Bachelorprojekt

Kunstig intelligens og spillerens progression i computerspil

Udarbejdet af:

Julian Møller	180387-1619	boolean@diku.dk
Steen Nordsmark Pedersen	150287-1435	acid@diku.dk
Klaes Bo Rasmussen	041087-1939	klaes@diku.dk

Vejleder: Jon Sporring

Datalogisk Institut
Københavns Universitet

Juni 2010

DADIU THE NATIONAL ACADEMY OF DIGITAL INTERACTIVE ENTERTAINMENT

Abstract

We describe a company internship at The National Academy of Digital, Interactive Entertainment (DADIU), where we developed three games. We have extended the games with functionality to log data, which we can use to create heatmaps. We have tested our games with children, and have found, that artificial intelligence is a factor that needs to be prioritized when designing computer games for 6-7 year olds.

Resumé

Vi beskriver et ophold i virksomhedspraktik på Det Danske Akademi for Digital, Interaktiv Underholdning (DADIU), hvor vi har udviklet tre spil. Vi har udvidet spillene med funktionalitet til at logge data, som vi kan lave heatmaps ud fra. Vi har testet vores spil med børn, og har fundet, at kunstig intelligens er en faktor, der skal prioriteres når et computerspil til 6-7-årige skal designes.

Formaliteter

Afsnittet om produktionen af Catnapped, samt delkonklusionerne for Catnapped på baggrund af vores test er skrevet af Klaes Bo Rasmussen.

Afsnittet om produktionen af Imachination, samt delkonklusionerne for Imachination på baggrund af vores test er skrevet af Steen Nordsmark Pedersen.

Afsnittet om produktionen af Sophie's Dream Leap, samt delkonklusionerne for Sophie's Dream Leap på baggrund af vores test er skrevet af Julian Møller.

Resten af rapporten er skrevet i fællesskab mellem Julian Møller, Klaes Bo Rasmussen og Steen Nordsmark Pedersen.

Indhold

Indhold **iii**

Figurer **v**

Tabeller **vii**

1 Indledning **1**

2 Problemformulering **2**

3 Begrænsninger for projektet **3**

4 DADIU **4**

5 Begrænsninger for spillene **6**

6 SCRUM **7**

7 Introduktion til Unity **8**

8 Vores projekter **9**
8.1 Catnapped: The Bear Affair 9
8.2 Imachination 15
8.3 Sophie's Dream Leap 23

9 DADIU evaluering **30**
9.1 Evaluering på ITU 31

10 Unity evaluering **33**
10.1 Fordele ved Unity 33
10.2 Ulemper ved Unity 34

11 Sammenligning af vores KI'er **35**

12 Afprøvning **36**
12.1 Forventninger ved brugertest 36
12.2 Testplan 37
12.3 Testresultater 38
12.4 Delkonklusioner 44
12.5 Metoderefleksion 47
12.6 Refleksion over produktionerne 48

13 Diskussion **49**

14 Konklusion **51**

Appendiks **53**

A Spillene **54**

B Litteratur **55**

C Heatmaps **57**
C.1 Imachination 57
C.2 Catnapped 61
C.3 Sophie's Dream Leap 66

D Fulde DADIU begrænsninger **69**

Figurer

8.1 Startmenuen, der viser hvor Catnapped starter. 10
8.2 Illustration af katteøjne, tændstikæsker og kakerlak. 11
8.3 Slutbilledet for Catnapped. 11
8.4 Startmenuen som ses når spillet startes. 15
8.5 Billede mellem hver bane viser spillerens progression i spillet. . . 16
8.6 Værkstedet mellem hver bane, hvor man kan bygge på sin maskine. 17
8.7 Bane nummer to i Imachination. 17
8.8 De to styringsplaner 20
8.9 Menuen i Sophie's Dream Leap 23
8.10 Gennemgang af styremåde 24
8.11 Jorden er blevet giftig. KI'en (øvre venstre hjørne) hopper igennem banen. 28
8.12 Fjenderne i Sophie's Dream Leap 29

12.1 2P's gennemløb af Imachination 42
12.2 KI'ens rute gennem banen 42
12.3 8Ds to gennemløb af Sophie's Dream Leap 47

C.1 1D. Spillede uden KI først. 57
C.2 2P. Spillede med KI først. 58
C.3 3D. Spillede med KI først. 58
C.4 4P. Spillede med KI først. 58
C.5 5D. Spillede uden KI først. 59
C.6 6D. Spillede uden KI først. 59
C.7 7D. Spillede kun med KI. 59
C.8 8D. Spillede uden KI først. 60
C.9 9D. Spillede uden KI først. 60
C.10 1D. Spillede kun med KI. 61
C.11 2P. Spillede kun uden KI. 62
C.12 3D. Spillede uden KI først. 62
C.13 4P. Spillede kun med KI. 63

C.14 5D. Spillede med KI først. 63
C.15 6D. Spillede med KI først. 64
C.16 8D. Spillede med KI først. 64
C.17 9D. Spillede med KI først. 65
C.18 1D. Spillede kun uden KI. 66
C.19 2P. Spillede kun uden KI. 66
C.20 4P. Spillede med KI først. 66
C.21 5D. Spillede med KI først. 67
C.22 6D. Spillede uden KI først. 67
C.23 7D. Spillede uden KI først. 67
C.24 8D. Spillede uden KI først. 67
C.25 9D. Spillede med KI først. 68
C.26 10D. Spillede med KI først. 68

Tabeller

12.1 Oprindelig testplan 37
12.2 Revideret testplan 39
12.3 Gennemførselstider for Catnapped 39
12.4 Gennemførselstider for Imachination 41
12.5 Gennemførselstider for Sophie's Dream Leap 43
12.6 Gennemførselstider for Sophie's Dream Leap. Første/andet gennemløb. 46
12.7 Antal dødsfald i Sophie's Dream Leap 46

Indledning 1

Vi har udviklet tre spil i forbindelse med et ophold i 'virksomhedspraktik' på DADIU[3]. Vi indgik i produktionshold på cirka 15 mennesker fordelt over en række forskellige specialiteter.

Fra DADIU's side blev der udstukket begrænsninger til spillene. Et af dem var, at spillene skulle henvende sig til børn i 3-7-års alderen, hvor det på alle vores produktioner blev besluttet, at indskrænke intervallet til 6-7 år. Et andet krav var, at der skulle være kunstig intelligens, som en central del af *gameplayet*.

Man kan derfor stille sig selv spørgsmålet, om tilstedeværelsen af kunstig intelligens overhovedet gør en forskel i spil rettet mod børn. Bruger børnene samme mængde tid på det samme spil, hvad enten det indeholder en kunstig intelligens eller ej?

Spørgsmålet er interessant, da der bliver udviklet flere og flere computerspil målrettet mod børn, og det derfor relevant at finde ud af, om kunstig intelligens er en faktor, der skal overvejes, når et computerspil designes. Spørgsmålet er endvidere interessant, idet der ikke umiddelbart er nogen tidligere litteratur at finde omkring kunstig intelligens' påvirkning af børns spiloplevelse.

Vi vil derfor, ved hjælp af tre spil, hvor de kunstige intelligenser spiller forskellige roller, undersøge om den kunstige intelligens har nogen indvirkning på spiloplevelsen.

I rapporten vil vi begynde med at beskrive DADIU og herefter de tre produktioner, vi hver i sær var en del af. Vi vil herunder evaluere ting, der har været specifikke for vores produktioner. Efterfølgende vil vi evaluere DADIU samt udviklingsmiljøet Unity3D (Unity)[10] generelt. Vi vil beskrive vores forsøg, hvad vi måler, og hvorfor vi måler det. Vi vil beskrive vores testplan og argumentere for, hvorfor den er udformet, som den er. Til slut vil vi diskutere og evaluere vores resultater og spil på baggrund af den udførte test. Vi vil også vurdere, om tilstedeværelsen og typen af en kunstig intelligens, i et spil målrettet mod børn, er en faktor der skal overvejes af *game designere* i forbindelse med spiloplevelsen.

Problemformulering

2

- Hvordan er DADIU-produktionerne forløbet?
- Hvilke problemer har der været på produktionerne?
- Hvordan er resultatet af produktionerne?
- Hvilke problemer er der ved at udføre brugertest med børn?
- Bør kunstig intelligens være en faktor, der prioriteres når spil til børn designes?

Begrænsninger for projektet

3

Da spilproduktionernes målgruppe var 6-7 år, er vores målgruppe den samme.

Idet målgruppen er børn. vil vi holde os fra at forsøge at måle subjektive vurderinger om 'sjovhed', evne til at underholde og lignende kvalitetsangivelser. Det gør vi, fordi det er en videnskab i sig selv at få relevant data ud af børn [8][2, side 4, første afsnit]. Derfor har vi valgt at fokusere på objektiv data, der beskriver spillerens progression gennem banerne, og ud fra disse vurdere, om den kunstige intelligens giver grobund for alternative spiloplevelser.

DADIU

4

DADIU er et samarbejde mellem en række uddannelsesinstitutioner. Hver institution bidrager med studerende med forskellige kompetencer, der sammen skal indgå i en computerspilsproduktion.

De roller, der var repræsenteret ved vores produktioner var:

- Projektleder
- Instruktør
- *Game designer*
- *Lead*-programmør
- Programmør
- *Art director*
- *Visual designer*
- *CG artist*
- Animator
- *Audio designer*

Derudover deltager arkitektskolen også og bidrager med en *Level designer* på afgangsproduktionen, der foregår i marts.

DADIU-produktioner er en mulighed for at lave spil i et miljø, der minder om det, der er at finde i rigtige spilvirksomheder. Selvom DADIU-produktionen blot er et studieprojekt, kommer både tidspresset og sammensætningen af holdet tæt på virkeligheden i branchen. Mange lignende forløb andre steder er udelukkende frivillige initiativer og det er kun DADIU, som man kan deltage i som en del af sit studieforløb.

Før produktionerne begynder, finder et fællespensumskursus sted. Det er beregnet til at lære alle de studerende noget om hvad de andre produktionsdeltagere kan. I to uger kommer folk fra de forskellige repræsenterede studier og spilbranchen og fortæller om deres felt og giver et minikursus i centrale elementer. Det giver et godt billede af hvad man kan forvente af de andre på holdet, omend kurserne i ens eget felt kan være lidt trivielle.

Efterfølgende er der et tre dages intensiv kursus hvor der bliver lavet øvelser med at udvikle brætspil. Kursets formål er at lære de studerende, hvad der er godt og dårligt ved et spil, og hvad der skal til for, at et spil er sjovt.

Når produktionerne er gået i gang, får holdene ugentligt besøg af en konsultationsgruppe. Konsultationsgruppen består af folk fra DADIU. Ved et af møderne er der også en person fra et professionelt spilfirma til stede, som skal agere spiludgiver. Holdene skal ved hvert besøg *pitche* og fremvise spillet. Konsultationsgruppen vil så komme med feedback og komme med gode råd.

Pitching er den fremlæggelse man laver, når man typisk skal sælge eller søge finansiering til sit produkt. Man skal fremvise sit produkt på en kort og kompakt måde.

Selve produktionen består af tre faser. Den første fase er onlinefasen. Her kan holdet komme med ideer og brainstorme over hvilket spil de kunne tænke sig at lave. I denne fase har projektlederen til opgave at tilrettelægge produktionsforløbet. Den næste fase er præproduktionen. Holdet mødes for første gang og skal blive enige om, hvilket spil der skal laves. Der skal udvikles en virkende prototype, samt designdokumenter, som skal afleveres og fremvises til konsultationsgruppen efter en uge.

Dernæst starter produktionsfasen. Gruppen har nu tre uger til at udvikle, færdiggøre og finpudse spillet, før det skal præsenteres i plenum foran et dommerpanel, de andre grupper og interesserede folk fra branchen.

Begrænsninger for spillene

5

Vores produktioner havde nogle begrænsninger udstukket af DADIU. De væsentligste var som følger:

- Spillet skulle kunne spilles som et *single player*-spil.
- Spillet skulle inkludere 3D-elementer og en hovedperson
- Målgruppen skulle være 3-7-årige børn (hvor vi i alle vores tre grupper valgte at indsnævre det til 6-7-årige)
- Efter maksimalt 5 minutters spil skal man have oplevet en fuld spiloplevelse.
- Det endelige spil skulle testes på målgruppen
- Spillet må kun bruge musen og venstre musetast til styring
- Spillet skulle indkludere kunstig intelligens
- Spillet skulle være både på dansk og engelsk
- Spillet skulle udvikles i Unity
- Spillet skulle kunne spilles på internettet ved hjælp af Unitys *web player* på en 'gennemsnitlig'[1] pc.

De fulde begrænsninger kan ses i appendiks D.

[1] Baseret på Unitys hardwareundersøgelser: `http://unity3d.com/webplayer/hardware-stats`.

SCRUM 6

SCRUM er en fleksibel måde at udvikle software på. I stedet for de traditionelle udviklingsmetoder, som for eksempel vandfaldsmetoden, erkender man i *SCRUM*, at der opstår uforudsigelige ting gennem en udviklingsproces. Man har derfor lavet et system, som nemt kan håndtere ændringer. I *SCRUM* arbejder holdet i 'sprints'. I traditionel *SCRUM* varer en sprint typisk to til fire uger. Ved enden af hver sprint skal der være et potentielt færdigt produkt.

I *SCRUM* har ejeren af produktet (ofte projektlederen) en 'product backlog'. Her kan ejeren lægge så mange krav i bunken, som han lyster. Hvert krav skal have en prioritering, og kravene med højest prioriteting er dem, der bliver lavet først. Ved starten af hver sprint estimerer holdet, hvor meget de kan nå i løbet af sprinten. De krav, de kan nå at opfylde flyttes over i en 'sprint backlog'. Hver dag under sprinten holdes der et stående møde, hvor alle på holdet fortæller, hvordan tingene står til.

SCRUM har en række fordele i forhold til de traditionelle metoder og især til mindre udviklingshold. Først og fremmest er der stor fleksibilitet. Det er nemt at justere tingene, hvis noget tager længere tid end forventet, eller hvis der opstår uventede problemer. Det er også nemmere at hjælpe hinanden internt på holdet, da alle ved, hvad de andre laver, samt hvilke problemer der måtte være. Man har også hele tiden et færdigt produkt der virker.

Introduktion til Unity

7

Unity er en den *editor*, vi brugte på vores DADIU-produktionener. Den er baseret på *WYSIWYG*[1]-princippet. Unitys primære formål er at gøre udviklingsprocessen let.

Alle projekter i Unity indeholder en lang række *assets*: Lydfiler, grafik, videoer, teksturer, 3D-modeller og *scripts*. Disse objekter kan man kombinere til nye 'superobjekter', som i Unity går under navnet *prefabs*. Fordelene ved at udnytte dem, er at man kan genbruge komplicerede objekter på tværs af baner og nøjes med at ændre eller opdatere dem ét sted. En bane i et Unity-spil kaldes en 'scene'. Scener kan dog også bruges til *loading*-skærme, menuer, mellemsekvenser og rulletekster. Unity påfører dermed den objektorienterede tankegang til en *drag and drop*-editor.

Unity lægger op til, at man programmerer sine spil, så de er *event*-baserede. Der er indbygget funktionalitet til *trigger*-zoner, hvor man automatisk kan sørge for, at der sker noget når et objekt kommer ind i zonen, opholder sig i zonen eller forlader zonen. Det kan man for eksempel bruge til at have områdespecifik kamerastyring.

Koordinatsystemet i Unity er vendt således, at (x, z)-planet er vandret og y-aksen peger opad.

[1] What You See Is What You Get

Vores projekter

8

Vi deltog efter eget ønske i tre forskellige spilproduktioner. Gruppedannelsen fandt sted, før kravene til spillene blev offentliggjort, og derfor ønskede vi at maksimere vores chancer for at få noget KI-materiale at skrive om.

I de følgende afsnit beskrives vores individuelle produktioner, der resulterede i tre vidt forskellige spil, med tre kunstige intelligenser med vidt forskellige formål.

8.1 Catnapped: The Bear Affair

Catnapped er en historie om en teddybamse og en kat. De bor i samme hus og rivaliserer om at få kærligheden fra drengen, der bor der. Vi ser på figur 8.1 hvordan spillet starter, med billede af drengen med kat og bamse i armene ovre til højre.

Spillet

Spillet begynder med at katten fanger vores hovedperson Teddy og putter ham i vaskemaskinen. Nu gælder det om som spiller at få styret Teddy gennem en række udfordringer i kampen for at undslippe den onde kat.

Den første bane udspiller sig i vaskemaskinen, hvor man langsomt bliver hvirvlet rundt mellem sokkerne. Spilleren skal forsøge at kaste den snor Teddy har på ryggen, hen på et håndtag inde i vaskemaskinen for at komme fri.

Næste bane handler om balance. Spilleren skal passere henover brækkede gulvbrædder uden at falde ned. Teddy er våd og tung, hvilket gør styringen vanskelig.

Tredje bane starter med at vise en ventilator, der står højt oppe. Spilleren får fat i ventilatoren med Teddys snor, og skal modvirke vindkraften for at trække sig helt tæt på ventilatoren. Når spilleren er kravlet ned fra ventilatoren, skal han bane sig vej til en skakt.

Teddy falder ned igennem skakten og befinder sig nu i en labyrint. For at forhindre at Teddy undslipper, forsøger katten at fange Teddy i lyskeglerne

Figur 8.1: Startmenuen, der viser hvor Catnapped starter.

fra sine øjne. Rundt omkring i labyrinten ligger der tændstikæsker. Disse kan bruges til at gemme sig bag, så katten ikke ser Teddy. Vi kan på figur 8.2 se Teddy der gemmer sig bag ved tændstikæsker for at undgå at blive set af katteøjnene. Tændstikæskerne fungerer samtidig som forhindring da de visse steder skal skubbes væk, for at Teddy kan komme videre. I labyrinten løber der kakerlakker rundt, de er med til at skabe liv i banen. Vi kan se en kakerlak nederst på figur 8.2.

Når spilleren er undsluppet fra labyrinten, møder han katten, som blokerer udgangen. Det gælder om at kaste sin snor på benene af den skammel, katten står på, så de går i stykker. Der er en *health bar*, der viser, hvor langt katten er fra at falde ned. Under kampen forsøger katten, som i labyrinten, at fange spilleren med øjnene.

Er spilleren sejrrig, undslipper Teddy og han kan sidde i ro og mag i sin skotøjsæske og blæse sæbebobler (figur 8.3).

Produktionsforløb

Vi brugte første dag i produktionsforløbet på at lære hinanden at kende. Vi diskuterede, hvilke muligheder vi havde for styring af spillet. Det blev til fire forskellige styringsmetoder. Jeg fik til opgave næste dag at implementere en af styremåderne til Teddy: Spilleren kunne kigge med musen og rykke sig fremad, når venstre musetast blev holdt nede. Dette var den mest simple styremåde. Andre løsninger indebar blandt andet, at man skulle bevæge sig

Figur 8.2: Illustration af katteøjne, tændstikæsker og kakerlak.

Figur 8.3: Slutbilledet for Catnapped.

rundt som om man havde vægt på musen: Hvis man holdt den til højre for Teddy, ville han vælte mod højre og gå hurtigere, jo længere væk musen blev flyttet, indtil han til sidst faldt. Dette princip blev aldrig rigtig brugt, men det fik os til at komme på de første ideer til spillet. Vi havde overvejet at Teddy kunne samle ting i huset til et lille et-bamsesband. Det vil sige, at Teddy skulle spille på tingene han samlede op, for at finde ud af, om det var noget, han kunne bruge i sit multiinstrument.

På tredjedagen blev vores endelige ide lagt fast, og vi begyndte at udvikle. Vi skulle nå at lave en masse forskellige prototyper, da vores testgruppe skulle lave afprøvning allerede en uge senere.

For os programmører var der hele tiden noget at se til, da der var mange baner planlagt fra starten. Det blev programmørernes job at holde styr på banerne, sætte dem sammen i Unity og sørge for at de kunne spilles. Det meste af tiden arbejdede vi på en bane hver. Mit primære fokus endte med at være labyrint-banen.

Vi havde møder hver morgen på vores produktion. Møderne var en opdatering om, hvad der var sket den foregående dag. Vi havde opsamling af, hvilke nye arbejdsopgaver der var til den kommende dag, så alle havde mulighed for at finde noget at lave.

Mine arbejdsopgaver

Jeg stod først og fremmest for KI-delen af vores spil. Det endte dog med, at vi havde svært ved at få noget som helst KI implementeret i spillet. Se afsnit om, hvorfor indholdet af KI i vores spil blev så begrænset på side 15.

I stedet endte mine arbejdsopgaver med at være følgende:

- Leveldesign i labyrinten
- Waypoints til de øjne der følger spilleren
- Funktionalitet, så øjnene kunne fange Teddy
- Spilmenuen
- GUI-elementer
- Diverse lydimplementering

Brugertest

Vi testede vores spil på børn mange gange i udviklingsforløbet. Nogle af børnene var bedre end andre, men de hjalp ofte hinanden. Børnene havde ikke umiddelbart problemer med at forstå styringsmåden. Det var dog ikke tilfældet for de voksne, der prøvede spillet. Ved brugertesten var det kun nogle

af børnene, der forstod styremåden med det samme. Resten havde nogle problemer med det. Hvis nogle af de andre børn havde problemer med at styre var der straks hjælp at hente hos sidemanden. Dette var ikke tilfældet med for eksempel den spiludgiver, der kom ud for at se spillet. Vi blev til sidst nødt til at forklare ham, at han skulle blive ved med at holde venstre museknap nede for at bevæge sig.

Jeg fik aldrig direkte indsigt i, hvad vores resultater af brugertestene var. Programmørene hørte kun, at det gik godt hver gang, og at børnene arbejdede sammen om problemer. Det vil sige at vi ikke fik nogen rigtig indsigt i, om der var noget ved vores styremåder, der skulle ændres. Det havde været lettere at optimere styremåden og regulere sværhedsgraden af spillet hvis vi havde været med til afprøvningerne eller fået ordentlig feedback.

Problemer og udfordringer

En af de største problemer der blev afdækket af vores brugertest var, at det var svært at finde rundt i labyrinten. Vi startede med et kamera, der hele tiden var bag Teddy. Det medførte at spillerne mistede orienteringen og farede vild. Vi fandt ud af at vi var nødt til at have et mere stationært kamera, så det var klarere, hvilken vej man gik. I første udgave roterede kameraet 90 grader af gangen, men spillerne blev stadig desorienterede. Banen så for anderledes ud fra de forskellige vinkler. Derfor endte vi med at fjerne al rotation fra kameraet og placere det over spilleren. Her fik vi det problem at kameraet let kom til at være placeret bag væggene i banen, så man ikke kunne se Teddy. Vi forsøgte at løse dette ved at finde den rigtige højde for kameraet. Det gjorde, at man både fik fornemmelsen af at kunne se toppen af kasserne, men det gjorde også, at kameraet ikke kom for langt væk fra Teddy.

KI i spillet

Den kunstige intelligens i spillet endte med at være kattens øjne i både labyrinten og den sidste bane, som er en *boss*-kamp.

I labyrinten er der en række *waypoints*. Det bestemmes ved hjælp af *pathfinding*, hvilket *waypoint* spilleren er tættest på. Ved dette *waypoint* aktiveres katteøjnene, som søger efter spilleren i et forudbestemt mønster.

Hvis spilleren bliver opdaget inde i lyskeglen fra et sæt øjne, bliver han fanget af katten og skal starte banen forfra.

Øjnene er kun tændt ét sæt af gangen. Kun det sæt øjne som er fundet værende tættest på spilleren, af den indbyggede korteste vej-algoritme, er tændt. De korteste veje fra forskellige punkter til det nærmeste sæt øjne bliver kun fundet, lige når banen bliver lavet. Informationen bliver gemt i punkter i gulvet, så det er kun nødvendigt at finde ud af hvilket punkt der er tættest på spilleren for at finde det tætteste øjesæt.

I *boss*-banen forsøger katten også at fange spilleren i lyset fra sine øjne. For at forhindre det, kan spilleren gemme sig bag nogle kasser i begge sider. Når spilleren gemmer sig, følger øjnene efter. Derefter begynder katten at søge rundt omkring kassen for at forsøge at fange spilleren. Søgningen foregår vilkårligt.

Vi havde under hele produktionen diskussioner om hvor og hvordan vi kunne få noget KI inkluderet i vores projekt, da det ikke umiddelbart gav mening at have noget. Det var rent ekstraarbejde, og ingen af de andre var opsatte, på at der skulle god KI med i spillet. De mente ikke, at KI ville være essentielt og ej heller at det ville påvirke spiloplevelsen nævneværdigt.

Mulige forbedringer

I slutningen af forløbet fandt vi ud af, at vi burde have styret kameraet med *triggers* igennem hele banen for at få kameraet til at opføre sig optimalt, men det havde vi ikke tid til at implementere.

Det var vores mening at labyrinten skulle kunne genereres proceduralt så vi havde en ny labyrint, hver gang man startede banen. Det blev dog for stort et arbejde, men kunne have gjort banen meget mere interessant. Alle algoritmerne vi brugte i labyrinten blev derfor lavet så, de kunne bruges igen og igen, lige meget hvilken labyrintopsætning der blev anvendt. Skulle labyrinten genereres proceduralt, skulle der dog også være implementeret noget verificering, så det blev undersøgt, om det var muligt at finde vej igennem den genererede labyrint.

Vi overvejede på et tidspunkt, i udviklingen af spillet, at indføre en ny figur, som skulle hjælpe Teddy rundt i labyrinten. Det skulle være et møl, som blev tiltrukket af Teddy og blev hans ven. Møllet skulle kunne flyve og derved se labyrinten ovenfra. Idéen var, at man skulle kunne skifte kameravinkel og se møllets perspektiv, så man kunne få et lidt bedre overblik. En anden idé var at møllet på en måde skulle guide spilleren igennem labyrinten. Møllet skulle også kunne advare mod farer, det vil sige katteøjnene.

Da vi fandt ud af, at vi ikke ville proceduralt generere labyrinten fjernede vi også møllet som hjælperen og ændrede planen for at få indført KI i spillet. I stedet skulle der implementeres *flocking*[9] på de kakerlakker, der går rundt i banen. De skulle søge hen mod Teddy, når han kom indenfor en given radius af dem. Jo flere kakerlakker der var omkring Teddy, jo sværere skulle det være at bevæge sig rundt. Samtidig skulle katteøjnene begynde at følge hurtigere og hurtigere efter kakerlakkernes 'lyd', jo flere kakerlakker der var. Dette skulle gøre banen langt sværerere at komme igennem, men samtidig give mulighed for alternativt gameplay, da vi ville implementere mulighed for at træde på kakerlakkerne og dermed få dem til at forsvinde. Stødte man senere ind i nye kakerlakker, ville de begynde at følge efter spilleren på ny.

Hvorfor KI i spillet var begrænset

Det er delvist mit ansvar, at der til sidst ikke kom så meget KI med i spillet som jeg gerne ville have. Det tog mig for lang tid at prøve at finde en algoritme til 2D flocking på kakerlakkerne: Jeg burde have skrevet den selv fra starten. Jeg fik ikke lov til at inkludere det møl, vi havde overvejet, da det ville kræve for meget ekstra arbejde fra de andre på holdet. Til sidst var der for mange ting at få rettet igennem til, at jeg fik tid til at se på flocking. Problemet var også at der i for lav grad blev taget hensyn til, at jeg gerne ville have mere KI med i spillet. At få flocking på kakerlakkerne ville ikke ændre det generelle udtryk, man fik af spillet som helhed. Det ville have været en ekstra gode, men vores game designer kunne ikke se, hvilke muligheder det kunne åbne op for. Jeg mente, at det kunne give spilleren et alternativt mål med banen, idet de måske ville finde kakerlakkerne sjove at lege med.

8.2 Imachination

Figur 8.4: Startmenuen som ses når spillet startes.

Spillet viser produktet af et barns fantasi. Startmenuen, som ses på figur 8.4, viser en dreng, der leger i sin sandkasse. I sin leg skal han grave sig til Kina, og for at komme igennem jorden må han bygge en maskine og kæmpe mod underlige væsener og andre maskiner. På figur 8.5 ses spillerens progression gennem jorden efter første bane.

Figur 8.5: Billede mellem hver bane viser spillerens progression i spillet.

Der er taget udgangspunkt i børns betagelse af store maskiner. Især drenge er meget fascineret af, hvordan maskiner virker, og det er som regel altid populært med køretøjer. Vi har i spillet prøvet at opnå en 'wauw'-effekt ved at se på, hvordan børn fantaserer og leger. "Hvad er endnu bedre end en kanon? En større kanon".

Imachination er todelt. Den ene del er en byggedel, hvor man skal konstruere sin maskine ud fra forskellige våben. Den anden del er en *RTS*[1]-del, hvor spilleren skal manøvrere sin maskine rundt igennem underjordiske grotter og bekæmpe modstandere.

Man bygger sin maskine ved at trække våben ned i *sockets*. Det er ligeledes muligt at fjerne eller flytte våben. Man kan rotere sin maskine ved hjælp af pilene i siden af skærmen, eller ved at trække i maskinen. På figur 8.6 kan værkstedet ses.

I spilledenen kører maskinen fremad, når man holder venstre museknap nede. Maskinen roterer hele tiden for at holde fronten mod musen. Det er muligt at trykke på sig selv og derved affyre et *repulsor*-våben, som skubber fjender væk. Målet med spillet er at positionere sig selv, så det er muligt at ødelægge de bygninger, der befinder sig rundt i banerne. Derefter skal man besejre bossen. Begge dele naturligvis uden at dø. Skulle man dø, kommer man tilbage til værkstedet og må starte forfra på banen.

Alle banerne har form som sandkassen. Hvis man kigger nøje efter, kan man se at mange af ikonerne til våben, fjender og lignende kan findes i sandkassen. På figur 8.7 kan man se en af banerne i spillet.

[1] Real Time Strategy

Figur 8.6: Værkstedet mellem hver bane, hvor man kan bygge på sin maskine.

Figur 8.7: Bane nummer to i Imachination.

For at underbygge følelsen af, at det er et barn, der forestiller sig hele spillet, valgte vi at få børn til at indtale alle lydeffekter i spillet. Det er en håndfuld børn fra det lokale fritidshjem som, udover at teste spillet, har indtalt skydelyde, eksplosionslyde og meget mere. Det er også et barn, der har indtalt *voice-over* stemmen til vores trailer[2] på engelsk, selvom han ikke kunne tale sproget. Manuskriptet er blevet læst højt for ham, og han er så blevet optaget, mens han forsøgte at gentage ordene. Det bidrager til følelsen af, at han sad og legede for sig selv i sin sandkasse.

Forløb

Før selve produktionen begyndte, var vi en håndfuld fra holdet, der satte os ned og snakkede om, hvilket type spil vi kunne lave, samt hvilket tema det skulle have. Gruppen bestod af vores instruktør, *game designer*, projektleder, *art director*, *visual artist* og jeg som programmør. Vi fik sorteret i nogle tidlige brainstormidéer og fastlagt en mere konkret idé. I det første udkast var det et spil om byggepladser. Man var en byggemand, der konkurrerede mod de omkringliggende byggeprojekter. Samtidig med at man byggede sit eget op, skulle man sabotere de andres byggeprojekter. Det var først da konceptet fik indkorporeret fantasien om at grave sig ned til kineserne, at spillet ændrede sig til det nuværende. Da spillet henvender sig til små børn, skal man passe på, hvad der bliver slået og skudt efter: Det er mere acceptabelt at skyde efter underlige væsener end mennesker[1].

På den første dag af produktionen blev idéen *pitched* til hele gruppen. Da ideen faldt i god jord, sparede vi værdifuld tid i præproduktionen, da vi allerede få timer efter at have mødtes kunne begynde at udvikle prototyper.

Da vores spil består af to vidt forskellige dele, byggedelen og køredelen, startede programmørerne med at lave prototyper inddelt i to grupper. Sammen med en anden programmør blev jeg sat til at lave en prototype af en en kasse der kunne følge efter musen i et isometrisk perspektiv. De to dele skulle senere sættes sammen i spillet.

Arbejdsmetoder

Vores projekt blev kørt med *SCRUM*-møder hver morgen. Hver enkelt deltager på holdet fortalte, hvad der var blevet lavet dagen før, og hvad der var planlagt, at man skulle bruge dagen på. Derudover var der ikke meget styring af udviklingen. Der var møder, hvor vores *leads* fra hvert felt snakkede med instruktør, designer og projektleder, men udover det var der ikke meget koordinering med resten af gruppen. Internt på programmørholdet havde vores *lead* et lille overblik over, hvor langt vi var nået, men derudover arbejdede vi bare derudaf.

[2] `http://www.youtube.com/watch?v=WJ-ca5-hhWk`

Vores leadprogrammør forsøgte sig med at sætte post-its op på væggen, med de forskellige opgaver, der skulle løses, samt en estimeret tid af ogpaven. Man skulle tage en post-it, når man gik igang med en opgave og returnere den, når man havde fuldført den. For hver opgave skulle man notere, hvor lang tid den reelt tog at løse. Systemet gik dødt, da det ikke blev opdateret med nye opgaver gennem projektet. Vi endte med at holde en fejlliste på et *whiteboard*, som blev opdateret løbende.

Vi arbejdede alle 13 i samme rum. Det gav en del støj, og på grund af afbrydelser var det til tider svært at koncentrere sig. Til gengæld gav det mulighed for hele tiden at følge med i, hvad andre lavede. Det gjorde, at de forskellige faggrupper aldrig gik skævt af hinanden.

Mit fokus

Til at begynde med var jeg den programmør, der skulle bruge mest tid på KI'en grundet min bacheloropgave. Det var idéen, at jeg ville kunne bruge anden og tredje uge på at udvikle KI'en til spillet. Det viste sig hurtigt, at det var et meget urealistisk bud, da holdet var meget tidspresset af blot at få spillet færdigt uden avanceret KI. Derfor blev det ikke til så meget KI, som jeg havde håbet, men en del mere implementering af diverse ting til spillet.

Som før nævnt blev vores prototypeudvikling opdelt i to grupper. Jeg skulle lave vores bevægelses system, hvor maskinen skulle følge efter musen, hvis man holdt museknappen nede.

Maskinen roterer hele tiden mod det punkt i terrænet, som musen peger på. Hvis man trykker på museknappen begynder maskinen at køre fremad. I den første prototype blev maskinen flyttet med translateringer, hvilket ikke fungerede sammen med det fysiksystem, Unity har indbygget. Det blev ændret således, at maskinen fik tilføjet en kraft i dens fremadretning. Det gjorde derimod at maskinen beholdte en kraft i en retning, selvom den drejede, hvilket får den til at skride ud. Denne metode kunne vi ikke bruge, da vi gerne ville have en mere kampvognsagtig opførsel. Vi sørgede for, at al kørselsrelateret kraft maskinen fik var i fremadretningen. Når maskinen drejer, vil al dens momentum blive ændret i drejeretningen. Vi måtte begrænse den opdrift maskinen kunne få, så sammenstød med bosser og lignende ikke fik den til at flyve.

Vi testede gentagne gange spillet på børn fra det lokale fritidshjem, og om børnene kunne finde ud af at manøvrere rundt i spillet. Adspurgt havde de intet at klage over. Alligevel viste det sig, at når voksne spillede spillet, følte de ikke, at maskinen reagerede, som de forventede. Det skyldes nok dels at børn er hurtigere til at acceptere, at tingene er som de er, frem for at ville have ting til at opføre sig, som de er vant til. Derudover skyldes det nok at børnene gerne ville være gode til at hjælpe og gode til at spille spillet, så de gav ikke udtryk for deres problemer. Vi indså derfor, at vi måtte gøre så meget som muligt for at hjælpe spilleren. Det skulle være klarere hvilken vej maskinen

ville køre på et givent tidspunkt, og det skulle være nemmere og sjovere at køre.

Da vores styring minder meget om at have en snor bundet fast i maskinens forende, valgte vi at forsøge at genskabe denne følelse. Når vinklen mellem den nuværende og ønskede retning på maskinen forøgedes, blev drejehastigheden også forøget. Hvis man vil dreje maskinen 180 grader, bliver drejeradiusen lille, selvom man har fart på.

Det viste sig også at være et problem for voksne at se, hvilken del af maskinen der var forenden, på trods af at maskinen altid forsøger at rotere mod musen. Derfor implementerede vi to styringsplaner, som kan ses på figur 8.8, omkring maskinen, der skal fortælle spilleren, hvordan den vender, samt hvilken retning, den forsøger at dreje imod.

Figur 8.8: De to styringsplaner

Jeg lavede en del prototyper af våbene til maskinen. Det var den oprindelige plan, at der skulle laves fem våben til at starte med, og den overskydende tid i projektet skulle bruges på at lave flere, sjovere og skørere våben. Det viste sig dog, at der ikke var noget overskydende tid, så de fem oprindelige våben (gravearm, flammekaster, laser, sav og dynamitkaster) blev til de endelige våben. Flammekasteren og saven er lavet med usynlige kasser. Så længe en fjende opholder sig inde i kassen, tager den skade. Dynamitkasteren og gravearmen fungerer ved at give højere skade og kraft, jo tættere fjenderne er på nedslagspunktet. Dynamitstangen bliver kastet og eksploderer, når den rammer noget. Laseren rammer alt i en lige linje, men kan ikke skyde gennem vægge og har en begrænset rækkevidde.

Det blev besluttet af vores *lead*-programmør, at vi skulle indbygge et bibliotek til A*-*pathfinding* som vi fandt på internettet[5]. Implementeringen havde både A*-*pathfinding* samt et modul med et nemt brugerinterface til at generere et navigationsgitter. Desværre var det ikke helt så smertefrit at implementere dette som først antaget. Vi fik nogle ydelsesproblemer, da vi i spillets tredje bane har 50 små fjender, som jagter spilleren på en gang. Derfor måtte dele af koden omskrives for at bruge mindre ressourcer. Det var hovedsageligt selve styringen/flytningen af enheder, der tog mange kræfter. Det resulterede i, at vi var nødt til at udskifte Unitys indbyggede funktionalitet til

at flytte objekter. Det blev erstattet af en ren kraftbaseret udgave, der betød at vi mistede Unitys kollissionsdetektering, hvilket betød fjenderne kunne gå igennem terrænet. Derfor bliver de hele tiden flyttet, så de altid er ovenpå terrænet.

Jeg arbejdede en del sammen med holdets animator, da han skulle animere de små fjender. Da vi havde overvundet de første problemer med at få tingene til at spille sammen med Unity, var det ganske nemt at skifte mellem animationer og samtidig få det til at se glidende ud.

Problemer og udfordringer

Undervejs i projektet gennemgik vores styrringssystem flere iterationer. Vi havde mange problemer med at få maskinen til at opføre sig naturligt uden at introducere nye problemer: For eksempel ville vi gerne have maskinen til at tippe, hvis den holdt på en skrå flade, men den måtte samtidig ikke kunne vende på hovedet. Af den grund lavede vi en maksimal hældning, maskinen kunne have på et givent tidspunkt.

Vi havde også en del problemer med den feedback spilleren skulle have: Undervejs i udviklingen var det en meget tom fornemmelse at køre rundt og skyde ting. Derfor blev der gjort noget ud af at tilføje effekter til både våben og fjender. Laseren har for eksempel fået tilføjet rekyl, der rykker maskinen en smule når våbenet affyres. Fjenderne flyver ud af billedet med ild i halen, når de bliver slået ihjel. Bygningen hvor fjenderne kommer fra, skubber spillerens maskine væk, når den skyder.

KI i spillet

Vores små fjender bruger en simpel A*-*pathfinding* algoritme. Der bliver genereret et gitter, der viser de gangbare områder. Fjenderne finder derefter den korteste vej til spilleren og roterer, så de har fronten mod næste punkt på vejen, hvorefter de begynder at gå. Når en fjende kommer tæt nok på spilleren, stopper den og begynder at angribe spillerens maskine.

Vores små fjender er dem der skaber liv i banerne. Deres formål er til dels at jagte spilleren, så der kommer masser af bevægelse i spillet, og dels så spilleren får lov til at se sine våben udfolde sig. Hvis der havde været mere tid til projektet havde disse fjender været en oplagt mulighed at udvide. For eksempel kunne de have fået nye våben. De kunne også dø på forskellige måder, alt efter hvilket våben de var blevet skudt med. Derved kunne spilleren få en masse feedback fra spillet.

Oprindeligt var der lagt op til to typer fjender. Vi ville have implementeret nogle skydende versioner af de små fjender, sådan at de ikke alle blot løber hen i døden. På grund af tidsmangel blev disse fraskrevet.

Det er små fjender, som vil skabe grundlaget for afprøvningen af KI'en i spillet. Det er dem, som skaber mulighederne for både alternativt gameplay, samt guider spilleren hen mod hans mål: bygningerne, der *spawner* fjender.

Vores *boss*-KI bruger samme pathfinding som de små fjender. Desuden vurderer den, om den kan ramme med nogle af dens skydevåben på afstand. Hvis den ser, at den kan ramme med et skydevåben, roterer den således, at skydevåbenet peger i spillerens retning. Jo længere tid der bliver brugt på at skyde med et våben, jo lavere vægt får det. Derfor bliver bossen mere tilbøjelig til at skifte strategi. Det medfører, at han kører hen til spilleren og forsøger at bruge nærkampsvåben. Skydevåbenets vægt stiger kontinuerligt over tid, så hvis man kører væk fra bossen, skifter han taktik til igen at skyde efter en.

I det originale design var det meningen, at vi ville have implementeret "seek and flee"[7, afsnit 3.3.3]. Desuden skulle bossen kunne vurdere sin våbenstyrke i forhold til spilleren i henholdsvis nærkamp og på afstand. På grund af tidsmangel i projektet, blev disse ting dog skåret fra. Holdet vurderede, at der var et for lille udbytte i disse features, da spillet henvender sig til 6-7 årige børn.

Bossen har flere formål: Den skal være udfordringen, som spilleren skal overkomme, for at gøre sig fortjent til at spille næste bane. Da de små fjender dør ekstremt let, er det mere tilfredsstillende at få lov til at slå en lidt større maskine ihjel. Derudover giver bossen et nyt våben som belønning, hvis det lykkes spilleren at vinde kampen. Bossen skal også vise tips og tricks til at bruge våbene og være et eksempel på, hvordan man kan bygge en maskine.

Konklusion på produktet

Der er en masse ting i udviklingsprocessen, der kunne være gjort anderledes og bedre. For eksempel ville det have været godt med lidt mere planlægning i udviklingen blandt programmørerne. Der er også flere ting i spillet, som kunne være blevet forbedret. Med nogle ting, så som styringen, var vi hæmmet af de krav, der var blevet udstedt fra DADIUs side. Andre ting, såsom flere våben, bedre feedback og mere varieret gameplay gennem de forskellige baner, kunne være opnået ved at have lidt længere tid til at udvikle. I sidste ende synes jeg dog, at det er et meget fint produkt, vi har fået skabt på meget kort tid. Det har potentiale til at blive rigtig godt, hvis der udvikles videre på det. Vi fik også meget positiv respons på spillet til evaluering på DADIU og alle tre dommere så potentiale i spillet. Det blev enda nævnt, at vi skulle ændre aldersgruppen 30 år op, så censorerne også kunne få lov at spille spillet. Censorerne foreslog også, at spillet blev omdannet til et multiplayer-spil, eventuelt i form af et MMO[3]-spil.

[3]Massively Multiplayer Online

8.3 Sophie's Dream Leap

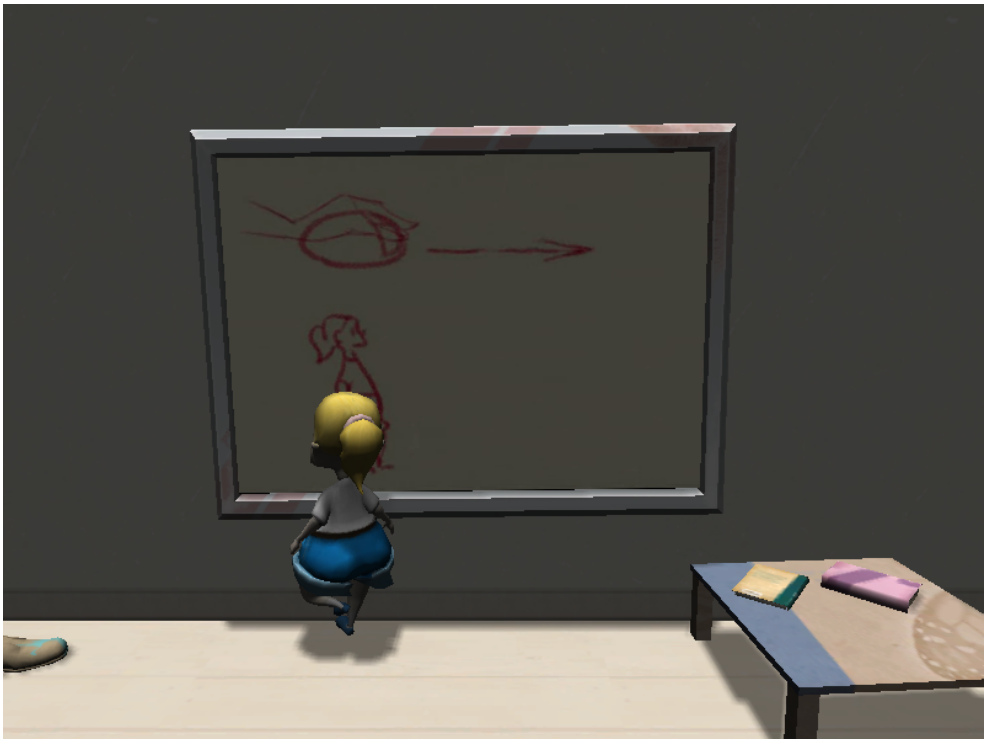
Spillet *Sophie's Dream Leap* er et 3D-platform spil. Spilleren indtager rollen som den seks-årige pige Sophie. Sophie er, bortset fra tilstedeværelsen af hendes spøgelsesbror, alene hjemme. Spillet starter med at Sophie følger efter broderen op på spisestuebordet, hvorefter jorden bliver til ætsende syre.

Instruktøren i gruppen ville lave et spil, der efterlod spilleren med en oplevelse. Han ville undgå at vi lavede et spil, hvor det primære formål for eksempel var at samle mønter op eller få den hurtigste tid. Desuden ville instruktøren gerne have et 'sprogløst' spil, så vi undgik tekst og tale. For eksempel startes spillet ved at trykke på Sophie i menuen (figur 8.9). Lyden fjernes fra spillet ved at trykke på den hoppende radio. I *tutorial*-banen vises styringsmetoden derfor også med video i stedet for ved hjælp af tale eller tekst (figur 8.10).

Figur 8.9: Menuen i Sophie's Dream Leap

Forløb

Idéen til spillet blev fundet ved hjælp af kollektiv brainstorming over den første halvanden dag i forløbet. Vi havde tre primære idéer, hvoraf 'Jorden



Figur 8.10: Gennemgang af styremåde

er giftig'-temaet blev valgt som vores foretrukne på den anden dag af produktionen.

Efterfølgende udviklede vi en række forskellige prototyper, for at få fastlagt, hvordan spillet skulle styres. Den ene var i isometrisk perspektiv, hvor man trykkede et sted og spilfiguren efterfølgende gik derhen. En anden var tredjepersonsperspektiv, hvor man bevægede sig ved at holde venstre musetast nede og hoppede ved dobbeltklik. Ingen af disse fungerede dog særlig godt, og derfor besluttede vi os for en helt tredje styringsprototype. I denne bevæger spilfiguren sig altid i musens retning. Hastigheden hvormed figuren bevæger sig er afhængig af afstanden til musen. Trykker man på venstre musetast hopper figuren. Hvis musetasten bliver holdt inde under hoppet, hopper man højere.

Udviklingsmetode

Vores projektleder ville gerne have, at vi benyttede os af *SCRUM*-metoden i vores udvikling. Det vil sige, at vi hver morgen startede med et stående møde, hvor alle på holdet sagde, hvad de havde lavet dagen før, og hvad de skulle lave den dag. På den måde havde alle et indblik i, hvordan udviklingen af spillet gik.

Endvidere blev hele forløbet inddelt i en række *sprints* af tre eller fire dage. Hver deltager vurderede så i starten af en *sprint*, hvor meget arbejde de regnede med at kunne få fra hånden inden for perioden. Det medførte, at alle som regel vidste, hvad de skulle tage sig til, samt at man ikke blev udsat for en uoverkommelig opgave.

Det viste sig at *SCRUM* medførte den ulempe, at vores kildekode blev meget rodet med tiden. Der blev tilføjet eller ændret *features* i hver *sprint*, som blot byggede oven på gammel funktionalitet. Problemet blev yderligere forværret af, at vi ikke havde tid nok i forløbet til at sætte os ned og rydde op. Fokus lå på at udvide og forbedre spillet, hvilket desværre blev sværere og sværere på grund af kodens lave kvalitet. Når kildekoden blev mere kringlet medførte det også, at det stort set kun var den oprindelige programmør, der til slut kunne gennemskue, hvor ændringer skulle foretages. Det gjorde, at de forskellige programmører fik meget skarpt opdelte roller: lyd, animation, kunstig intelligens, og så videre.

Rolle i produktionen

Min rolle i produktionen var blandet. Det primære fokus lå på at implementere den kunstige intelligens, der skulle drive broderen i spillet. På produktionen var der dog reelt kun tre fuldtidsprogrammører, hvilket betød at jeg var nødt til også at bruge en del tid på forefaldende programmerings- og implementeringsopgaver. Blandt disse kan nævnes:

- *Debugging* og arbejde med vores animationssystem
- Websiden `http://www.dream-leap.com`
- Funktionalitet til vores lyddesigner
- Komprimering og indsætning af *assets* i vores spil
- Finpudsning af baner i forhold til den kunstige intelligens' opførsel
- Funktionalitet til rulletekster

Min arbejdsindsats i de forskellige områder har været dikteret af, hvad der havde højest prioritet i den igangværende *sprint*.

Afprøvning af spillet

Én gang om ugen blev vores spil afprøvet på en gruppe bestående af et par børn i 6-7-års alderen. Børnenes færd i spillet blev optaget[4], samtidig med at vores *game designer* og *project manager* observerede. Disse observationer var

[4]Det vil sige børnenes musebevægelser og vej igennem spillet, men altså ikke børnenes reaktioner og lignende

grundlaget for, hvilke ændringer der blev foretaget i banedesign såvel som styremetode.

Desuden brugte vi programmører en del tid på at spilteste undervejs i forløbet. Dette var dog mest for at finde fejl, mangler og mærkelige ting, idet vi allerede efter kort tid var blevet så vant til vores spil, at det var alt for let for os, og vi dermed ikke kunne bruges til at finpudse sværhedsgrad og lignende.

Kunstig intelligens

Det blev hurtigt i forløbet besluttet, at vi kunne udnytte DADIUs krav om kunstig intelligens i spillet på en god måde. Vi ville derfor lave en bror, der kunne fungere som vejleder i spillet. Broderen skulle gå foran spilleren og vise en vej igennem banerne. Samtidig skulle broderen fungere som gemme-punkt visse steder. Den kunstige intelligens i *Sophie's Dream Leap* består derfor primært af en implementering af *pathfinding*-algoritmen *A**. Algoritmen er dog modificeret til også at tillade hop rundt i banen, så broderen ikke behøvede en fast vej fra start til slut.

På grund af den valgte algoritme, blev der sat nogle begrænsninger for de baner, vi kunne have i spillet: Vi tillod ikke gangbare overflader, der lå lige over hinanden. Det vil sige for ethvert (x, z)-koordinat, måtte der kun være et gangbart punkt. Var der alligevel flere, ville den kunstige intelligens ikke være opmærksom på andre overflader end den, der stod øverst i Unitys scenehierarki.

For at den kunstige intelligens ikke skulle tage meget svære eller obskure veje igennem banerne, blev hop gjort dyre i algoritmens vægtfunktion.

Idet miljøerne, spilleren bevæger sig rundt i, er dynamiske (objekter kan flyttes, væltes, destrueres og lignende), implementerede vi også funktionalitet i den kunstige intelligens, så den blev opmærksom på sine omgivelser. Hvis den kunstige intelligens stødte på en forhindring på sin beregnede vej, blev der beregnet en ny vej til det ønskede mål.

Den kunstige intelligens brugte et ret fintmasket koordinatsystem til at indele banen i. Derfor fik vi problemer med hakkende bevægelser, hvis hele stien frem til mål ikke var kendt på forhånd. For at løse dem, lod vi den kunstige intelligens undersøge felterne, den ville inddrage i sin sti. Hvis feltet var optaget, blev det med det samme smidt væk. Det betød at vi mistede den meget menneskelige egenskab, at KI'en kunne gå forkerte veje i første omgang. Til gengæld undgik vi, at skulle bruge en masse tid på at udglatte bevægelserne langs vejen for at undgå de tidligere beskrevne hak.

Fordi vores baner var så forholdsvis små, var det ikke synligt på spillets ydelse, når den kunstige intelligens beregnede sine veje.

For dels at hjælpe den kunstige intelligens igennem svære områder, og dels at kunne forbedre gameplayet, implementerede vi såkaldte *triggers*. Disse kunne blive aktiveret, når enten spilleren eller den kunstige intelligens passerede igennem, eller hvis de begge var til stede. Det brugte vi til at aktivere

den kunstige intelligens, når spilleren havde bevæget sig lidt ind i en bane. Vi brugte det også til at få den kunstige intelligens til at agere gemmepunkt og vente på spilleren, indtil hun nåede frem.

For at gøre den kunstige intelligens mere resistent over for fejlberegninger, uventede forhindringer og lignende, teleporterede vi KI'en til det næste punkt på sin rute, hvis den sad fast eller kom for langt væk. Det gjorde, at vi kunne tillade os at have nogle upolerede områder, hvor den kunstige intelligens ikke opførte sig optimalt. Endvidere blev den kunstige intelligens portrætteret som et spøgelse, hvilket tillod os lidt unormal opførsel, uden at det blev akavet.

Idet spillerens hop og generelle styring har meget lidt med virkelighedens fysik at gøre, besluttede vi os, efter mange problemer, for også at snyde med den kunstige intelligens' bevægelse. Hop bliver derfor lavet helt uden om det indbyggede fysiksystem. Den kunstige intelligens bliver blot flyttet langs en *bezier*-kurve som er lavet mellem start- og slut-punktet for hoppet.

Havde vi haft mere tid ville vi gerne have implementeret noget *path smoothing* [7, side 251 og 253], så broderen gik mere flydende og uden hak når han skiftede retning. Broderen, der hopper igennem banen, kan ses på figur 8.11.

Spillets tredje bane var desuden meget virkelighedsfjern, så der besluttede vi, at leveldesigneren skulle have frie tøjler. For at der overhovedet ikke skulle tages højde for KI'ens gang, udnyttede vi at han var et spøgelse: Vi lod ham langsomt *fade* ud, når spilleren havde indhentet ham og gemt spillet, og efterfølgende blev han med det samme flyttet til næste *waypoint*.

Udover den primære kunstige intelligens i form af broderen, blev der også udviklet fjender, der skulle stå i vejen for spillerens fremgang (figur 8.12). Fjenderne vandrer rundt på en platform og vender om i en vilkårlig retning, hvis de finder ud, af at der er en kant foran dem. Når fjenderne ser spilleren, begynder de at opbygge vrede, hvorefter de løber imod spilleren og forsøger at skubbe hende ud over kanten.

Vi nåede desværre ikke at indbygge funktionalitet i fjenderne, så de elegant kunne undvige kanter. Vi ville også gerne have inkluderet mulighed for, at fjenderne forsøgte at forudsige, hvor spilleren ville være lidt ude i fremtiden og så sigte efter det punkt i deres fremløb.

Problemer og udfordringer undervejs

Vi havde en række problemer og udfordringer undervejs i produktionen.

For eksempel stødte vi ind i problemer med Unitys lyssætning, skygger, fysiksystem, lyd og lignende. De var forårsaget af, at vi ikke benyttede en normal skala i vores spil. Spisebordet i vores første bane var i Unity for eksempel ti meter bredt og ti meter langt, og vores hovedpersoner er omkring tre og seks meter høje. Det gjorde at beregningen af skygger, lyd og lys blev forskruede, idet Unity troede vores baner var langt større end de skulle forestille at være.

Et af vores største problemer var imidlertid, at vi allerede efter to dage mistede en programmør. Desuden var vores *lead*-programmør nødsaget til at

Figur 8.11: Jorden er blevet giftig. KI'en (øvre venstre hjørne) hopper igennem banen.

følge kurser og tage til eksamener på sit studie. Det resulterede i, at vi blot var tre programmører til stede det meste af tiden.

I begyndelsen af processen valgte vi at benytte os af *Locomotion System*-biblioteket [6] til Unity for at få realistiske bevægelser for vores figurer i spillet. Det medførte dog en række problemer, idet der var krav til animationerne og programkoden vi kørte for at få systemet til at fungere korrekt. Det betød, at vi brugte en masse tid i løbet af projektet på at finde og rette fejl og implementere funktionalitet, der tillod os at benytte *Locomotion*-systemet. Selvom det krævede meget arbejde, var slutresultatet det værd, idet vores ganganimationer blev meget realistiske. Systemet tillod også at vores figurer bevægede sig med variable hastigheder uden at animationerne så underlige ud, eller at benene "skøjtede" på underlaget.

Baseret på resultaterne af vores brugertest brugte vi også utrolig lang tid på at forbedre, udvide og ændre vores system til styring af spilleren. Skulle jeg lave noget om, ville det være at afholde flere brugertest, især i starten af forløbet. Dermed tror jeg vi tidligere kunne have fundet flere fejl og mangler ved styremåden. Havde vi gjort det, kunne vi forhåbentlig have fundet de rigtige parametre langt tidligere. Dermed kunne programkoden være blevet "låst" og programmørtiden i slutningen af forløbet brugt på andre, mere presserende,

Figur 8.12: Fjenderne i Sophie's Dream Leap

ting. Derudover tror jeg også, det ville have været en fordel, hvis der var en brugertest efter hver *sprint* og ikke blot efter hver anden *sprint*. Jeg tror også, at vi kunne have haft gavn af at have en programmør med ude til testene, da vi var mere bekendte med ømme punkter i spillet.

Naturligvis voldte den kunstige intelligens også problemer undervejs. Mange af problemerne skyldtes, at instruktørens og *game designerens* vision for den kunstige intelligens var enormt ambitiøse. Desuden ændredes planerne for den kunstige intelligens undervejs, hvilket var et problem, idet udviklingen så skulle skifte kurs. Havde målet været klart (og endeligt) tidligere i processen, ville vi formentlig kunne have lavet en bedre løsning, da vi ikke havde behøvet at fokusere på generelle tilfælde, men kun på vores specifikke scenarier.

DADIU evaluering

9

Først og fremmest er DADIU et mægtigt godt initiativ. Det giver os, der ønsker at lave vores egne computerspil en mulighed for at afprøve processen under nogle mere virksomhedsrealistiske forhold.

Det har været en fornøjelse at arbejde med folk, der har specialiseret sig, og som hver kan bidrage med en ekspertise til projektet. Man kan dog påklage at ikke alle, der er på projektet, er der af fri vilje. Instruktørerne fra filmskolen har for eksempel ikke selv valgt at deltage i netop en computerspilsproduktion. Det er en del af deres uddannelse, og de er derfor tvunget til at deltage. For nogle af grupperne har dette betydet at deres instruktør ikke har været opsat på at lave et godt computerspil, men blot udfolde deres talent eller egenskaber til at lave film. Instruktøren på både Imachination og Sophie's Dream Leap var heldigvis både enormt gode til at styre holdet, men derudover også opsat på at lave et godt produkt, selvom deres kendskab til computerspil var begrænset. For eksempel var instruktøren på Imachination enormt god til at lytte til folk, der rent faktisk vidste noget om enten computerspil, eller det rent tekniske i det der skulle løses. Han traf sine beslutninger efter at have fået input fra disse personer, fremfor blot at beslutte det han umiddelbart synes var bedst.

Gamedesignerens rolle virkede en smule overflødig. Det virkede som om de kunne en masse teori om *gamedesign*, men manglede praktisk erfaring så de rent faktisk kunne bidrage med noget til spillet. Selvfølgelig er DADIU til for at give praktisk erfaring, men det virkede ikke som om de kunne noget, en erfaren computerspiller ikke kunne have gjort lige så godt. På Imachination produktionen var en af de få opgaver *gamedesigneren* havde at balancere våbenes skade samt fjenders, bygningers og spillerens liv. Selv denne opgave er ikke løst særligt godt, da nogle våben er langt bedre end andre. *Gamedesigneren* endte med at bruge det meste af sin tid på leveldesign, dels fordi der ikke var andre til det, og dels fordi han ikke selv havde noget at lave. Så så vi hellere, at der var en fra arkitektskolen på projektet som leveldesigner, og vi derved kunne få lavet nogle rigtig fede baner. I Sophie's Dream Leap ville

gamedesigneren utrolig gerne fungere som programmør og gøre det hele selv, hvilket resulterede i en masse tidsspilde.

Et andet problem ved DADIU er frafald og udskiftning. Det er meningen, at de hold der var til produktionerne skal lave en ny produktion næste år. Den nye produktion skal fungere som en afgangsprøve, hvor man får et diplom som bevis på sin deltagelse i DADIU forløbet. Inden der er gået et år, er der naturligvis nogen som falder fra af forskellige grunde, og det er der intet at gøre ved. Et problem der kan gøres noget ved, er at deltagere fra designskolen kun er en del af én produktion. Hvis denne produktion er afgangsproduktionen, mangler folkene fra designskolen erfaring, som de resterende medlemmer har.

Et tredje problem er, at der mangler feedback til programmørernes arbejde under udviklingen. Konsultationsgruppen kommer med mange kommentarer til indholdet i spillet, det visuelle og lyden når de får demonstreret spillet. Som programmør bliver det antaget, at du kan dine ting, og hvis der er problemer i spillet, forventes det, at du kan fikse det. Der er ingen kontrol, sparring eller feedback på, hvordan koden er stykket sammen, eller eventuelle designvalg som er taget. Det er besynderligt, at der er så lille kontrol eller fokus på programmørerne, da store dele af projektet enten brister eller bærer på baggrund af deres arbejde.

Vi oplevede på samtlige produktioner, at meget af programmørernes tid går spildt, fordi de bliver det centrale punkt i implementeringen i Unity. Dette skyldes, at programmørerne bliver *nødt* til at bruge Unity og dermed hurtigt får det bedste kendskab til miljøet. Den spildte tid kunne for eksempel være blevet brugt bedre på at udvikle flere *features*. Det er en del af vejen projektlederens ansvar at beskytte programmørerne fra denne arbejdsbyrde, men det værste af problemet kunne være afhjulpet ved at lære folk, hvordan man bruger Unity. Derfor ville det være en stor forbedring, hvis der i det indledende forløb var tvungen Unity-undervisning.

9.1 Evaluering på ITU

Da produktionsforløbet var forbi, havde vi en dag, hvor alle vores produkter blev evalueret af tre dommere:

- David Mariner, Head of Games, NDS
- Julie Heyde, Marketing expert, Nordic Game Production (NOGAP)
- Jonas Raagaard, `http://spiludvikling.dk` og 3D Artist, Massive Entertainment.

Hver gruppe blev kaldt op en efter en, hvorefter nogle fra holdet skulle *pitche* spillet, fortælle om udviklingsforløbet og give et bud på, hvordan spillet

kunne blive publiceret. Herefter kom dommerne med kommentarer, hvorefter holdet kunne svare igen. Vores tre spil fik generelt meget ros, og dommerne havde kun få, hvis nogen, bekymringer om spillene. Catnapped blev dog kritiseret for at være alt for svært.

Til slut på dagen var der en programmør-evaluering, der dog viste sig blot at være et sted, hvor man kunne komme ud med sine frustrationer over de andre produktionsmedlemmer. Også her manglede der altså noget relevant evaluering af programmørernes arbejde.

Unity evaluering

10

Unity er en udviklingsplatform til spil. Det kan lave eksekverbare filer til Windows, Mac, Wii, iPhone og iPad. Desuden kan man vha. en *plugin* spille Unity-spil i en internetbrowser.

10.1 Fordele ved Unity

En af fordelene ved Unity er den hastighed, hvormed man kan få sit basale spil udviklet. På grund af de mange indbyggede *features* kan man hurtigt lave spilbare ting, uden at skulle starte fra bunden. Desuden er der et godt 'drag-and-drop'-interface, der nemt lader en give objekter i spillet forskellige egenskaber, materialer og opførsler.

Unity tillader udvikling af stort set alle typer spil. Selvom andre *engines* måske er bedre til en specifik type, klarer Unity sig overordnet set ret godt i alle typer. Derved er det ideelt til udviklere, der laver mindre spil, idet man ikke behøver skifte *engine* når man skifter genre.

Unity er let at sætte sig ind i. Selvom ingen af os havde beskæftiget os med Unity før DADIU-produktionen, blev vi hurtigt vant til det, og kunne hurtigt få en del arbejde fra hånden.

Når man udvikler i Unity, får man mange ting foræret. Disse er for eksempel lys, terræn, diverse standardiserede styremåder og et fysiksystem. Fysiksystemet tager sig af ting som tyngdekraft, friktion og kollision. Desuden kan man påføre kræfter til objekter ved hjælp af *scripts*.

Scripts i Unity kan skrives i tre forskellige sprog: Javascript, C# og 'Boo'. Javascript har den fordel, at det er dynamisk typet. Til gengæld er det væsentligt langsommere end C#. Ingen af os har nogen erfaring med 'Boo'. På alle tre spilproduktioner benyttede vi os af C#.

Hvis man har specielle behov i Unity-*editoren*, kan man let udvide den ved hjælp af *scripts*. Unity-*editoren* er primært produceret vha. Unity selv, så det er meget få ting, man ikke har mulighed for at tilgå eller ændre. Dette kan for eksempel bruges til specielle animations- eller lydsystemer, hvor man udvider den basale funktionalitet, der er til stede i Unity.

10.2 Ulemper ved Unity

Der er også fejl og mangler ved Unity. Størstedelen resulterer heldigvis kun i irritation og ikke blokerende fejl.

Et af problemerne er, at banerne i Unity gemmes i et binært filformat. Det betyder, at versionskontrol ikke kan finde ud af at flette to versioner sammen. Det vil sige, at der kun er én person, der kan arbejde i en scene på et givent tidspunkt, og det skaber en flaskehals i udviklingsprocessen. Dette ser heldigvis ud til at blive løst i næste udgave af Unity, hvor de skifter til at gemme scenerne i et tekstformat.

Interfacet i Unity er desuden ikke helt gennemarbejdet. For Eksempel er der ved alle fejlmeddelelser (selv de ubetydelige) en dialogboks midt på skærmen, der spørger om man vil *Try Again*, *Cancel* eller *Force Quit*. Det kan resultere i meget tabt arbejde, hvis man ved et uheld rammer *Force Quit* enten med musen, eller fordi man er ved at skrive på tastaturet.

I Unity kan man tilføje *layers* og *tags*, der kan bruges til at angive egenskaber til objekter. På trods af navnene, kan man dog kun tilføje ét *layer* og ét *tag* til hvert element. Det er et problem, hvis objekter har flere egenskaber, man gerne vil *tagge*. Løsningen bliver, at man tilføjer tomme *scripts* og lader dem agere *tags*.

Unity lægger meget op til brug af tråde i form af *coroutines*. Problemet er, at Unity ikke er trådsikkert, så der er noget funktionalitet, man må undlade at bruge, hvis man vælger at bruge tråde.

Sammenligning af vores KI'er 11

Alle tre spil anvender simpel *pathfinding*. Derved minder de meget om hinanden, men de udfylder tre helt forskellige roller.

I Catnapped optræder KI'en i en rolle som direkte forhindring. Dens eneste formål er, at gøre det sværere for spilleren at bevæge sig rundt i gangene, og spillerens tvinges til at time sine bevægelser for at pasere katteøjnene.

I Imachination er KI'en flere ting. Den er til for at tvinge spilleren til at bevæge sig rundt i banen og ikke blot stille sig et sted og skyde efter bygningerne. Derudover fungerer den som et vink til spilleren om, at bygningerne skal bekæmpes. Det skyldes, at bygningerne bliver ved med at lave fjender, indtil de er blevet slået ihjel. Ønsker spilleren derfor at stoppe den uendelige strøm af modstandere, må han bekæmpe bygningerne. Til sidst udfylder KI'en en ren tidsfordrivsrolle. Nogle finder det sjovt blot at køre rundt og skyde de små modstandere med deres forskellige våben.

I Sophie's Dream Leap fungerer KI'en som en guide. Dens rolle er at vise vejen gennem banen. Spillet bliver også gemt, når man indhenter KI'en på bestemte steder i banen. Det er ved ens seneste gemmepunkt, at man bliver genoplivet, hvis man dør. Dermed arbejder den for at gøre spillet nemmere, i modsætning til de to andre spil.

Hvad er det så, der er fælles for vores tre kunstige intelligenser? I alle spillene har de indflydelse på spillerens progression gennem spillet. Hvad end det er hjælp eller forhindring, eller om det skaber alternative legemuligheder, påvirker de alle både, hvor meget tid spilleren bruger i spillet, men også udfordringsniveauet.

Afprøvning 12

For at efterprøve vores hypotese, aflagde vi et besøg hos Bryggehuset Fritidshjem, hvor vi afprøvede vores tre spil på en række børn.

I de følgende afsnit vil vi angive testpersonerne med et tal, der angiver hvilket nummer de havde i vores test, samt et D eller P, der angiver henholdsvis dreng/pige.

12.1 Forventninger ved brugertest

Som resultatet af vores brugertest forventer vi at kunne se klare ændringer på de heatmaps fra vores spil henholdsvis med og uden KI.

I Catnapped forventer vi, at spilleren kommer til at bevæge sig hurtigere rundt i labyrinten. Om det vil føre til, at de hurtigere kommer ud af den, er ikke til at sige. Børnenes valg bliver muligvis mindre velovervejede, da der ikke længere er fare ved at bevæge sig rundt.

I Imachination forventer vi, at banerne bliver klaret hurtigere. Der er ikke længere et krav om at bevæge sig for at holde sig i live. Det er muligt blot at skyde bygningerne på afstand uden den store udfordring. Spilleren vil hovedsageligt opholde sig helt tæt på bygningerne, da de ikke bliver tvunget til at køre rundt i resten af banen. Der er heller ikke længere mulighed for, at skyde ting uden at komme tættere på at gennemføre banen. Desuden er der ikke nogen mulighed for at børnene fokuserer på blot at dræbe de små fjender.

I Sophie's Dream Leap forventer vi, at det tager længere tid at komme gennem banerne. Vi forventer også at spilleren vil dø mere, da de formentlig vil forsøge at lave sværere spring. Der vil være mere udforskning, da der ikke længere er en guide til at pege dem i en bestemt retning. Derfor vil vejene, som spillerne tager, afvige meget.

Vi forventer at se en betydelig forskel på de børn, der spiller med KI først, og dem der spiller uden KI først. For eksempel er Sophie's Dream Leap en del nemmere at spille uden KI, hvis først børnene har spillet det med, da man kender vejen rundt i banen.

Barn	Spil 1	Spil 2	Spil 3	Spil 4	Spil 5	Spil 6
1	i-	c	s-	i	c-	s
2	i	s-	c-	i-	s	c
3	c-	i	s	c	i-	s-
4	c	s	i-	c-	s-	i
5	s-	i-	c	s	i	c-
6	s	c-	i	s-	c	i-
i, *s* og *c* angiver henholdsvis Imachination, Sophie's Dream Leap og Catnapped. - betyder at spillet spilles uden KI						

Tabel 12.1: Oprindelig testplan

12.2 Testplan

For at få de bedst mulige resultater ud af vores test, var vi nødt til at lægge lidt tanker bag den måde og rækkefølge, vi ville teste vores spil på.

Vi skulle teste to udgaver af hvert spil (én uden kunstig intelligens og én med), og vi ville gerne se, om der var en forskel i måden, spillet blev spillet på. Derfor skulle vi både have børn, der spillede udgaven *med* kunstig intelligens først, og børn der spillede udgaven *uden* kunsting intelligens først.

Fordi styringen i de tre spil er i samme kategori, kan det også have betydning, hvilken rækkefølge spillene bliver spillet i. Derfor skal alle spil spilles både som nummer ét, to eller tre.

Da der er tre spil, og hvert spil er i to versioner, skulle testen udføres med minimum seks børn. Seks børn gav mulighed for at teste hvert spil som 'første spil' i begge udgaver. Desuden vil vi også have testet alle udgaver af alle spil én gang som 'andet spil' og én gang som 'tredje spil'. Vi vurderede at seks resultater for hvert spil ville give os nok data til at finde en sammenhæng.

Vi valgte at variere om børnene spillede et spil med eller uden KI først, da deres spilresultater unægteligt ville blive påvirket hvis de før har spillet spillet igennem.

For at børnene ikke skulle spille den samme bane to gange i træk, valgte vi at skifte mellem spillene efter hver bane.

Vi valgte på baggrund af ovenstående, at producere den testplan, der ses på tabel 12.1.

Da børnene skulle spille alene og uden hjælp fra andre, valgte vi kun at have de børn, der spillede spillet i rummet med computerne. Børn er meget sociale, når de spiller computer, og vi forventede, at dem der så på ville have en tendens til at komme med forslag til, hvad spilleren skulle gøre.

Efter test med de første to børn måtte vi indse, at der var en række fejl i vores testplan. Vi havde simpelthen ikke taget højde for, hvordan test med

børn adskiller sig fra test med voksne. Vi havde i høj grad fokuseret på at få de bedste resultater ud af testen, frem for at sikre os at vi *fik* resultater.

Vi erfarede at børnene gav op, hvis spillet var for svært, og de derfor slet ikke ville gennemføre resten af testen. Det gjorde at vi, som det kan ses af resultatet af D1's test (tabel 12.2), kunne ende med at mangle data, som barnet muligvis ville have kunnet give os. I 1D's tilfælde nåede han til Catnapped anden gang og besluttede sig for, at han ikke ville spille mere. Det er meget muligt, at det er Catnapped, der fik ham til at give op. Havde han spillet er andet spil i stedet, ville han muligvis have fortsat.

Et andet problem var, at nogle af børnenes koncentrationsevne var nedsat, da de var nervøse for at gå glip af, hvad deres venner lavede udenfor.

For at løse disse problemer valgte vi at ændre vores fremgangsmetode. Vi ændrede rækkefølgen børnene spillede spillene i så de spillede hvert enkelt spil med og uden KI lige efter hinanden. Derved fik vi den data vi skulle sammenligne for hvert spil så hurtigt som muligt. Skulle barnet opgive at spille videre midtvejs gennem testen, ville vi have et fuldt sæt og et halvt sæt data, istedet for tre halve sæt.

For at afhjælpe problemet med koncentrationsevnen valgte vi at have flere børn i rummet af gangen. Der var stadig kun ét barn der spillede, og med hjælp fra en pædagog lykkedes det os at holde de andre i ro uden at forstyrre. Blot det at de ikke var alene i rummet gjorde, at børnene fik langt større tålmodighed til at spille spillene igennem.

Vi forsøgte at gøre vores brugertest så uformel som muligt. Som vi tidligere har omtalt så er børn meget opsatte på at hjælpe til, hvis man spørger dem om det. Vi ville gerne have at børnene spillede, som hvis de selv sad alene derhjemme og hyggede sig med spillene. Derfor skulle vi undgå, at de ændrede deres spillemønster for at tilfredsstille os. Vi gjorde derfor meget lidt ud af at fortælle dem, hvad det var de medvirkede i og spurgte dem blot, om de ville spille computerspil.

12.3 Testresultater

Som tidligere nævnt, måtte vi ændre vores testplan under testen for at sikre os at få noget data, vi kunne bruge. Den faktiskte testplan ses i tabel 12.2.

Der var en del af børnene, der ikke gennemførte alle spillene eller som besluttede sig for, at de ikke ville spille mere midt i forløbet. Derfor valgte vi at teste med ti børn i stedet for de oprindelige seks planlagte.

I de følgende afsnit vil vi nøjes med at vise nogle eksempler på heatmaps. Alle vores genererede heatmaps kan ses i bilag C. Alt indhold i tabeller, der viser gennemførselstider, er angivet i sekunder.

Barn	Spil 1	Spil 2	Spil 3	Spil 4	Spil 5	Spil 6
1D	i-	cx	s-	ix	c-#	s#
2P	i	s-x	c-	i-	s#	c#
3D	c-	c	i	i-	s#	s-#
4P	s	s-x	ix	i-x	cx	c-#
5D	s	s-	i-	i	c	c-
6D	i-	i	cx	c-x	s-	s
7D	s-	s	i-	i#	c-#	c#
8D	c	c-	s-	s	i-	i
9D	s	s-	i-	i	c	c-
10D	s	s-	c-#	c#	i#	i-#

i, *s* og *c* angiver henholdsvis Imachination, Sophie's Dream Leap og Catnapped.
- betyder, at spillet er spillet uden KI
Et *x* angiver at barnet gav op undervejs i spillet.
angiver at barnet ikke har spillet.

Tabel 12.2: Revideret testplan

Barn	1D	2P	3D	4P	5D	6D	7D	8D	9D	10D
Med KI	162*	-	221	112*	272	167*	-	589	679	-
Uden KI	-	205*	462	-	137	226*	-	63	112	-

* betyder, at barnet ikke har gennemført banen
- betyder, at barnet ikke har spillet banen

Tabel 12.3: Gennemførselstider for Catnapped

Catnapped

I tabel 12.3 ses tiden børnene brugte på at spille banen henholdsvis med og uden kunstig intelligens.

1D Efter at være død fire forskellige steder til tre forskellige sæt øjne, gav han op. Desværre gav han så meget op at han ikke ville prøve spillet uden KI.

2P Efter hun var gået rundt i banen uden KI et stykke tid, uden at kunne finde vej, gav hun op.

3D Han var gået rundt i banen længe uden KI først. Han prøvede at finde huller i banen, så han kunne kigge ud af den. Samtidig fulgte han efter kakerlakker, når han mødte dem. Han brugte banen til at lege i, før han gjorde det, der var meningen: At finde ud af labyrinten. Med KI inkluderet gik han

mere målrettet efter at komme ud. Han døde tre gange undervejs, men fandt til sidst udgangen.

4P Hun gik kort rundt i starten af labyrinten, så døde hun en gang og gad ikke mere. Vi kunne ikke få hende overtalt til at prøve banen uden KI.

5D Denne dreng klarede det ret godt, han døde kun tre gange til at begynde med i labyrinten. Derefter fandt han rimelig hurtigt ned til den næste del af banen og fandt selv ud. Da han bagefter prøvede det uden KI, blev han lidt desorienteret af at øjnene ikke var der, han havde svært ved at finde den samme vej igennem banen.

6D Han gav op efter at være død to gange til øjnene da KI'en var på, men han havde mod på at prøve det uden KI. Denne udgave tog et stykke tid, men han kom til sidst igennem.

8D Den første der virkelig havde det hårdt med at komme igennem banen var denne dreng. Men han var meget standhaftig, så han klarede det med KI til sidst, efter 589 sekunder og at starte forfra ni gange. Da han skulle prøve banen bagefter uden KI kunne han genkende det hele og løb næsten igennem så hurtigt, det er muligt.

9D Den sidste dreng der prøvede labyrinten var også meget standhaftig. Han døde syv, gange før han rigtig kom gennem den første halvdel af labyrinten. Da han endelig kom videre, gennemførte han ret hurtigt banen. Det tog ham ikke længe at finde vej, da han skulle finde målet uden KI. Han kom i problemer, da han skulle forbi en masse æsker, der lå på jorden.

Overordnet Generelt har børnene gennemført labyrinten hurtigere uden den kunstige intelligens. Nogle har haft sværere ved at finde vej da den ikke var indkluderet og nogle fik alternative progressionsveje. Eksempelvis ham der var ligeglad med at komme igennem banen: Han ville udforske og se på kakerlakker eller prøve at finde fejl, så han kunne komme ud af banen på uautoriseret vis.

Imachination

På figur 12.4 ses den tid, børnene brugte på at spille banen henholdsvis med og uden kunstig intelligens.

1D Testen med KI blev opgivet halvvejs da han døde og mistedet modet. Der ses dog stadig forskel på heatmappet med og uden KI. Han har brugt mere af banen i versionen med KI.

Barn	1D	2P	3D	4P	5D	6D	7D	8D	9D	10D
Med KI	303*	317	319	98*	93	238	-	167	342	-
Uden KI	233	238	124	179*	124	112	136	167	87	-
* betyder, at barnet ikke har gennemført banen - betyder, at barnet ikke har spillet banen										

Tabel 12.4: Gennemførselstider for Imachination

2P Der ses tydeligt forskel på heatmappet med og uden KI. Der er meget lidt bevægelse i heatmappet uden KI, mens heatmappet med udnytter næsten hele banen.

3D I udgaven uden KI er utroligt lidt af banen udnyttet, og der bliver kørt meget målrettet og præcist. I udgaven med KI er der lidt mere bevægelse i banen, men stadig ikke hele banen bliver brugt.

4P Havde meget svært ved spillet. I testen uden KI kunne hun ikke finde ud af, hvad målet var og kørte lidt forvirret rundt. Derfor ser vi en større spredning i bevægelserne end i testen med KI. Gav op næsten med det samme med KI.

5D Der ses en smule mere bevægelse i testen uden KI end i testen med. Det skyldes nok mest, at han spillede uden KI først og derfor har godt styr på, hvad målet var i gennemløbet med KI.

6D Meget lidt bevægelse i begge gennemløb.

7D Meget målrettet i testen uden KI. Blev hentet af sin mor, inden han fik spillet med KI.

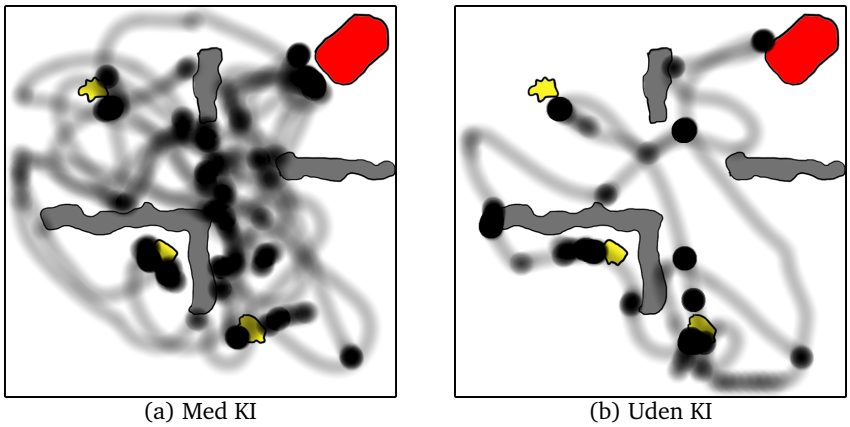
8D Der ses ikke en bemærkelsesværdig stor forskel mellem udgaven med og udgaven uden KI.

9D Stor forskel på med og uden KI. Langt mere bevægelse i testen med KI, mens testen uden er tæt på den minimale bevægelse.

10D Spilleren nåede ikke at spille Imachination.

Hvis man kigger på heatmaps fra børnene der gennemførte spillet både med og uden KI, er der en tendens, der går igen i de fleste af tilfældende. I figur 12.1 ser vi et tydeligt eksempel på forskellen med og uden KI. Der er stor

forskel på, hvor meget af banen, der bliver benyttet af spilleren. I stedet for at parkere få steder på banen, bliver der kørt langt mere.

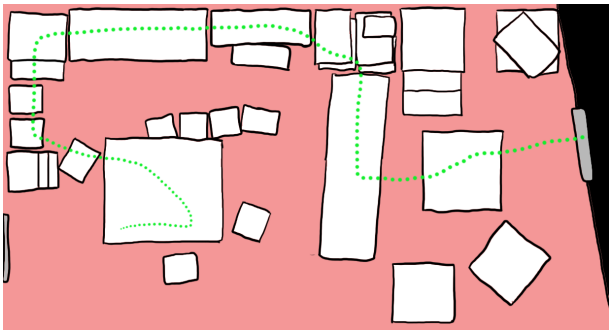


(a) Med KI (b) Uden KI

Figur 12.1: 2P's gennemløb af Imachination

Sophie's Dream Leap

På figur 12.5 ses den tid, det tog børnene at spille banen henholdsvis med og uden kunstig intelligens. Figur 12.2 viser den rute KI'en går gennem banen, repræsenteret ved grønne prikker. Den røde farve angiver syre, der ikke kan betrædes. Hvide områder angiver møbler, der kan betrædes. Den grå kasse i venstre side af banen er indgangsdøren. Den grå kasse i højre side af banen er målet.



Figur 12.2: KI'ens rute gennem banen

1D Spillede kun uden KI. Er død mange gange. Har brugt meget tid på at forsøge at komme igennem banen mod højre. Fandt til sidst en lettere vej igennem banen og gennemførte.

Barn	1D	2P	3D	4P	5D	6D	7D	8D	9D	10D
Med KI	-	-	-	188	161	61	84	81	87	50
Uden KI	184	301*	-	42*	165	17	33	132	29	74
* betyder, at barnet ikke har gennemført banen - betyder, at barnet ikke har spillet banen										

Tabel 12.5: Gennemførselstider for Sophie's Dream Leap

2P Spillede kun uden KI. Døde rigtig mange gange i et forsøg på at bevæge sig igennem banen mod højre. Gav op tæt ved mål, da hun døde.

3D Spillede ikke Sophie's Dream Leap.

4P Havde problemer med styringen og døde derfor et par gange i starten. Fulgte efter KI'en rundt i banen. Gav op i banen uden KI, da hun døde.

5D Forsøgte at bevæge sig igennem banen mod højre, men faldt i syren en del gange. Fulgte til slut efter KI'en og gennemførte. Fulgte samme vej, da KI'en ikke var til stede længere.

6D Opdagede at måldøren var åben til at starte med når KI'en ikke var til stede, og gik derfor direkte igennem banen uden at aktivere syren. Fulgte efter KI'en igennem banen, da den var til stede.

7D Gennemførte hurtigt banen uden KI og fulgte den rute, som KI'en også bruger. Brugte mere tid, da KI'en blev introduceret, idet han var nødt til at følge med KI'en for at få åbnet måldøren.

8D Startede med at hoppe op på sofaen, døde efter et hop og blev *respawnet* på bordet. Fulgte efter et par forsøg KI-vejen. Havde nemt ved at følge KI'en, da den blev introduceret.

9D Fulgte efter KI'en rundt i banen. Kom igennem banen på den hurtigste tid (bortset fra 6D, der 'snød') i udgaven uden KI.

10D Fulgte efter KI'en og kom igennem banen uden at dø. Da KI'en blev fjernet, forsøgte han at komme direkte igennem banen mod højre, men døde nogle gange. Klarede til sidst banen ved stort set at bruge KI-vejen.

12.4 Delkonklusioner

Catnapped

Der er stor forskel på målene, når man spiller med og uden KI. Hvis KI'en er med i spillet, forsøger børnene at slå den ved at bevæge sig rundt uden at blive set.

Som det tydeligt kan ses på heatmappene af Catnapped (appendiks C.2), har gennemløbene mange 'varme' områder når KI'en er med. Det skyldes, at børnene er gået i stå eller har ventet på at komme forbi katteøjnene. Til forskel er der næsten ingen 'varme' områder på nogen af de heatmaps til gennemløbene uden KI. Det betyder, som forventet, at spillerne har bevæget sig mere, når KI'en ikke var til stede i banen.

Hvis det var for svært for børnene med KI'en til stede, gav de op. Spillet er ikke sjovt, hvis man ikke har nogen chance for at gennemføre det, når KI'en er der. Til gengæld var det stadig muligt for nogle af børnene at have det sjovt når KI'en ikke var til stede, selvom de ikke gennemførte. Det kom især til udtryk hos den dreng, som fulgte efter kakerlakkerne. Han forsøgte også at komme ud af banen på uautoriseret vis, ved at lede efter huller imellem kasserne, der danner væggene i banen. Han fik en helt anden spiloplevelse ved at ændre sit eget mål. Som vi forventede, blev hans valg mindre velovervejde i forhold til det primære mål, da der ikke var nogen fare ved at bevæge sig rundt. Til sidst endte han dog med at lede efter udgangen.

Succesraten for gennemførsel af spillet var langt højere, når KI'en ikke var med. KI'en tog modet fra børnene, selvom de synes det var spændende, når den onde kat kiggede efter dem. Det ses på tiderne i tabel 12.3, at børnene har spillet banen igennem på kortere tid uden KI, på trods af at de ikke gennemførte, da KI'en var til stede.

Da man får mest ud af labyrinten ved at gennemføre, eller at finde alternative mål, er banen mest vellykket, hvis den nuværende KI ikke er til stede. Hvis det skulle give en bedre spiloplevelse at inkludere KI'en, skulle det være lettere at undgå øjnene. Samtidig kunne man have inkluderet mere KI af andre former, for eksempel at kakerlakkerne var en reel del af spillet.

Imachination

Hvis man kigger på heatmappene i appendiks C.1, kan man se en gennemgående tendens til mere bevægelse, når børnene spiller med KI. Flere af børnene udtrykte begejstring over at skyde de små fjender og få lov til at se dem flyve ud af skærmen. Derfor er der gået en del tid med både at undvige fjenderne for ikke at dø, men også med at skyde dem for morskabens skyld. Man kan med stor sikkerhed antage, at den første version børnene spiller, vil tage lidt længere tid, da de skal lære spillet at kende. Når de kommer til anden udgave, har de allerede de fundamentale ting om spillet på plads og kan derfor komme

lidt hurtigere igennem. Man kan se af tabel 12.4 på side 41 at tiderne med KI er højere end tiderne uden. Det er nemmere for spilleren at bekæmpe bygningerne hurtigt og effektivt, da der ikke er nogen fare for at blive omringet af fjenderne. I de få tilfælde, hvor tiden uden KI er lige så høj som tiden med KI, har spilleren spillet uden KI først, og har dermed indlæringstiden med heri. I 4P's tilfælde opgav hun spillet med KI, da hun døde, og derfor er hendes tid med KI lavere end den uden.

At KI'en fungerer som et hint til spilleren om at bygningerne skal bekæmpes, har vi et fremragende eksempel på. Resultaterne fra 4P viser, at hun ikke kunne finde ud af, hvad målet med spillet var, og hvordan hun skulle komme videre, da hun spillede uden KI. Det lykkedes hende i testen at "snyde" sig igennem tutorialen og derved ikke forstå at hun skulle ødelægge bygningerne for at komme videre i banen. Derfor er hendes heatmap i versionen uden KI, som hun spillede først, med meget mere bevægelse end hun behøvede, fordi hun blot kørte rundt uden at kunne finde ud af, hvad hun skulle.

KI'en tilføjede også en smule udfordring til spillet. Selvom de små fjender ikke giver så meget skade, var der nogle af børnene, der døde og måtte starte forfra på banen. På trods af at man både kan dø til bygningerne og til bossen i slutningen af banen var der ingen af børnene, der døde i udgaven uden KI. Så KI'en må have tilføjet noget udfordring, men da der ikke er nogen af børnene der er døde mere end en gang, er det nok et fornuftigt niveau. Ingen af børnene holdt en sikker afstand til bygningerne, selvom de i udgaven uden KI kunne ødelægge dem fra distancen uden selv at tage skade.

Sophie's Dream Leap

Vi forventede, at det tog længere tid for børnene at komme igennem banen, når KI'en ikke var til stede. Hvis børnene fik fat på styremåden i løbet af første gennemløb, tog andet gennemløb kortere tid.

5D havde svært ved styremåden og døde derfor mange gange både med og uden KI. Da KI'en agerer gemmepunkt visse steder, betyder en død i udgaven uden KI, at spilleren *respawner* næsten i starten af banen. Derfor er hans gennemløbstider for de to udgaver stort set ens, selvom han døde færre gange i det andet gennemløb, hvor KI'en ikke var til stede.

7D fangede med det samme både styremåde, samt den nemmeste vej igennem banen. Hans andet gennemløb er derfor langsommere end det første, da han i KI-udgaven bliver nødt til at vente på at KI'en også får gennemført banen førend måldøren åbnes.

8D og 9D følger et ens mønster: Deres første gennemløb var en del langsommere end det næste.

10D gennemførte første gennemløb af banen uden at dø. Ved andet gennemløb, som var uden KI, forsøgte han at komme igennem banen direkte mod højre, som kan være en hurtigere vej, men også er sværere. Da han er

Barn	5D	7D	8D	9D	10D
Første gennemløb	161	33	132	87	50
Andet gennemløb	165	84	81	29	74

Tabel 12.6: Gennemførselstider for Sophie's Dream Leap. Første/andet gennemløb.

Barn	1D	2P	3D	4P	5D	6D	7D	8D	9D	10D
Med KI	-	-	-	7	11	0	0	1	0	0
Uden KI	10	35	-	1*	8	0	0	9	0	8
* betyder at barnet ikke har gennemført banen - betyder at barnet ikke har spillet banen										

Tabel 12.7: Antal dødsfald i Sophie's Dream Leap

død meget tæt på mål, og derfor skulle løbe hele banen igennem igen, blev hans anden gennemløbstid langsommere end den første.

Gennemløbstiderne for de børn der gennemførte begge gennemløb (og ikke 'snød', som 6D gjorde i sit første gennemløb), kan ses i tabel 12.6.

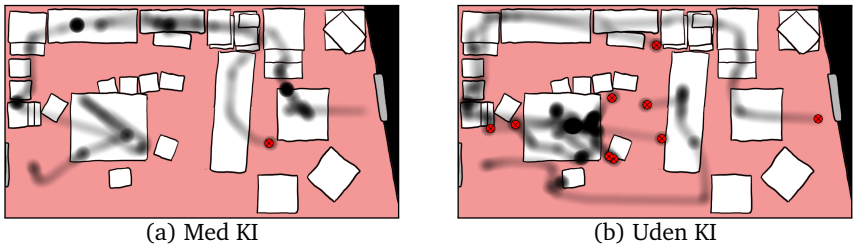
Tilstedeværelsen af KI, var altså ikke det, der gjorde udslaget på gennemløbstide, men derimod blot om børnene havde prøvet spillet en gang før eller ej. Vores forventninger om sammenhæng mellem KI og gennemløbstid stemte altså ikke overens, med de resultater vi opnåede.

Vi forventede, at der ville være flere dødsfald i Sophie's Dream Leap, når KI'en blev fjernet, da spillerne ikke længere blev guidet til den letteste vej gennem banen.

De antal gange børnene døde i deres gennemløb kan ses på tabel 12.7. 5D havde cirka samme antal dødsfald i begge gennemløb. 4P kan man se bort fra, da hun ikke ville spille mere, da hun døde første gang i andet gennemløb, og det derfor ikke er til at vide, hvordan hun ville have klaret sig. 6D, 7D og 9D klarede sig igennem begge gennemløb uden at dø. Det interessante kommer, når man kigger på 1D, 2P, 8D og 10D. 1D og 2P spillede aldrig banen med KI, men uden KI var deres antal dødsfald meget høje. Både 8D og 10D har højere antal dødsfald i gennemløbet. Det kan skyldes, at de ikke er blevet ledt den lette vej igennem banerne, som man også kan se på de vedlagte heatmaps. Det er især klart på figur 12.3, der viser 8Ds vej igennem banen i de forskellige gennemløb. Uden KI prøver han en lang række forskellige veje og døde mange gange. Da KI'en i hans næste gennemløb bliver introduceret, vælger han den letteste vej igennem banen fra start og dør kun en enkelt gang.

Forventningen om antal dødsfald stemte altså overens med de resultater, vi opnåede fra testen.

Vi forventede, at der ville være mere udforskning, og at vejene, som børnene tog igennem banen, ville variere, når KI'en ikke var til stede. Børnene



(a) Med KI (b) Uden KI

Figur 12.3: 8Ds to gennemløb af Sophie's Dream Leap

prøvede oftere at komme igennem banen direkte til højre, når KI'en ikke var der. Når KI'en var til stede, tog de den lange vej rundt. Forsøg på at tage den direkte højrevej, ses tydeligt på 1D, 2P og 8D's gennemløb uden KI (se bilag C.3). Som den eneste, forsøgte 5D at tage højrevejen igennem banen da KI'en var til stede. Vores forventninger om, at spillerne ville tage alternative veje, når KI'en ikke var der til at guide dem rundt må derfor siges at være blevet bekræftet.

12.5 Metoderefleksion

Overordnet set er vi tilfredse med vores afprøvning i fritidshjemmet. Det lykkedes os at få brugbart data, og vi fik også den ønskede mængde. Der var dog nogle ting, vi erfarede, at vi ikke havde gjort godt nok. Vi havde ikke vurderet, hvilke huller der var i vores testplan. Derfor blev løsningen på børnenes utålmodighed improviseret på stedet. Det ville have været fornuftigt at have en backupplan hvis vores testplan skulle fejle. Derudover skulle vi have overvejet, hvordan vi motiverede børnene hjemmefra. Det resulterede, i at antallet af børn vi testede med, voksede fra seks til ti, da vi var usikre på om vi havde fået nok data. Vi er dog tilfredse med, at vi fik justeret vores testplan, da vi kunne se at vores oprindelige tilgang ikke virkede. Da vi fik justeret så tidligt i testforløbet, undgik vi at komme hjem med utilstrækkeligt data og dermed være nødt til at tage ud at teste igen.

Under testen opdagede vi en væsentlig mangel i vores heatmap-modul: Det skulle have været muligt at starte og stoppe logningen af data med en genvejstast. Vores implementering antog, at barnet blev ved med at spille, og gemte derfor først data, når banen blev gennemført. Som vi tydeligt så af testen, var det ikke tilfældet ved alle børn, og i de tilfælde, hvor barnet forlod spillet midt i det hele, måtte vi spille banen færdig selv og bagefter gå ind at slette alt data der var blevet genereret efter barnet havde forladt computeren. Grunden til at det skulle være en genvejstast er, at børnene trykkede mange tilfældige steder på skærmen, som kunne have stoppet optagelsen og skrevet logfilen, hvis det havde været styret af en synlig knap.

Vi har prøvet at kompensere for, at børnene er bedre til spillet anden gang de spiller det end første, ved at blande det sådan at halvdelen spiller med KI først og den anden halvdel omvendt. På den måde får vi de mest præcise resultater, selvom vi ideelt skulle have to helt ens børn til at spille de to udgaver, uden at have prøvet nogen af dem før.

Vi er af den opfattelse, at vores data er blevet virklighedstro, fordi det lykkedes, os at få børnene til at spille, som de ville have gjort, hvis de sad derhjemme og spillede. Vores bevidste valg om ikke at fortælle børnene, hvad de indgik som forsøgspersoner i, bidrog til resultaternes kvalitet.

12.6 Refleksion over produktionerne

Efter at have været ude at teste i fritidshjemmet er vi blevet meget kritiske over for de test, der blev lavet på vores produktioner. I alle tilfælde var det gamedesigner, instruktør og projektleder, der var ude at teste spillene med børnene. Enten har det feedback de har givet videre til resten af gruppen ikke være korrekt, eller også har testen været meget ringe udført. Der viste sig tydelige problemer under vores forholdsvis simple test af spillene. På den éne dag vi har testet spillene, er det gennemgående resultatat, spillene har været for svære. Leveldesignet på banerne har været langt fra optimalt og børnene har haft problemer med styringen. Flere af børnene har givet op på spillene, fordi de ikke har fået nok feedback fra spillet og er gået i stå. Mange af disse problemer kunne have været afhjulpet, hvis holdet havde investeret noget tid i at løse dem, og det havde hjulpet meget på spiloplevelsen.

Vi har en mistanke om, at børnene blevet hjulpet undervejs i testene. Hvis det er tilfældet, at børnene ikke har spillet 100 % selv, kan man i realiteten ikke rigtig bruge resultaterne til noget. Vi har derfor konstateret, at der til næste DADIU-produktion skal være programmører med til testene, da de studerende fra de andre uddannelser tilsyneladende ikke får tilstrækkelig træning eller undervisning i, hvordan man tester software.

Diskussion 13

Hvis vi havde haft mere tid til projektet, ville vi gerne have forbedret det på en række områder.

Vi er i vores projekt blevet reduceret til at undersøge brugerens progression ud fra data, som var muligt at registrere. Vi ville gerne have undersøgt, hvordan spilleren opfatter den kunstige intelligens, og hvordan det påvirker oplevelsen ved at spille spillet.

Til dette kunne vi godt have tænkt os, at få børnene til at give deres vurderinger, så vi kunne have haft et andet aspekt end det faktuelle. Vi kunne for eksempel have lavet nogle skemaer, der indeholdt *VAS*[1]-skalaer[4]. En *VAS*-skala består af en præcis 10 cm lang linje, hvor yderpunkterne er beskrevet, men der ellers ikke er nogen finere inddeling. Man angiver sit svar ved at sætte en streg imellem de to yderpunkter, svarende til ens mening. Udformningen af et *VAS*-skema ville kræve samme mængde arbejde, som normale spørgeskemaer ville, idet vi stadig skulle udforme spørgsmålene, så de er passende til børn. Til gengæld kunne de måske give bedre resultater, da det ikke er et krav, at børnene skal forstå alle svarmulighederne, men kun yderpunkterne.

En mindre realistisk undersøgelse ville gå ud på at måle spillernes hjerneaktivitet, mens de spiller, og så analysere forskellen med og uden KI. Man ville kunne undersøge forskellen i, hvor stresset og udfordret spilleren er, og hvor meget tilfredsstillelse de får ved at opfylde forskellige mål.

Selvom vi i vores test fik indsamlet det data, vi havde planlagt, kunne vi godt have tænkt os at have udført brugertesten i større skala. Hvis børnene kun skulle spille én af udgaverne af hvert spil, kunne vi have undgået den påvirkning, der forekom ved andet gennemløb af en bane. Det ville have krævet langt flere børn at mindske effekten af deres forskellige spilleevner. En større mængde børn ville være nødvendig for at opnå nok statistisk materiale til at drage en konklusion. En sådan test ville kunne foretages online over længere tid, hvor for eksempel en automatiseret *A/B*-test[11] kunne udføres.

[1] Visual Analogue Scale

Det kunne være interresant at undersøge forskellen i måden at spille på samt spiloplevelse, hvis børnene blot bliver et år eller to ældre, det vil sige 8-9 år istedet for vores 6-7 år. Det ville dog udvide fokusfeltet for meget, da vores DADIU-produktioner er rettet mod 6-7 årige.

Hvis vi havde haft mulighed for at udføre nogle af de ovenstående metoder, ville vi have kunnet give et mere præcist, anvendeligt og statistisk korrekt billede af, hvilken effekt, og i hvor stort omfang, tilføjelse af en kunstig intelligens kan have på et spil. Det der holdt os fra det er naturligvis den store mængde tid, det ville kræve at lave sådanne undersøgelser. Det ville dermed have ligget uden for de realistiske grænser for et virksomheds- og bachelorprojekt af denne art.

Ved at lægge mere fokus på udviklingen af den kunstige intelligens, kunne de alternative eller primære mål i vores spil være styrket. For eksempel kunne der være lavet flere typer af fjender i Imachination, så der blev mere variation og udfordring i 'legen' med de små fjender. I Sophie's Dream Leap kunne broderen blive en hjælper i stedet for blot en vejviser, hvis han for eksempel kunne komme tilbage og hente spilleren hvis hun ikke kunne komme videre. I Catnapped kunne livet i labyrinten øges, ved at kakerlakkerne kunne flygte fra eller tiltrækkes af spilleren og samles i større grupper. De kunne også gøres til en udfordring, ved for eksempel at gøre det sværere at komme forbi katteøjnene, hvis man havde følge af kakerlakker.

Konklusion 14

Vi har af vores test set at KI'en kan give grobund for alternative spiloplevelser i form af leg. Vi har dog også erfaret, at KI'en kan modvirke alternative spilopvelser. I Imachination fik børnene glæde af at lege med den kunstige intelligens, mens vi i Catnapped så, at KI'en direkte modvirkede alternative mål, da børnene fokuserede mere på ikke at dø, end at udforske og lege, for eksempel med kakerlakkerne. Vi har dermed erfaret, at man med KI i høj grad kan påvirke den måde spilleren spiller spillet på, men at man også har mulighed for at styre spilleren ud i, eller væk fra delmål. Dette sås tydeligt i Sophie's Dream Leap, hvor KI'en aktivt ændrede de primære veje, spilleren tog gennem banen.

I rapporten har vi beskrevet vores udviklingsforløb hos DADIU og vores arbejde i Unity. Vi har givet et indblik i de problemer og udfordringer vi stødte på undervejs, og har givet vores bud på forbedringer en fremtidig produktion kan drage nytte af.

Vi har udvidet vores spil med heatmap funktionalitet, der gør os i stand til at registrere hvor i spillene en spiller bevæger sig, samt hvordan tidsfordelingen ligger i forskellige områder i banerne. Derudover kan vi se, hvor lang tid det tager spilleren at færdiggøre baner, samt se hvor de eventuelt dør. Disse data har vi anvendt til at skabe et billede af den kunstige intelligens påvirkning i de forskellige spil.

Vi har erfaret, at der er særlige ting, man skal tage højde for, når man udfører en brugertest på børn. Børnenes koncentrationsevne er begrænset – især når deres venner leger udenfor. Derfor skal man nøje overveje, hvordan man får relevant data fra sin test, uden at kræve for mange ting af børnene.

Vi betragter vores projekt som værende succesfuldt, da vi har haft tre vellykkede praktikforløb hos DADIU, som alle har resulteret i fine produkter. Vores heatmapfunktionalitet har givet os en mulighed for at vurdere den kunstige intelligens påvirkning på spillerens progression i vores spil. Desuden har vi kunnet vurdere og diskutere, i hvor høj grad den kunstige intelligens spiller en rolle – både i forhold til spilprogressionen, samt den oplevelse de får.

Når KI'en giver så store muligheder for at påvirke spillet, er det væsentligt at *game designeren* bruger en del tid på at udtænke og finpudse den kunstige intelligens. I alle vores projekter var gamedesigneren langt fra interreseret i den kunstige intelligens påvirkning på spillet: Der blev brugt mere tid på detaljer i andre dele af spillet. Det på trods af at vi tydeligt kan se, at KI'en spiller en stor rolle. Vi er sikre på, at alle produktionerne kunne være blevet bedre, hvis KI'en i vores spil havde fået mere fokus i designprocessen.

Appendiks

Spillene

A

- Klaes' produktion: `http://catnapped.dk`
- Steens produktion: `http://imachination.dk`
- Julians produktion: `http://dream-leap.com`

Litteratur

B

[1] Entertainment Software Rating Board. Game ratings & descriptor guide, juni 2010. URL http://www.esrb.org/ratings/ratings_guide.jsp.

[2] Natacha Borgers, Edith de Leeuw og Joop Hox. Respondents in survey research: Cognitive development and response quality. *Bulletin de Methodologie Sociologique*, 66:60-75, 2000.

[3] Det Danske Akademi for Digital Interaktiv Underholdning. DADIU, juni 2010. URL http://www.dadiu.dk.

[4] Dinah Gould, Daniel Kelly, Len Goldstone og John Gammon. Information point: Visual analogue scale (vas). *Journal of Clinical Nursing, 10*, side 697-706, 2001.

[5] Aron Granberg. A* pathfinding, juni 2010. URL http://www.arongranberg.com/unity/a-pathfinding/.

[6] Rune Skovbo Johansen. Locomotion system, maj 2010. URL http://unity3d.com/support/resources/unity-extensions/locomotion-ik.

[7] Ian Millington og John Funge. *Artificial Intelligence for Games, Second Edition*. Morgan Kaufmann, 2009.

[8] Janet C. Read og Stuart MacFarlane. Using the fun toolkit and other survey methods to gather opinions in child computer interaction. I *IDC '06: Proceedings of the 2006 conference on Interaction design and children*, side 81-88, New York, NY, USA, 2006. ACM.

[9] Craig W. Reynolds. Flocks, herds, and schools: a distributed behavioral model. I *Seminal graphics: poineering efforts that shaped the field*, side 273-282. ACM, New York, NY, USA, 1998. URL http://www.red3d.com/cwr/boids/.

[10] Unity Technologies. Unity: Game development tool, juni 2010. URL http://unity3d.com.

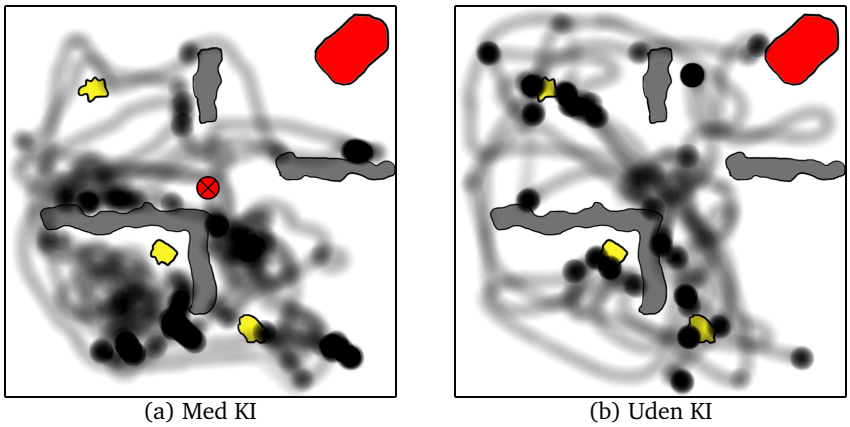
[11] Wikipedia. A/b testing, juni 2010. URL http://en.wikipedia.org/wiki/A/B_testing.

Heatmaps

C

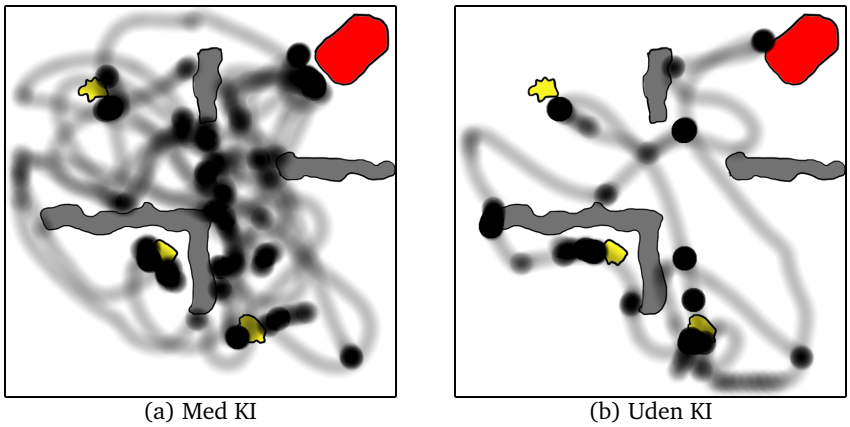
I dette afsnit fremviser vi alle heatmaps der blev gemt under testen. Såfremt der er et hvidt felt skyldes det at barnet ikke spillede den pågældende udgave.

C.1 Imachination



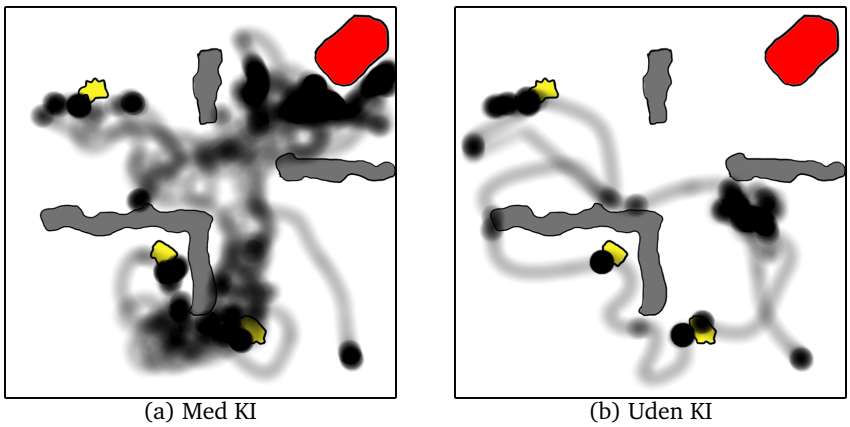
(a) Med KI (b) Uden KI

Figur C.1: 1D. Spillede uden KI først.



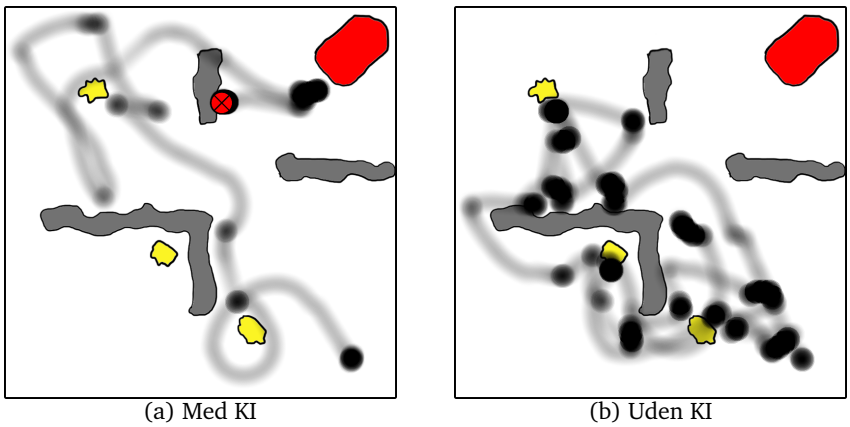
(a) Med KI (b) Uden KI

Figur C.2: 2P. Spillede med KI først.



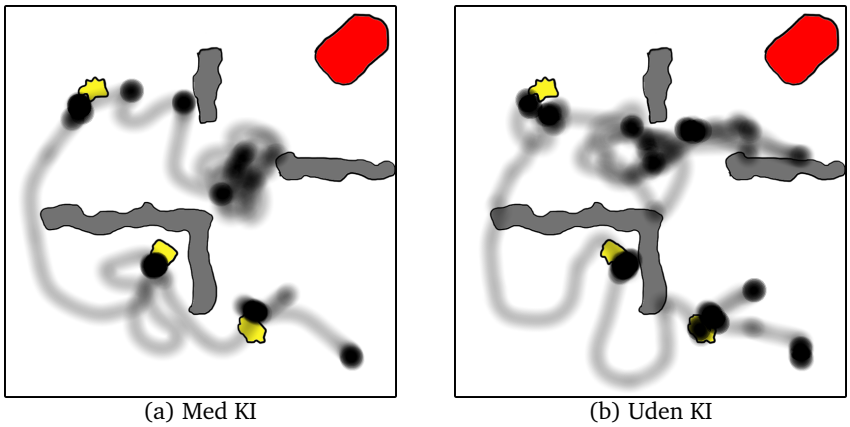
(a) Med KI (b) Uden KI

Figur C.3: 3D. Spillede med KI først.



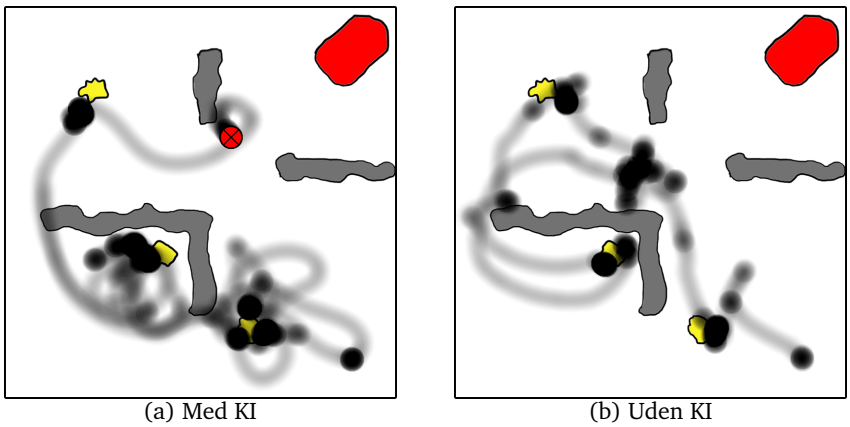
(a) Med KI (b) Uden KI

Figur C.4: 4P. Spillede med KI først.



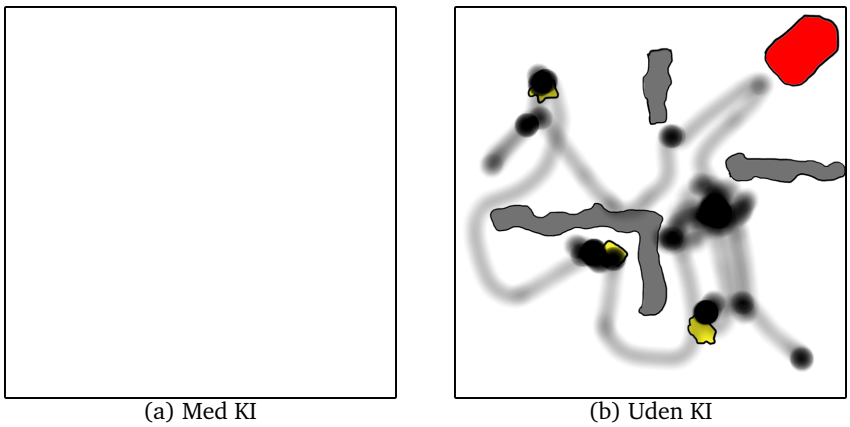
(a) Med KI (b) Uden KI

Figur C.5: 5D. Spillede uden KI først.



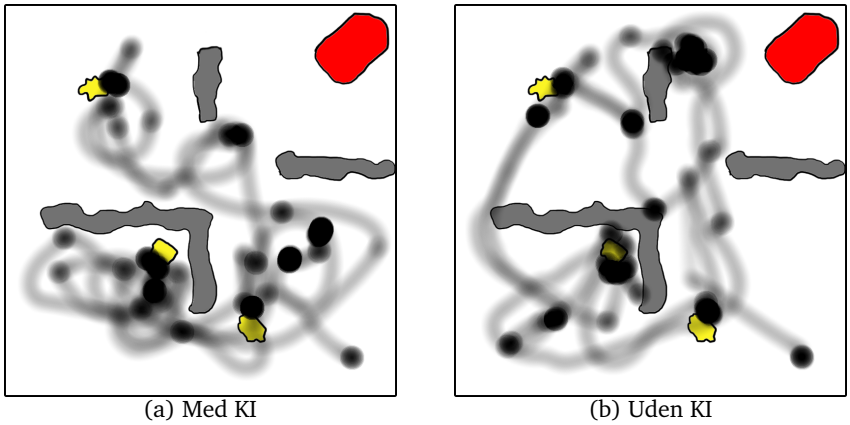
(a) Med KI (b) Uden KI

Figur C.6: 6D. Spillede uden KI først.



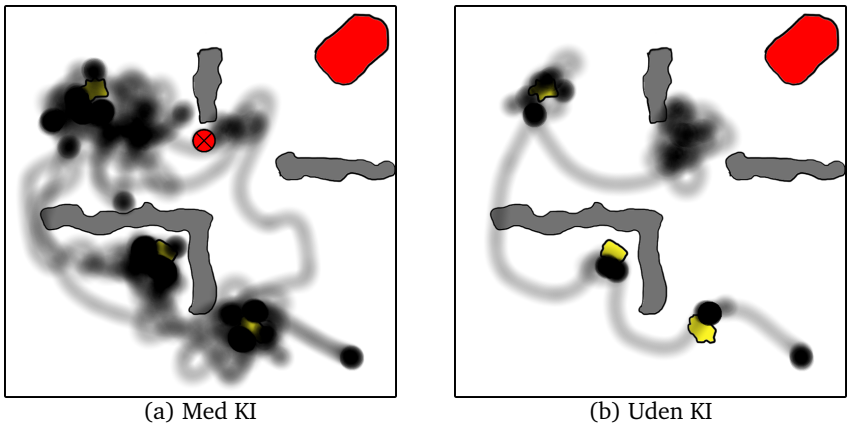
(a) Med KI (b) Uden KI

Figur C.7: 7D. Spillede kun med KI.



(a) Med KI (b) Uden KI

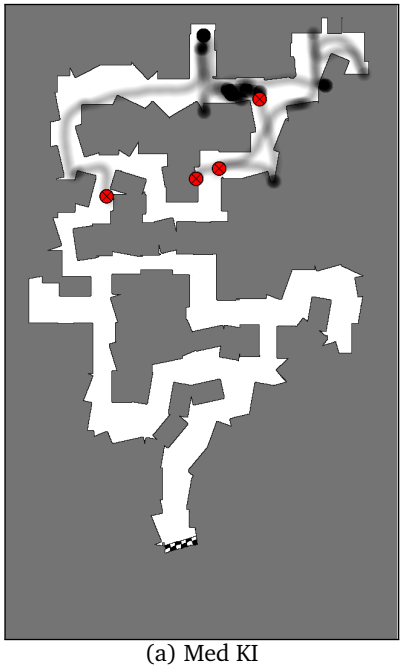
Figur C.8: 8D. Spillede uden KI først.



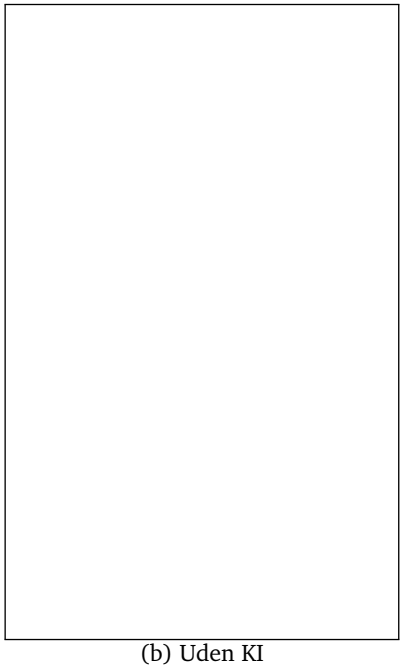
(a) Med KI (b) Uden KI

Figur C.9: 9D. Spillede uden KI først.

C.2 Catnapped



(a) Med KI



(b) Uden KI

Figur C.10: 1D. Spillede kun med KI.

(a) Med KI (b) Uden KI

Figur C.11: 2P. Spillede kun uden KI.

(a) Med KI (b) Uden KI

Figur C.12: 3D. Spillede uden KI først.

(a) Med KI (b) Uden KI

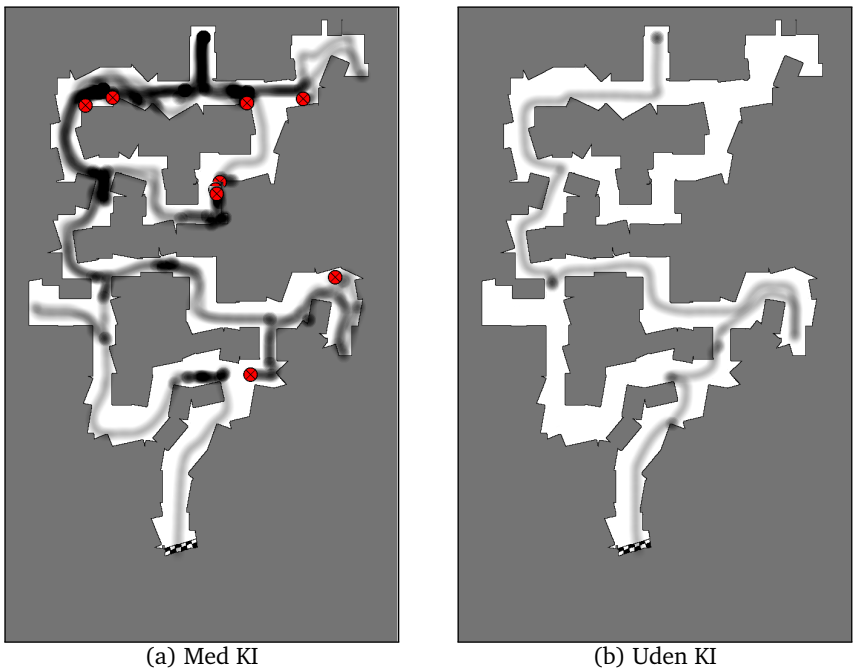
Figur C.13: 4P. Spillede kun med KI.

(a) Med KI (b) Uden KI

Figur C.14: 5D. Spillede med KI først.

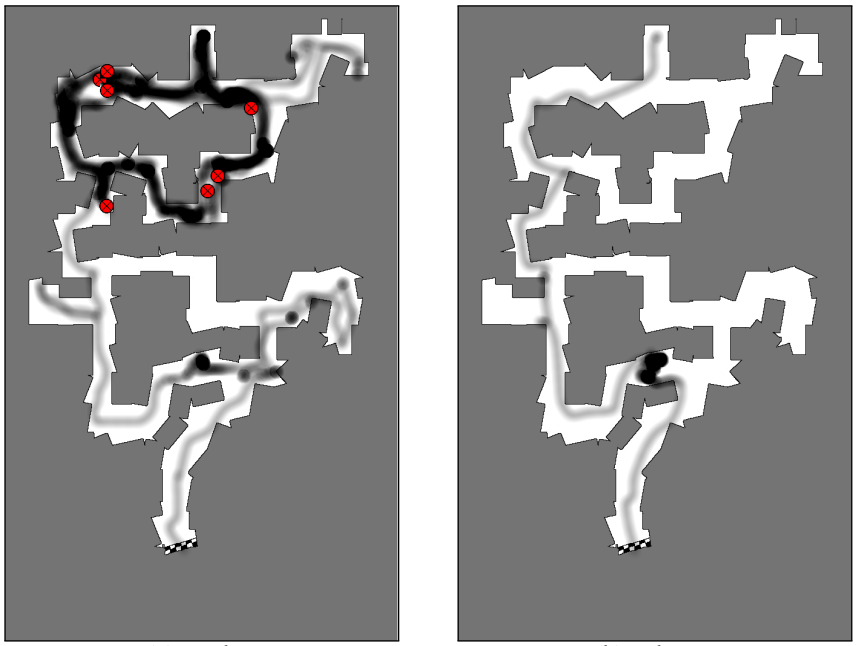
(a) Med KI (b) Uden KI

Figur C.15: 6D. Spillede med KI først.



(a) Med KI (b) Uden KI

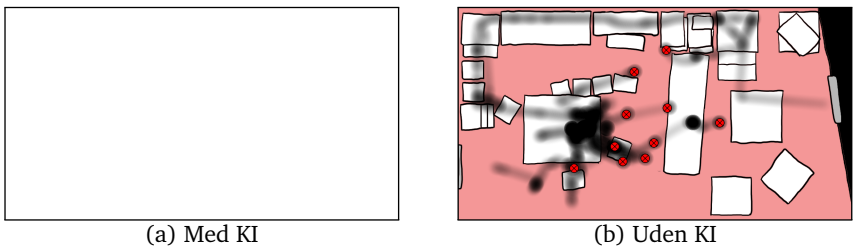
Figur C.16: 8D. Spillede med KI først.



(a) Med KI (b) Uden KI

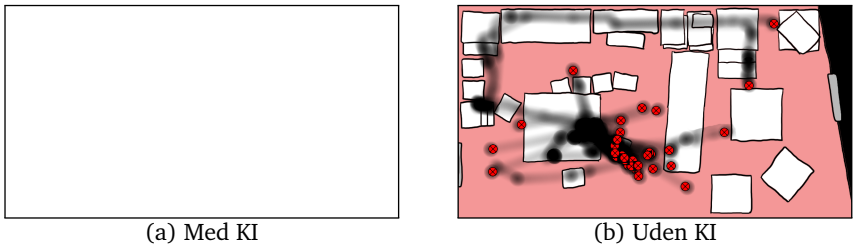
Figur C.17: 9D. Spillede med KI først.

C.3 Sophie's Dream Leap



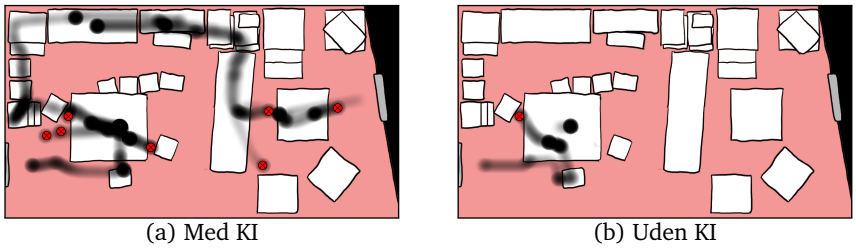
(a) Med KI (b) Uden KI

Figur C.18: 1D. Spillede kun uden KI.



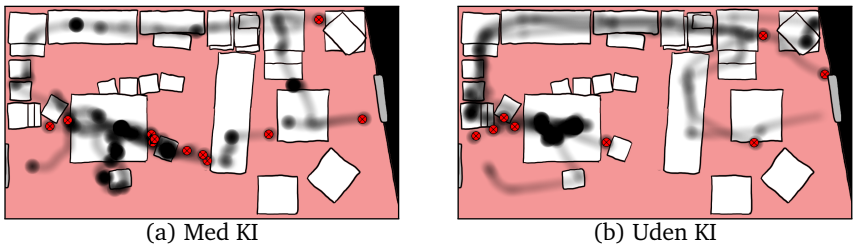
(a) Med KI (b) Uden KI

Figur C.19: 2P. Spillede kun uden KI.



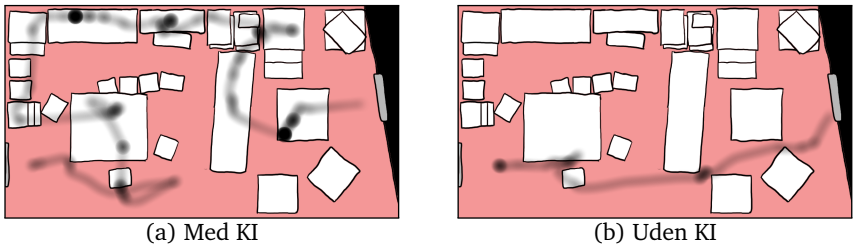
(a) Med KI (b) Uden KI

Figur C.20: 4P. Spillede med KI først.



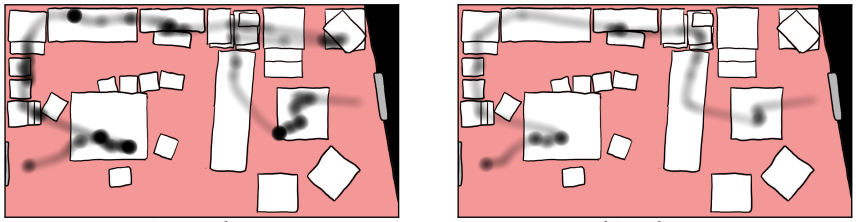
(a) Med KI (b) Uden KI

Figur C.21: 5D. Spillede med KI først.



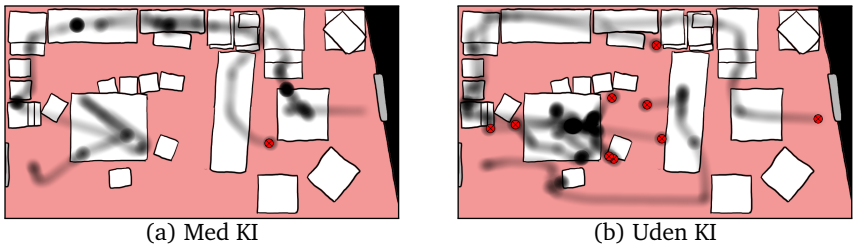
(a) Med KI (b) Uden KI

Figur C.22: 6D. Spillede uden KI først.



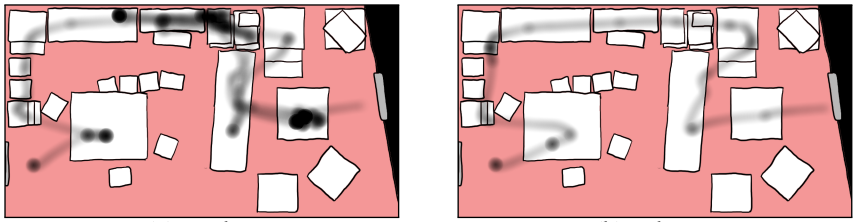
(a) Med KI (b) Uden KI

Figur C.23: 7D. Spillede uden KI først.



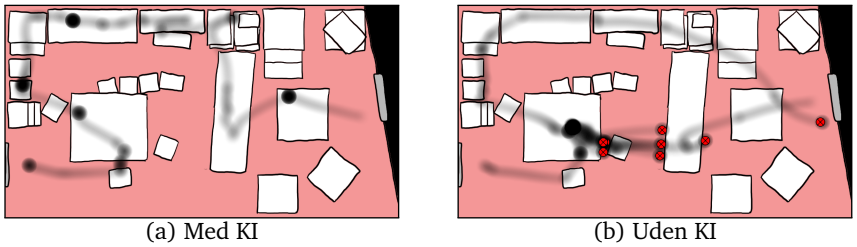
(a) Med KI (b) Uden KI

Figur C.24: 8D. Spillede uden KI først.



(a) Med KI (b) Uden KI

Figur C.25: 9D. Spillede med KI først.



(a) Med KI (b) Uden KI

Figur C.26: 10D. Spillede med KI først.

Fulde DADIU begrænsninger

D

DADIU DET DANSKE AKADEMI FOR DIGITAL INTERAKTIV UNDERHOLDNING

Theodor Christensens Plads 1
1437 København K
Danmark

Telephone: +45 3268 6519

dadiu@dadiu.dk
www.dadiu.dk

Conditions for the Joint Production, May 2010

The Assignment

Goals for the May 2010 DADIU Production
For many this will be their first DADIU production and therefore you will be given a set of strict conditions to help you focus and scope the project. Being part of a DADIU production is as much an exercise in working in a large team as it is an exercise in creating a finished computer game.

The delivered game **must** be a finished, polished and fully playable game. In order to accomplish this within such a short production period, you will have to focus on very few game features and re-use them as much as possible.

On participating in your first DADIU production
In the first DADIU production the focus is more on the process than the product. You will also be evaluated on your game, but you should yourself reflect and evaluate yourself as how you worked as a team player. Since the conditions for the first DADIU production are quite strict you might find that your talents and skills cannot be used to your own satisfaction. In the second DADIU production you will be given a lot more freedom and should then know the nature of a DADIU team.

So participate with an open mind and regard the first production for what it is: a learning experience and a unique chance to create a finished computer game.

The Conditions

Conditions for Content

- The game **must** be playable as a single player game
- The game **must** include 3D elements and a main character
- The target audience **must** be 3-7 year old children
    - A first time player (or players) **must** have a well-rounded game experience after playing the game for a maximum of 5 minutes
    - The final (or near final) game **must** be tested on the intended target audience and the result of this test is a delivery on its own (see below)
    - You may optionally narrow down the target audience (example: 5-6 year old boys)
- The game **must** be controlled by a mouse and only use the left mouse button
- The game **must** communicate clearly to the player how the game is played
    - The games ability to teach the player the game rules and how to play the game **must** be part of the final test
- The game **must** include an AI (artificial intelligence) feature
    - The AI feature (or features) **must** be directly related to the core game mechanics
    - The scope of the AI feature(s) may be as simple as a path-finder or as complex as a competent AI opponent or an adaptive difficulty system

(those are just examples), but the scope **must** be defined in the Technical Document (see conditions on the Technical Document below for further details)

- The game **must** present itself in a Danish and an English version involving a technical and practical solution to handle translations

Technical Conditions

- The game **must** be developed in Unity
- The game **must** be playable through the Unity web player. This means that you will have to consider load times and asset sizes carefully
- The game **must** be playable on a standard mid-range pc[1]
- The game **must** be completed in the sense that it is optimized and can be played by others

April			May		
1	T	”Skærtorsdag”	1	L	
2	F	”Langfredag”	2	S	
3	L		3	M	**Delivery: Design Doc, Production Doc, Prototype**
4	S	”Påskedag”	4	T	**Production Phase**
5	M	”2. Påskedag”	5	O	*Consultation Group (on loc.)*
6	T		6	T	
7	O		7	F	
8	T		8	L	
9	F		9	S	
10	L		10	M	week 19
11	S		11	T	
12	M	**Online Phase, Conditions**	12	O	*Consultation Group (on loc.)*
13	T		13	T	”Kristi Him.dag”
14	O		14	F	
15	T		15	L	
16	F		16	S	
17	L		17	M	Uge 20
18	S		18	T	
19	M	week 16	19	O	*Publisher visit (on loc.)*
20	T		20	T	
21	O		21	F	
22	T		22	L	
23	F	(no deliviries)	23	S	”Pinsedag”
24	L		24	M	”2. Pinsedag”
25	S		25	T	**Delivery: Final Game, Test Analysis Doc, Post Mortem**
26	M	**Preproduction Phase**	26	O	
27	T		27	T	Evaluation (CPH)

[1] Check http://unity3d.com/webplayer/hwstats/ for details on users hardware.

28	O	*Consultation Group (on loc.)*
29	T	
30	F	”St. Bededag”

28	F	Evaluation (CPH)
29	L	
30	S	