na začiatok zahrnutý len ako capsule collider a hlavná forma jeho správania bola spracovaná bez štruktúry. Toto rozhodnutie viedlo ku skorej neprehľadnosti kódu a bolo potrebné prejsť na strom chovania. Bola možná implementácia aj konečného automatu, ale jednoduchá modifikácia stromu chovania je uprednostnená pri veľkom počte možných stavov. UI agent má riadiace skripty na mnoho úloh a to konkrétne:

- FearFactorAI - Riadi a ovplyvňuje hodnotu strachu v rozmedzí 0-100
- AiHealth - Riadi a ovplyvňuje hodnotu života v rozmedzí 0-100
- AiTreeConstructor - Vytvára stromy chovania a vyhodnocuje ich

FearFactorAI

Hlavným faktorom pri rozhodovaní čo sa udeje je stav premennej FearState, ktorej hodnota je pozmenená z jednotlivých uzlov stromu chovania. Jednotlivý stav ovplyvňuje rýchlosť tvorby strachu alebo hlášky, ktoré UI použije v danej situácii.

- CALM - Stav kľudu. Všetky premenné sú na základnom nastavení.
- OBSERVING - Objasnenie situácie. Tvorba strachu je ovplyvnená časom stráveným skúmaním situácie s hráčom.
- RUNNING - Útek. Tvorba strachu zvýšená na základe dĺžky času, ktorú UI uteká.
- UNCONSCIOUS - Bezvedomie. Deaktivácia všetkých komponent agenta a zapnutie kinematic vlastnosti na rigidbody.

[2] https://assetstore.unity.com/packages/3d/props/guns/double-barrel-gun-163068#description

Strach sa nadobúda postupne za čas určený v konštante **timeBetweenFearAddition** a rekurzívne sa táto funkcia volá, za určitý čas nastavený pred začatím scény, pomocou vstavanej Unity funkcie **Invoke()**[3].

AiHealth

Spravuje funkcie **TakeDamage(float amount)**, ktorá odoberá prijatú čiastku od bodov života UI. Ak sú životy pod kritickou hranicou, volá sa funkcia **Die()**, ktorá musí deaktivovať niektoré zo skriptov a grafický interface , zastaví agentov pohyb, aktivuje atribút kinematic a pozastaví animátor, pomocou funkcie skriptu **Ragdoll**, pre prirodzenejšie padnutie postavy po smrti alebo upadnutia do bezvedomia. Takisto ponúka funkciu **Restore(float amount)** pre obnovenie zdravia UI agenta, keď sa dostane do úkrytu.

AiTreeConstructor

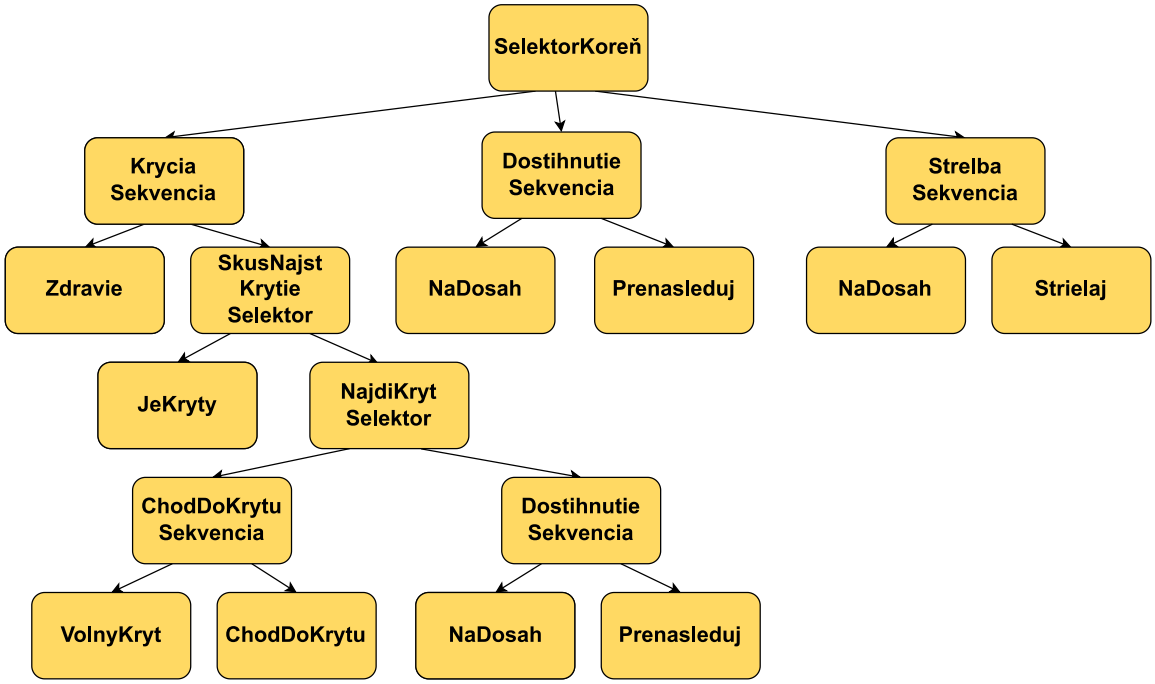
Musí zhromaždiť všetky dôležité komponenty, ktoré vyžadujú konštruktory uzlov a všetky sa zhromaždia teda vo funkciách Unity **Awake()**[4] a **Start()**[5]. Typ stromu chovania je možné nastaviť pred začatím, ale nie počas behu programu. **ConstructBehaviorTreeByNumber(int numberOfTree)** určí požadovaný strom chovania a následne je skonštruovaný. Strom sa vyhodnocuje v funkcií **Update()**[6] a za pomoci vyhodnotenia funkciou **Evaluate()** koreňového uzlu. Každý strom je vytvorený bez vizualizéru, a preto je z editoru možná iba ukážka kódu 5.1. Vizualizácia kódu je vytvorená softwarom tretej strany a vyzerá nasledovne 5.5.

```
Node korenovyUzol; // vytvori korenovy uzol typu Node
VytvorStrom(string menoStromu, Node korenovyUzol) //

//vyhodnot stav stromu, ak agent nemoze realizovat ziadnu ulohu, zastav ho
while(SimulaciaBezi){
    korenovyUzol.VyhodnotStav();
        if ( korenovyUzol.stav == neuspech ) //
           ZastavAgenta();
}
```

Výpis 5.1: Ukážka kódu ConstructBehaviorTreeUniversal v jazyku C♯

[3] https://docs.unity3d.com/ScriptReference/MonoBehaviour.Invoke.html
[4] Funkcia sa volá pred spustením hry
[5] Funkcia sa volá pred prvým snímkom po zapnutí skriptu hrou
[6] Funkcia sa volá každý snímok



Obr. 5.5: Vizualizácia základného stromu chovania a jeho zostrojenie podľa pseudokódu 5.1, ktorý je zoskladaný na základe univerzálnych uzlov chovania v kapitole 5.4. Ako prvá sa kontroluje krycia sekvencia, keďže život agenta má najvyššiu prioritu. Ak je zdravie pod určitou úrovňou, hľadá úkryt. Ak je kryty vracia úspech, v opačnom prípade hľadá kryt. Ak nie je dostupný kryt, prenasleduje nepriateľa. Bez zranenia prebieha sekvencia dostihnutie. Ak je hráč na dosah, agent ho prenasleduje. Ak je na dosah streľby, môže na hráča strieľať.

Nepriateľ ako herný objekt

Postava je prebratá z Starter Assets - Third Person Character Controller[7] (ovládanie charakteru s pohľadom z 3.osoby) Asset Storu Unity[8]. Má základné animácie a štruktúrovaný skeleton skladajúci sa z telesných časti, ktoré sú využité pri animáciách, určovaní a tvorení jednotlivých hitboxov (časti teľa, ktoré zaznamenávajú zásahy). Takisto pomáha vizualizovať stav odpadnutia pri aktivácii atribútu kinematic všetkých jej **rigidbody** komponent, čo slúži takisto ako skvelá vizualizácia pri zneškodnení. Nepriateľ má takisto svoj vlastný herný objekt **Canvas**(2D plátno) na ktorom sú vložené všetky grafické prvky. Text zobrazuje stav nehrateľného charakteru, čiže uzol ktorý sa aktuálne vyhodnocuje alebo zasiela success, v ktorom sa nachádza. Následne obsahuje objekt **healthBar**, ktorej ukazovateľ je riadený skriptom **AiHealth**. Objekt healthbar obsahuje atribút slider, ktorý má hodnoty od 0 po 1, čiže prevedením na tieto hodnoty v skripte vieme zobraziť aktuálne body života graficky. Tým istým spôsobom sa rieši aj ukazovateľ strachu. Herná postava je na obrázku 5.6.

[7] https://assetstore.unity.com/packages/essentials/starter-assets-third-person-character-controller-196526

[8] Digitálny obchod Unity poskytujúci vytvorené herné assety https://assetstore.unity.com

Obr. 5.6: Ukážka postavy UI s herným objektom canvas, na ktorom sú prvky Heatlhbar, FearBar a text pre stav v strome chovania

5.3 Strom chovania

Skladá sa zo základného skriptu **Node**, ktorý definuje všetky následne vytvorené uzly a vyhodnocuje nasledujúce kroky s funkciou **NodeState Evaluate()**. Uzly majú stav, ktorý je iba čitateľný a možné stavy sú:

- RUNNING - Stav beží: vyhodnotenie stavu nie je možné alebo nechcené (running)
- SUCCESS - Stav je úspešný: uzol bol vyhodnotený ako úspech (success)
- FAILURE - Stav je neúspešný: uzol bol vyhodnotený ako neúspech (failure)

Základné typy uzlov pre konštrukciu stromu chovania sú nasledovné:

- Skript **Node** zabezpečuje uzly akcie a uzly podmienok pre vyhodnocovanie stavov v hre a pre následnú realizáciu zadaných úloh.
- Skript **Selector** funguje ako logické **OR** a tým pádom stačí, ak aspoň jeden z uzlov zo zoznamu vráti success.
- Skript **Sequence** funguje ako logické **AND** a to vráti success, len, keď všetky z uzlov zoznamu vrátia success.
- Skript **Inverter** mení success na failure a vice versa. Stav running sa nemení.

5.4 Univerzálne uzly chovania

Jedná sa uzly, ktoré sú použité vo viacerých scénach a dokazujú modulárnosť a znovupoužiteľnosť uzlov stromu chovania z kapitoly 3.1.

Chase seqeunce / sekvencia prenasledovania

Skladá sa z uzlu podmienky **JeVDostihnutelnejVzdialenosti** - Je pozícia medzi hráčom a nepriateľom v dostatočná ? A uzlu akcie **DostihniHráča** - Naháňaj hráča pokiaľ je vzdialenosť dostačujúca. Ak je hráč v dostatočnej vzdialenosti, chasingRangeNode odosiela success na čo následne je chaseNode v stave runnig až pokým, agent nedostihne hráča do dostatočnej vzdialenosti a následne vyšle succes, že hráča dostihol.

JeVDostihnutelnejVzdialenosti je uzol typu vzdialenosti, ktorý vyžaduje parametre prah vzdialenosti, polohu objektu A a polohu objektu B, medzi ktorými je vypočítaná vzdialenosť v hernom svete a ak je menšia ako prah vracia success. Pseudokód podmienky 5.2.

```
vzdialenost = Vector.Vzdialenost(pozicia.hrac, pozicia.nepriatel);
return vzdialenost <= prah vzdialenosti ? Uspech : Zlyhanie;
```

Výpis 5.2: Pseudokód uzlu podmienky pre vzdialenosť

Dostihnutie vyžaduje polohu cieľa a NavMeshAgent komponentu umelej inteligencie, pre nastavenie destinácie agenta na polohu cieľa. Náhľad pseudokódu 5.3.

```
if(vzdialenost >= hranica)
{
    agent.Dobehni(hracova poloha)
    return bezi
}
return uspech
```

Výpis 5.3: Pseudokód uzlu Dostihnutia

Shooting sequence / sekvencia streľba

Agent musí získať zbraň, aby mohla sekvencia začať. Následne po dosiahnutí dostatočnej vzdialenosti UI agent môže začať útočiť pokým je hráč v dostatočnej vzdialenosti.

Cover sequence / sekvencia úkrytu

Ak je zdravie agenta pod zvoleným prahom, vyhľadáva krytie. Selektor typu ZiskajKrytie vyberá z dvoch možností a to nasledovných: UI je už v zákryte alebo musí kryt nájsť. Ak neexistuje žiaden úkryt agent aplikuje pseudokód 5.3 uzlu dostihnutie. Avšak, ak úkryt existuje, agent zistí možnosť voľných úkrytov a presunie sa do najbližšieho voľného úkrytu.

Heal seqeunce / sekvencia Liečby

Ak je agent v úkryte, následne môže začať regenerácia jeho zdravia.

FindWeapon sequence / sekvencia nájdi zbraň

UI zistí dostupnosť zbraní v hernej scéne. Pri zbrani dostupnej, UI sa dostaví na pozíciu zbrane a získava zbraň, ktorú následne môže použiť.

5.5 Herné scény

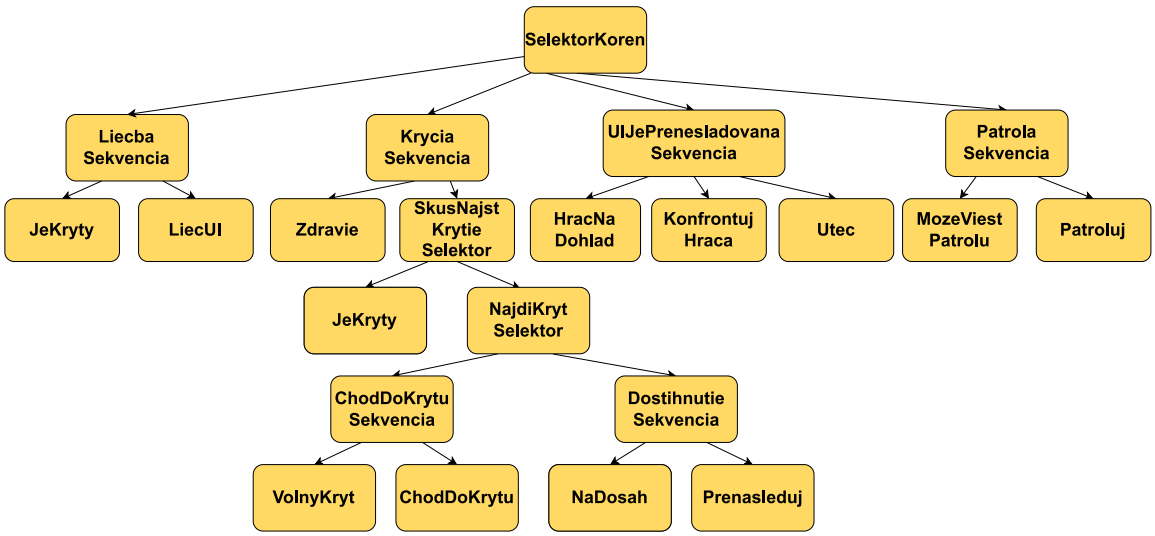
Simulácia obsahuje 4 scény, ktoré zobrazujú rôzne stromy chovania a ich zoznam je nasledovný :

- Prenasledovanie
- Výťah
- Mrtvé Telo
- Konfrontácia nepriateľa

5.5.1 Prenasledovanie

UI agent sa pohybuje na základe skriptu **NavigationPathForAI** (NavigačnáTrasaUI), ktorý iteruje cez zoznam destinácii. UI pomocou senzorov zachytí hráča a následne kontroluje situáciu a snaží sa od hráča vzdialiť. Ak hráč naďalej zasahuje do priestoru UI agenta, agent sa rozhodne utiecť, ak sa mu to nedarí, vyhľadá najbližšie krytie a schová sa dovtedy, kým nie je úroveň jeho strachu opäť na nulovej hodnote. Vizualizácia na obrázku 5.7. V scéne sú implementované nasledujúce skripty uzlov:

- MozeViestPatrolu - Overuje prítomnosť hráča, vysiela success, ak je hráč mimo senzor
- Patroluj - UI sa pohybuje po svojej vyhradenej trase
- HracNaDohlad - Potvrdzuje prítomnosť hráča v priestore UI za pomoci senzorov
- KonfrontujHraca - Sústreďuje sa pozornosť na hráča a slovne ho konfrontuje o tom, aby ho prestal prenasledovať
- Utec - UI utečie od hráča



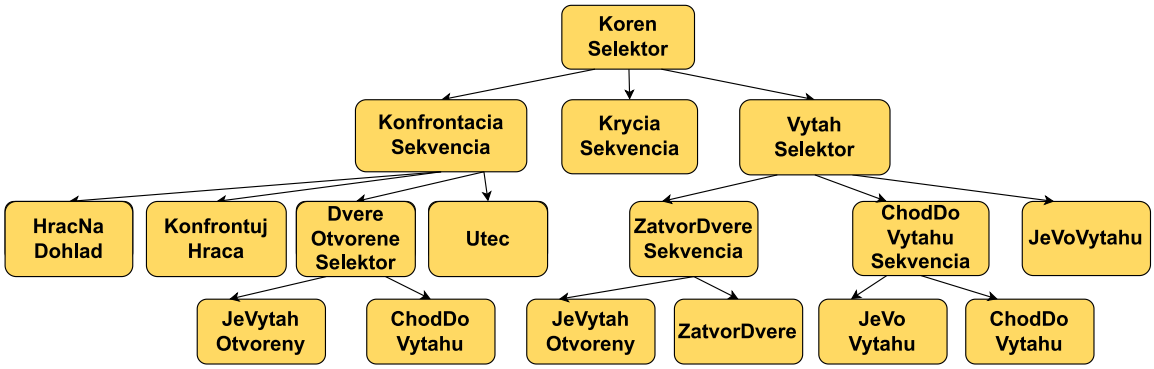
Obr. 5.7: Vizualizácia chovania UI pri prenasledovaní. UI vedie patrolu, pokiaľ nie je v jej okolí nepriateľ. Ak je, hráč je konfrontovaný a ak hráč neodíde, UI utečie. Ak je na UI agenta zaútočené kryje sa a lieči sa ak je sám v úkryte.

5.5.2 Výťah

UI nastúpi do výťahu a pri konfrontácii s hráčom sa rozhodne pre útek. Záleží pri tom na: vzdialenosti hráča od agenta alebo od útoku hráča na agenta. Pri postupnom približovaní bude UI agent nútený utiecť. Pri napadnutí opätuje útok. Vizualizácia na obrázku 5.8. V scéne sú implementované nasledujúce skripty uzlov:

- JeVoVytahu - Vracia success ak je agent v priestore výťahu
- JeVytahOtvoreny - Vracia success pri otvorených dverách výťahu
- ChodDoVytahu - Nastaví cieľovú destináciu agenta do výťahu, vracia success ako náhle sa do neho dostane
- ZatvorDvere - Zatvoriť dvere je možné ak sa dvere nehýbu a dvere sú otvorené, pri splnení podmienky vracia success.
- OtvorDvere - Otvoriť dvere je možné ak sa dvere nehýbu a dvere sú zatvorené, pri splnení podmienky vracia success.

K tomuto scenáru je takisto pripojený skript **ElevatorCheck**, ktorý implementuje chovanie pre otvorenie, zatvorenie dverí a poskytovanie informácii o stave pozícii dverí.



Obr. 5.8: Vizualizácia chovania UI scény výťah. UI prechádza má rutinu, kde vchádza do výťahu, zavrie dvere a po zavretí ich po určitom čase otvorí. Ak je hráčom konfrontovaná rozhodne sa pre útek a pri zranení sa presúva do krytia za pomoci už použitej v univerzálnej krycej sekvencie.

5.5.3 Mrtvé Telo

V scéne hráč zneškodní UI jedného zo skupiny agentov. Je pravdepodobnosť, že agenti, ktorí spozorujú mŕtve telo odpadnú do bezvedomia. Ak sa ostatní agenti rozhodnú zaútočiť je tu možnosť, že sa agenti automaticky vzdajú ak do určitého času hráč zneškodní zvolený počet nepriateľov. Strom chovania je vyobrazený na obrázku 5.9. V scéne sú implementované nasledujúce skripty uzlov:

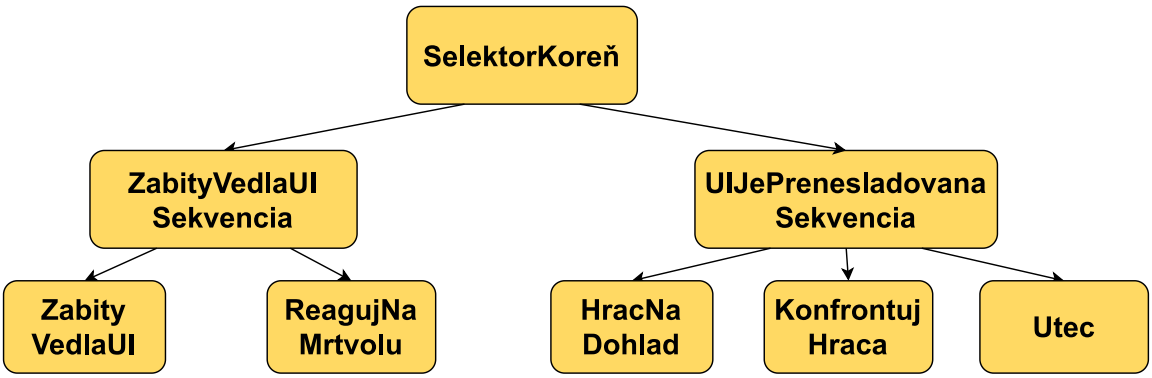
- SúVBlízkostiMrtviSpojenci - Pomocou senzoru a **Rigidbody** komponent AI agentov zistí, prítomnosť mŕtvych spojencov za pomoci overenia ich upraveného atribútu tag (značka), ktorý je nastavený na **MrtvyNepriatel**

- BolaMrtvolaKontrolovana - Prevencia pred repetetívnou kontrolou tej istej mŕtvoly
- SkontrolujMŕtvolu - Presunie svoju pozornosť na mŕtvolu a reaguje na základe svojich parametrov
- ZabitýVedľaUI - Ak je UI zneškodnená vedľa druhej, tak živá skontroluje komponentu tela na zemi a posiela success, ak je nájdená
- ReagujNaMrtvolu - agent sa pozrie na mŕtvolu a jeho strach sa zvýši na toľko, že odpadne a tým pádom sú jeho komponenty deaktivované takisto ako pri smrti UI

Scéna poskytuje dve možnosti, ako zneškodniť nepriateľov:

1. Hráč zneškodní stanovený počet nepriateľov za časový limit a v tom prípade sa nepriateľa podľa strachu rozpŕchnu, utečú, odpadnú.
2. Hráč eliminuje nepriateľa v blízkosti jej spojenca, kde spojenec reaguje odpadnutím.

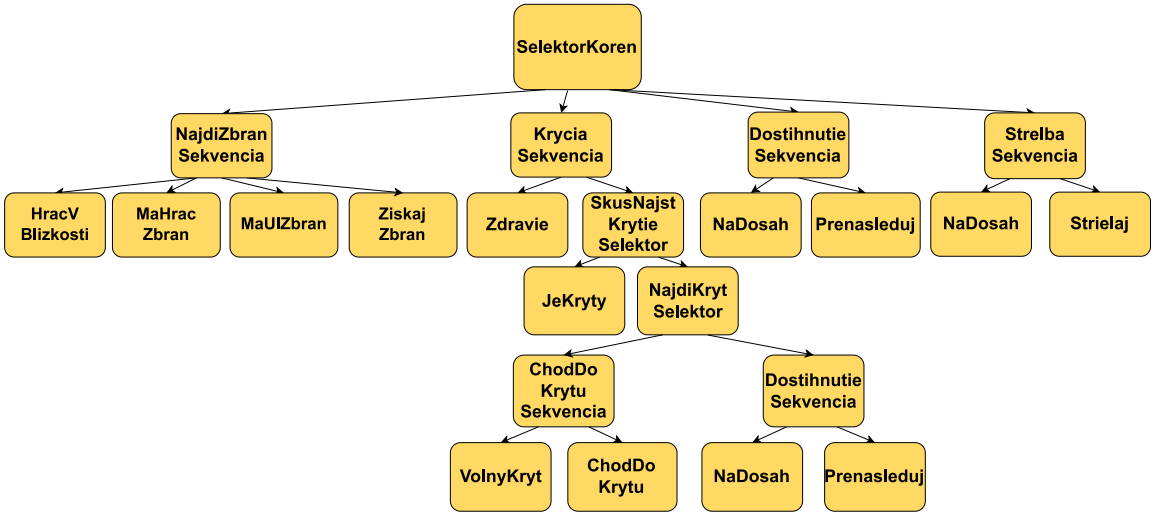
Správanie bude riadené skriptom **KillsInTimeManager.cs**, ktorý určí či hráč stihol v zadanom čase eliminovať dostatok nepriateľov. Uzle špecifické pre túto scénu:



Obr. 5.9: Vizualizácia chovania UI pri zneškodnení spojencov UI

5.5.4 Konfrontácia nepriateľa

Umelá inteligencia je na svojej patrole, ak ju hráč konfrontuje so zbraňou beží pre svoju zbraň a útočí na hráča, v inom prípade pokračuje v svojej rutine. Priebeh scény na obrázku 5.10.



Obr. 5.10: Vizualizácia chovania UI pri strete hráča so zbraňou.

5.6 Užívateľské prostredie

Užívateľské prostredie simulácie poskytuje výber medzi scénami v hlavnom menu a nastavenie základných faktorov strachu a ich hodnôt v konfiguračnom okne pred začatím scény. Demo je možné zastaviť pauzovacím menu a následne sa do nej vrátiť.

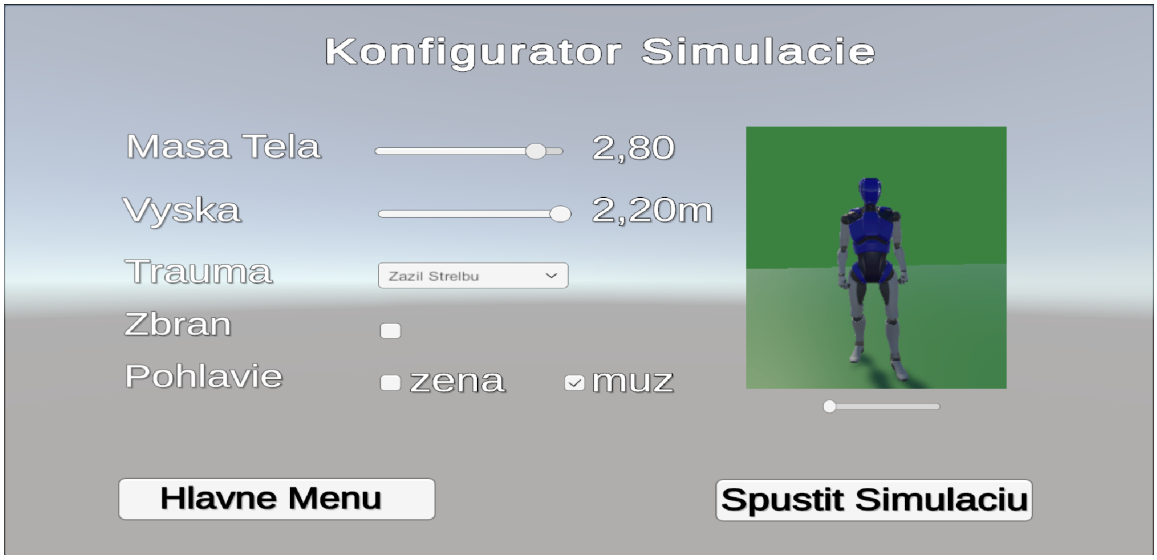
5.6.1 Main menu / hlavné menu

Hlavné menu sa skladá z modifikovateľných tlačidiel, ktoré spúšťajú vybrané scény alebo opustenie aplikácie. Náhľad menu je na nasledujúcom obrázku 5.11.

Obr. 5.11: Ukážka hlavného menu simulácie

5.6.2 Konfigurátor

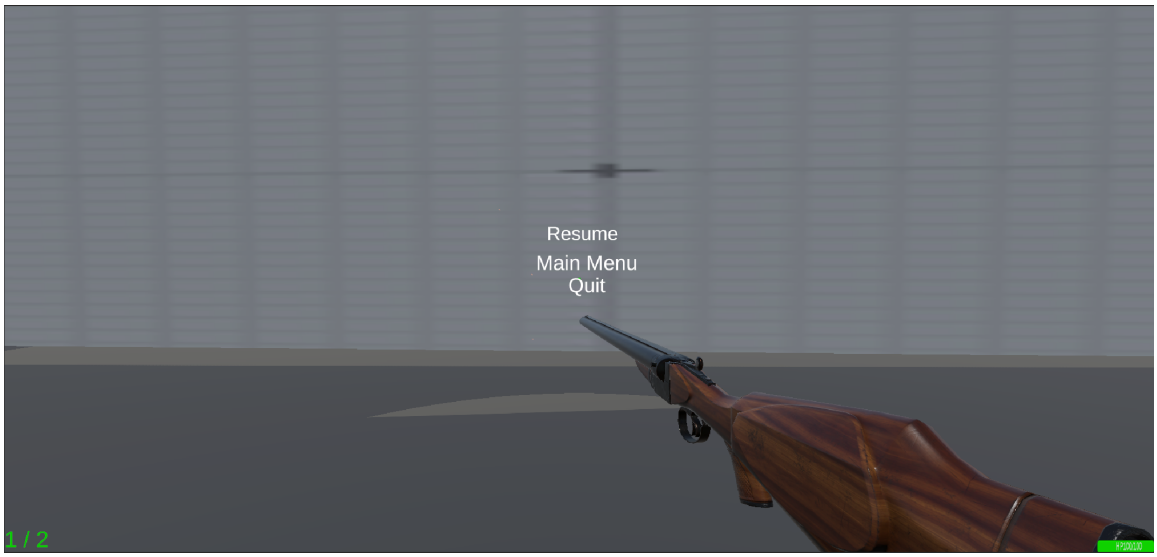
Konfiguračné okno ovplyvňuje hodnoty v skripte **AiConstraintsConfig**, ktorý pred spustením simulácie upraví konštanty, ktoré ovplyvnia správanie umelej inteligencie v každej zo scén. Náhľad Konfiguračného okna na obrázku 5.12.



Obr. 5.12: Ukážka konfigurátora, ktorý poskytuje výber výzoru postavy UI pre váhu a výšku. Takisto umožňuje nastavenie traumy a pohlavia, ktoré ovplyvní správanie v niektorých scénach a výber či bude mať UI agent zbraň.

5.6.3 Pause menu / pauzovacie menu

Menu pauzy poskytuje návrat do hry tlačidlom - Resume, návrat do Hlavného menu tlačidlom - Main Menu a návrat na plochu opustením aplikácie tlačidlom - Quit. Menu sa zavolá klávesom **P** alebo klávesom **Esc**. Náhľad na menu z hry na obrázku 5.13.



Obr. 5.13: Záber z hry pri aktivovaní Pauzovacieho menu

Kapitola 6

Uživateľská štúdia

Cieľom štúdie bolo zistiť na základe objektívnych názorov, ktoré prvky sú v bakalárskej práci využité správne a čo naopak treba zdokonaliť alebo v budúcnosti rozšíriť. Štúdia bola vykonaná na 29 ľuďoch, ktorí zodpovedali, ktoré prvky zo zoznamu sú podľa nich najlepším ukazovateľom emócii. Konkrétne pre emóciu strach v kapitole 6.1. Následne v kapitole 6.2 situácie vyvolávajúce strach vyberali z piatich možnosti reakcie na danú situáciu. Opýtaní museli vyjadriť, v kapitole 6.3 hodnotenia scenárov , čo na scéne bolo vhodné a čo naopak pridať alebo upraviť. Náhľad dotazníku v prílohe B na obrázku B.1.

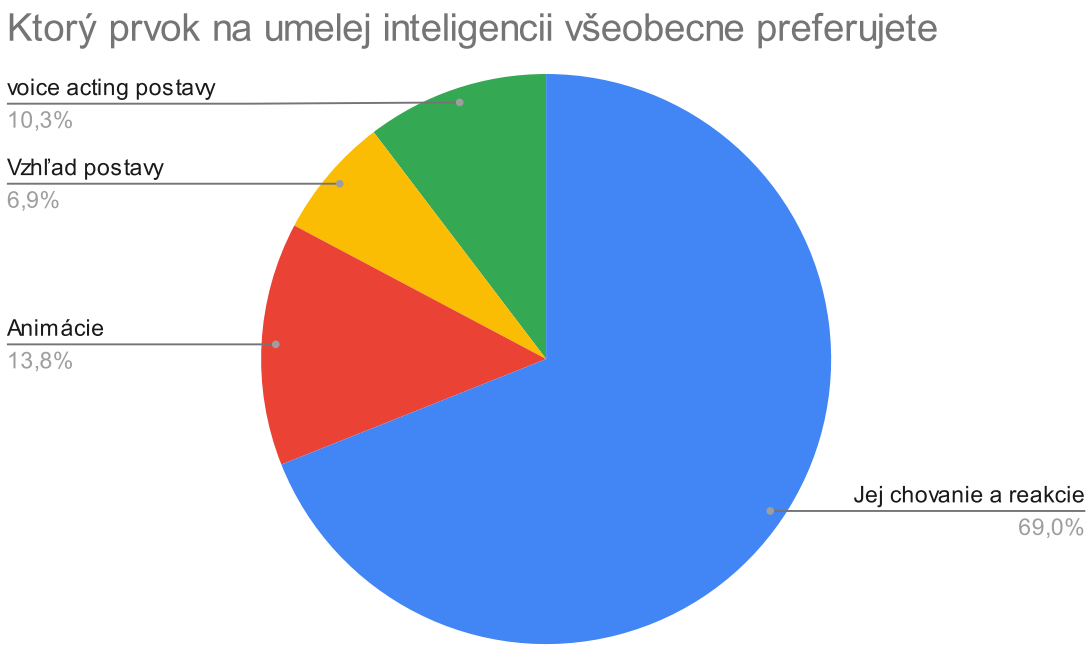
6.1 Preferencie zobrazenia emócii UI

Na základe zodpovedaných dát, viď. graf 6.1, na otázku, ktorý prvok na umelej inteligencii preferujete všeobecne, mohli užívatelia zodpovedať podľa nasledovného zoznamu:

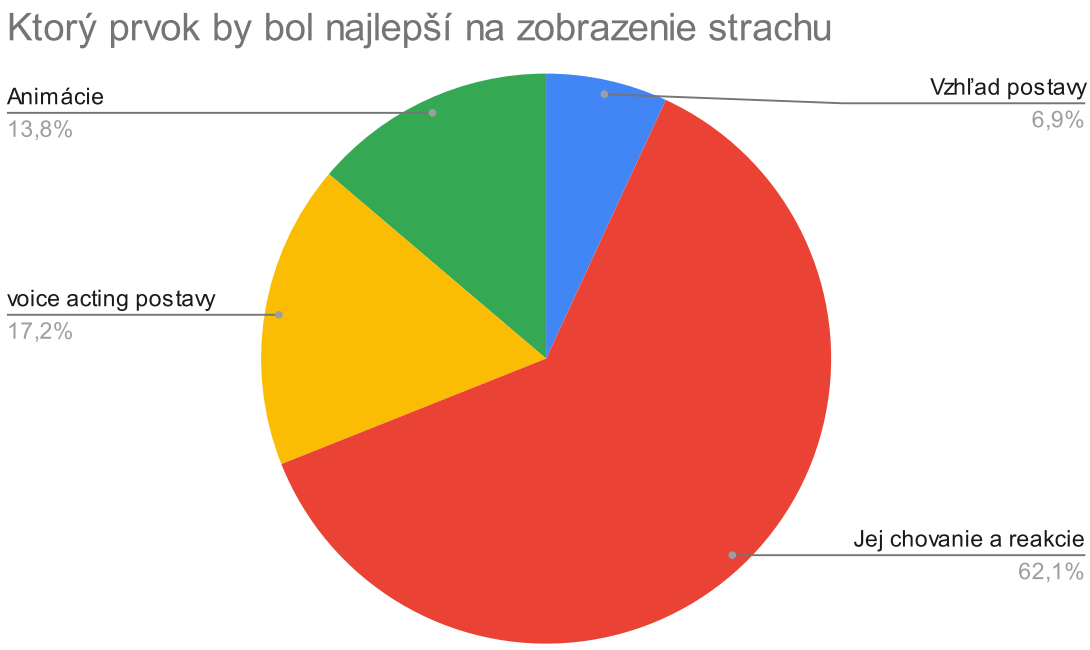
- Jej chovanie a reakcie na hráča
- Vzhľad postavy
- Animácie
- Voice acting[1] postavy

Užívatelia preferujú pri UI jej chovanie a reakcie. Na druhom mieste sú animácie týchto NPC postáv. Čo môže značiť ich dôležitosť. Avšak pri otázke: „Ktorý prvok by bol najlepší na zobrazenie strachu“ z tých istých prvkov vidno na grafe 6.2, že sa užívatelia prikláňajú aj k voľbe voice actingu a animácii, kde avšak je vzhľad postavy najmenej obľúbeným prvkom.

[1] Dabing postavy herného alebo filmového sveta



Obr. 6.1: Koláčový graf zobrazujúci preferenciu užívateľov pri zobrazení emócii UI



Obr. 6.2: Koláčový graf zobrazujúci preferenciu užívateľov pri zobrazení emócii strachu UI

6.2 Situácie vyvolávajúce strach

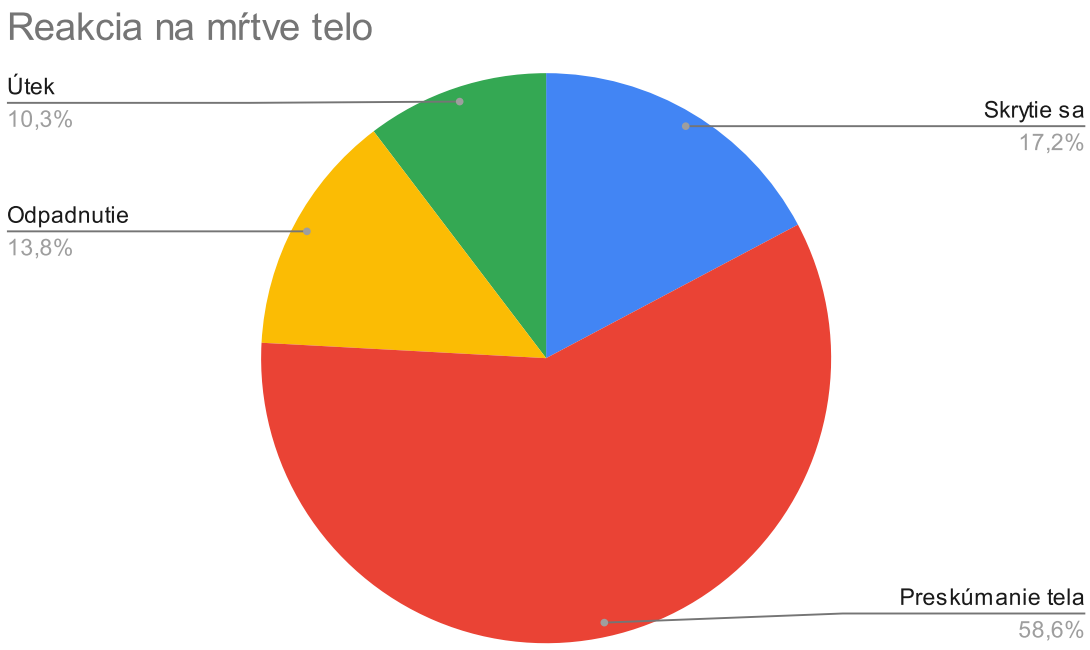
Ďalej boli využité otázky pre približné overenie správnosti postupov chovania použitých v práci, kde užívatelia opäť vyberali z predefinovaných možností na riešenie situácie.

6.2.1 Mŕtve telo

Jedna z nich bola otázka: „Ako by ste reagovali na mŕtve telo?“. Užívatelia mali možnosť vybrať zo zoznamu možností nasledovného:

- Útek
- Skrytie sa
- Preskúmanie situácie
- Odpadnutie

Zaujímavá bola zhoda, navrhnutej scény mŕtveho tela, na obrázku 4.9 z návrhu, a prieskumu. Podľa výberu najprv užívateľ preskúma situáciu, skontroluje telo a keď zistí, že je mŕtve, odpadne alebo utečie. Výsledné dáta sú zobrazené na grafe 6.3.



Obr. 6.3: Koláčový graf zobrazujúci preferenciu užívateľov pri výbere reakcie na situáciu nájdenia mŕtveho tela

6.2.2 Konfrontácia so zbraňou

Otázka bola: „Ako by ste reagovali na hráča so zbraňou?“. Užívatelia mali možnosť vybrať zo zoznamu možností nasledovného:

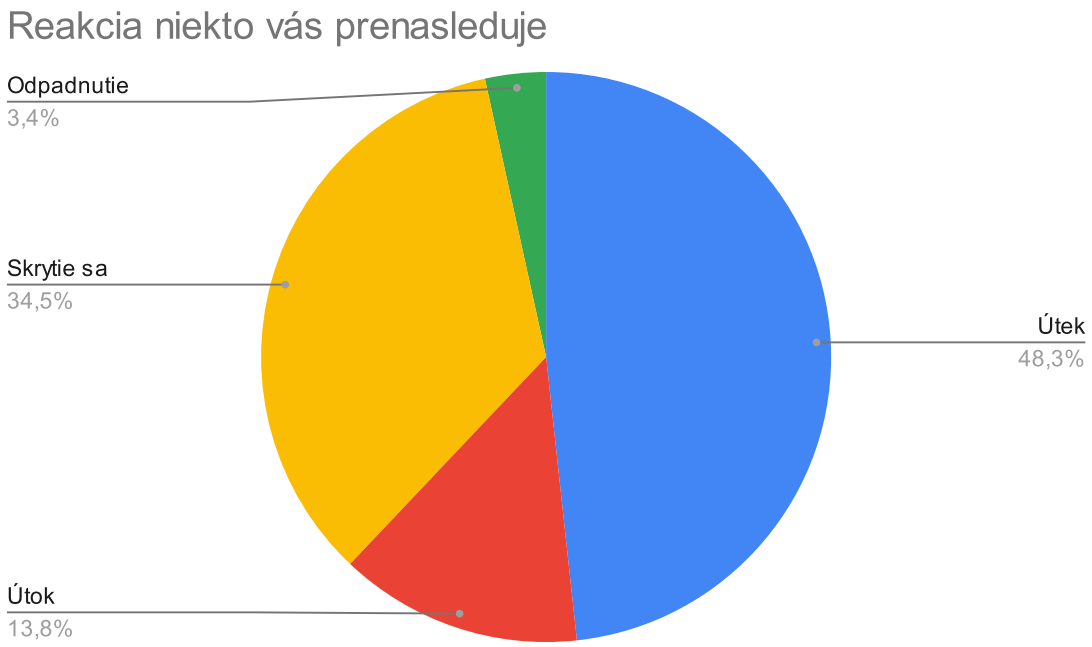
- Útek
- Skrytie sa
- Preskúmanie situácie
- Odpadnutie
- Útok

Naopak v tomto scenári, mnou navrhnutá scény 4.7 nesúhlasila s prieskumom. Podľa výberu by sa užívateľ skryl alebo vzdal. Útok je len tretia najčastejšia možnosť, zatiaľ čo práca ho zobrazuje ako prvotnú reakciu. Výsledné dáta sú zobrazené na grafe 6.4.

Obr. 6.4: Koláčový graf zobrazujúci preferenciu užívateľov pri výbere reakcie na situáciu stretnutia človeka so zbraňou

6.2.3 Prenasledovanie

Otázka bola: „Ako by ste reagovali na prenasledovanie?“. Užívatelia mali možnosť vybrať zo zoznamu možností rovnakého ako v podkapitole 6.2.2. Štúdia súhlasí s výsledkami štúdie z kapitoly 2.1, kde postupnosť úteku a skrytia sa sú rovnaké. Výsledné dáta sú zobrazené na grafe 6.5.



Obr. 6.5: Koláčový graf zobrazujúci preferenciu užívateľov pri výbere reakcie na situáciu prenasledovania

6.3 Hodnotenie scén

Hodnotenie scenárov prebiehalo na základe video demonštrácie konkrétnej scény a po prezretí videa bola položená séria otázok:

- Boli inštrukcie jasne zrozumiteľné?
- Ako vierohodne je vyzobrazená scéna od 1 do 5? (1 najhoršie 5 najlepšie)
- Čo bol najlepší prvok scény? Možnosti animácia reakcii, textové oznámenia stavu v ktorom sa NPC nachádza, naprogramovaná reakcia, oranžový ukazovateľ strachu, textové správy od umelej inteligencie
- Čo v scéne chýbalo, aby bola scéna lepšie zobrazená - krátka textová odpoveď

Otázka, či sú inštrukcie na začiatku levelu jasne zrozumiteľné bola vo všetkých prípadoch 90% áno. Dáta pre najlepší prvok boli vyhodnotené z grafov v prílohe B.

6.3.1 Mŕtve telo

Ohodnotenie scény malo medián 3,9 z 5, ktorý vytvára priestor na vylepšenia. Najlepšími prvkami boli podľa užívateľov ukazovateľ strachu a naprogramovaná reakcia. Nedostatkami boli podľa nich hlavne pomalá reakcia na situáciu a animácia odpadnutia.

6.3.2 Konfrontácia so zbraňou

Táto scéna bola ohodnotená na 4,3 z 5. Najlepšími prvkami boli podľa užívateľov naprogramovaná reakcia a animácia. Nedostatkami boli podľa nich hlavne, že UI by mala reagovať skôr na prítomnosť hráča so zbraňou a napríklad poznámka od jedného z užívateľov, kde by UI mala vyhodnotiť či stíha ísť pre zbraň, predtým ako ju hráč konfrontuje.

6.3.3 Prenasledovanie

Ohodnotenie scény malo najvyšší medián 4,5 z 5. Najlepšími prvkami boli podľa užívateľov textové správy UI a textové oznámenia aktuálneho stavu. Nedostatkami boli podľa nich instantná regenerácia zdravia a strachu v kryte.

Kapitola 7

Záver

V práci sú vytvorené simulácie scén zobrazujúce strach za pomoci stromov chovania. Podarilo sa vytvoriť veľa univerzálnych uzlov, ktoré slúžia pre viac typov scén, čím sa potvrdila znovu použiteľnosť a jednoduchá modifikácia oproti konečným automatom. Dôležitým výsledkom je takisto využiteľnosť simulačných uzlov jednotlivých scén v klasickom režime akčnej hry z pohľadu prvej osoby, ktorá bude nadstavbou doterajšej práce.

V budúcnosti na základe vybudovaných simulácii bude možné vytvoriť hru a v nej leveli využívajúce implementované stromy chovania. Tým pádom by bolo možné mať dva samostatne fungujúce prvky a to simuláciu a hru, ktoré by mohli každé samostatne vyjsť. Simulácia by mohla v budúcnosti poskytnúť zábery z rôznych uhlov a tým pádom využiť radšej pozíciu pozorovateľa ako hráča.

Budúce možnosti zlepšenia vidím v pridaní tepu srdca, ktorý ako samostatná premenná bude ovplyvňovaný faktormi strachu a zdravia, ktorá by umožňovala vidieť hráča za objektmi na vzdialenosť závisiacej od tepu srdca. Hráč by mal teda špeciálnu schopnosť pozorovať nepriateľov s vysokým úderom srdca za stenami.

Hlavným vylepšenie v budúcnosti vidím vo využití novej technológie od konkurenčného herného enginu Unreal Engine, ktorý ponúka modifikovateľné humanoidné postavy, ktoré môžu byť editované a hlavným faktorom je vytváranie emócii na tvári, ktorý by zdokonalil zobrazenie strachu.

Užívateľská štúdia ukázala, ktoré scény a čo v nich treba zlepšiť a hlavným problémom je pomalé budovanie strachu a tým pádom aj pomalé reakcie UI, ktoré bude potrebovať balansovanie pre užívateľský zážitok. Výsledky štúdie takisto ukazujú, že animácie sú potrebným dopĺňajúcim prvkom pri zobrazení emócie strachu. Túto problematiku je potrebné zvládnuť pre zdokonalenie výsledku vykresľovania animácii všeobecne.

Prácu vnímam ako prínos zriedka používanej techniky zobrazovania strachu, ktorá skrýva mnoho výziev, ktoré sa budem snažiť v budúcnosti riešiť za pomoci skúseností, ktoré som získal pri tvorbe umelej inteligencie, aby tento nápad pokračoval vo svojej realizácii.

Literatúra

[1] Bracken Grissom, H., Ahyong, S., Wilkinson, R., Feldmann, R., Schweitzer, C. et al. The Emergence of Lobsters: Phylogenetic Relationships, Morphological Evolution and Divergence Time Comparisons of an Ancient Group (Decapoda: Achelata, Astacidea, Glypheidea, Polychelida). *Systematic biology*. 1. vyd. Február 2014, č. 63, s. 457–479.

[2] Cologne, G. I. f. Q. a Health Care, E. in. *How does the pituitary gland work?* [online]. Medical News Today, apríl 2018 [cit. 2022-05-10]. Dostupné z: https://www.ncbi.nlm.nih.gov/books/NBK279389/.

[3] Kasia Kozlowska, L. M. a. P. C. *What is the fight, flight, or freeze response?* [online]. Medical News Today, august 2021 [cit. 2022-01-29]. Dostupné z: https://www.medicalnewstoday.com/articles/fight-flight-or-freeze-response.

[4] Kyong Sok "KC" Chang, D. Z. *Hierarchical Finite State Machine (HFSM) & Behavior Tree (BT)* [online]. Stanford University, október 2015 [cit. 2022-5-15]. Dostupné z: https://web.stanford.edu/class/cs123/lectures/CS123_lec08_HFSM_BT.pdf.

[5] Laviola, G. et al. *Neuroscience & Biobehavioral Reviews*. 1. vyd. Elsevier, 2022. ISBN 80-200-1327-X.

[6] Michele Colledanchise, P. O. *Behavior Trees in Robotics and AI: An Introduction*. 1. vyd. CRC Press, 2018. ISBN 1138593737.

[7] Newman, T. *Dissecting terror: How does fear work?* [online]. Medical News Today, október 2021 [cit. 2022-05-10]. Dostupné z: https://www.medicalnewstoday.com/articles/323492.

[8] Paravati S, W. S. *Physiology, Catecholamines* [online]. Medical News Today, október 2021 [cit. 2022-05-10]. Dostupné z: https://www.ncbi.nlm.nih.gov/books/NBK507716/.

[9] Ray Barrera, A. S. K. et al. *Unity AI Game Programming*. 2. vyd. Packt Publishings Ltd., 2015. ISBN 978-1-78528-827-2.

[10] Thau L, S. S. *Physiology, Cortisol* [online]. Medical News Today, september 2021 [cit. 2022-05-10]. Dostupné z: https://www.ncbi.nlm.nih.gov/books/NBK538239/.

[11] Unity. *Manual* [online]. Unity, máj 2022 [cit. 2022-05-08]. Dostupné z: https://docs.unity3d.com/Manual/.

Príloha A

Obsah priloženého pamäťového média

Priložené médium sa skladá z nasledujúcej adresárovej štruktúry:

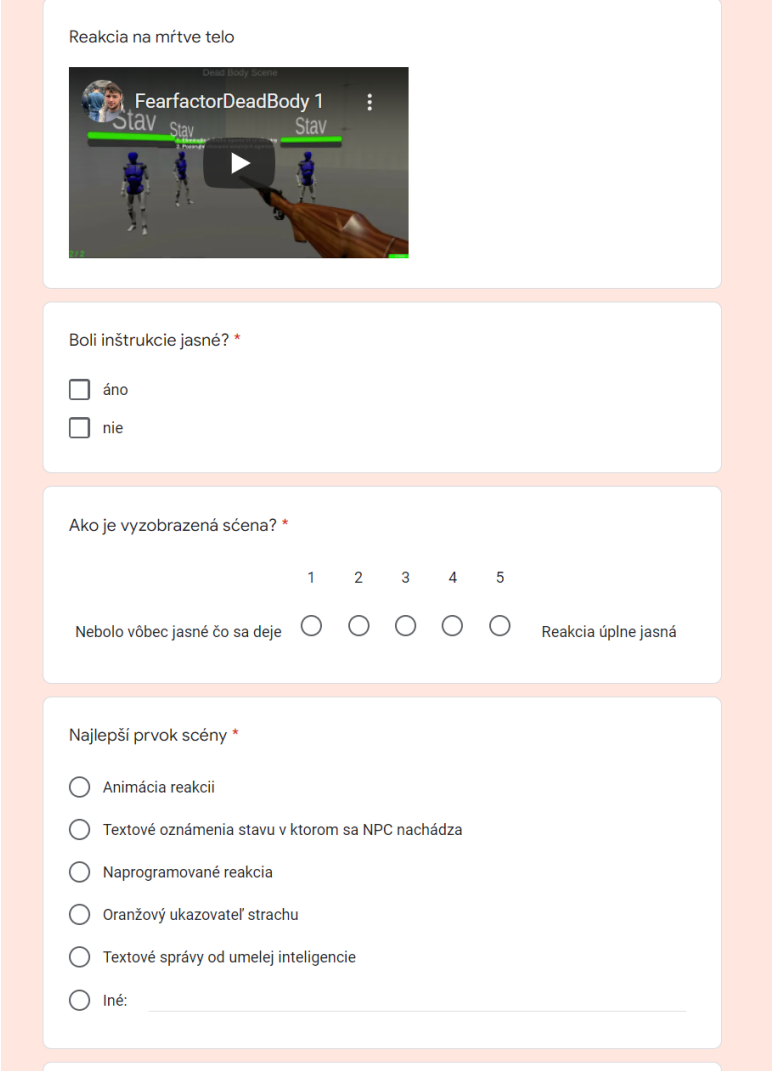
- src – zdrojové súbory simulácie
- src_latex – zdrojové súbory textovej časti práce
- executable – obsahuje build aktuálnej verzie aplikácie spustiteľnej na zariadeniach s operačným systémom windows

Ďalej adresár obsahuje video, ktoré demonštruje simulačné scény aplikácie, súbor readme, ktorý obsahuje spôsob akým spustiť projekt vo engine Unity.

Príloha B

Dotazník užívateľskej štúdie

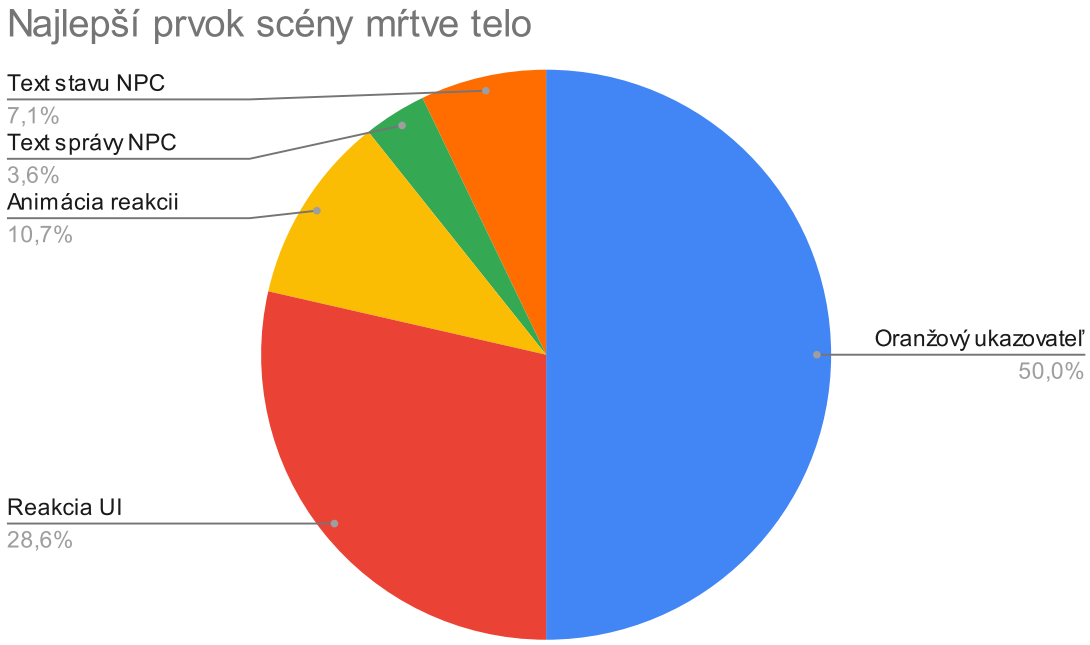
Náhľad na dotazník vytvorený v aplikácii Google Forms je na obrázku B.1.



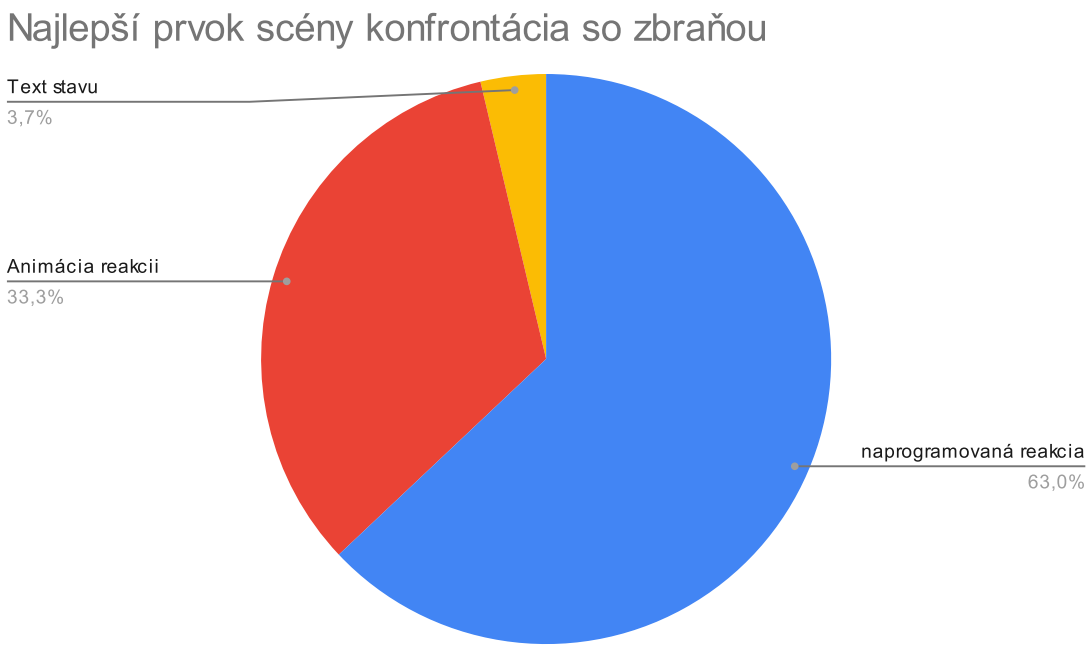
Obr. B.1: Dotazník obsahujúci video scénu a následné otázky ku kvalite spracovania danej scény

B.1 Grafy scén

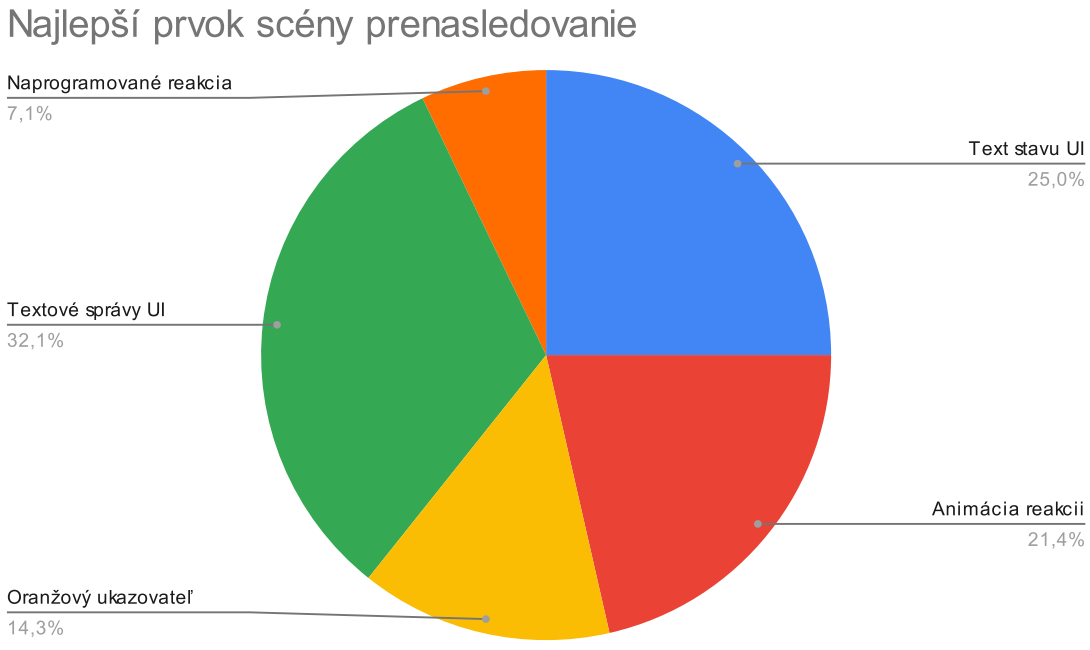
Grafy referencované z kapitoly 6.3 sú na nasledujúcich grafoch. Najlepší prvok scény mŕtve telo na grafe B.2. Najlepší prvok scény konfrontácia so zbraňou na grafe B.3. Najlepší prvok scény prenasledovanie na grafe B.4.



Obr. B.2: Koláčový graf zobrazuje výber najlepšieho prvku scény mŕtveho tela



Obr. B.3: Koláčový graf zobrazuje výber najlepšieho najlepšieho prvku scény konfrontácia so zbraňou



Obr. B.4: Koláčový graf zobrazuje výber najlepšieho prvku scény prenasledovanie