Examensarbete

HÖGSKOLAN I SKÖVDE

HUR SPELARERFARENHET PÅVERKAR STRATEGIVAL VID IMPASSE-DRIVNA INLÄRNINGSMOMENT I ETT ACTIONPLATTFORMSPEL

HOW PLAYER EXPERIENCE AFFECTS CHOICE OF STRATEGY DURING IMPASSE-DRIVEN LEARNING IN AN ACTION PLATFORM GAME

Examensarbete inom huvudområdet Informationsteknologi
Grundnivå 30 högskolepoäng
Vårtermin 2019

William Jarlow
August Hassnert

Handledare: Erik Sjöstrand
Examinator: Jenny Brusk

Sammanfattning

Syftet med studien är att besvara hur strategierna *Repair* och *External Information Retrieval* (*EIR*) används av spelare av varierande erfarenhetsnivå för att tackla *impasse*-drivna inlärningsmoment i actionplattformspel. En artefakt i form av en spelprototyp inom actionplattform-genren har skapats för att besvara frågeställningen.

Studiens testgrupp bestod av tre mycket erfarna spelare, tre mellan-erfarna spelare och två icke-erfarna spelare. Testdeltagarna ombads att spela igenom spelprototypen och förklara hur de tänkte enligt think-aloud metoden. Intervjuer utfördes både innan och efter spelsessionerna för att avgöra deltagarnas erfarenhetsnivåer och tydliggöra varje deltagares bakgrund inom spel. Varje enskild spelsession spelades in med deltagarnas tillåtelse för att möjliggöra analys vid senare tillfällen.

Studiens resultat var att de mer erfarna spelarna använde *Repair* oftare än mindre erfarna spelare, och att de presterade bättre när de gjorde det. Icke-erfarna spelare föredrog *EIR*, men det visades också att mindre erfarna spelare behövde mer detaljerad information för att utföra effektiv *EIR*.

Nyckelord: *Impasse*-driven inlärning, inlärningsstrategier, erfarenhet, spel.

Innehåll

1 Introduktion

Erfarenhet är en faktor som starkt påverkar hur spelare interagerar med spel. En mer erfaren spelare har en djupare förståelse för spelet hen spelar, och kan därför använda sig av mer avancerade strategier än en icke-erfaren spelare kan (Elias, Garfield & Gutschera, 2012). Det medför därför att spelare har olika krav på speldesignen beroende på hur erfarna de är, vilket tydligt kan synas inom tutorial-design (Suddaby, 2012; Hedges, 2017).

VanLehn (1988) utförde en studie där han myntade uttrycket *impasse*-driven inlärning, vilket är en inlärningsmetod som grundas i att överkomma utmaningar. En *impasse*, det vill säga ett dödläge, uppstår när någon ställs inför ett hinder som de inte har kunskapen för att klara av. Själva inlärningen uppstår när man söker efter ny information och till slut klarar av utmaningen. Det finns två strategier som kan användas vid *impasse*-driven inlärning: *Repair* och *External Information Retrieval*, hädanefter kallad *EIR*. *Repair* går ut på att man försöker lösa en *impasse* helt självständigt genom sin problemlösningsförmåga och sina tidigare erfarenheter, exempelvis genom att utföra trial-and-error för att se vad som fungerar. *EIR* innebär att man söker extern information för att lösa en *impasse*, exempelvis genom att fråga någon som vet eller att söka på internet.

Det här arbetet undersöker hur spelarerfarenhet påverkar hur spelare använder sig av *Repair* och *EIR* vid *impasse*-drivna inlärningsmoment. Forskningsämnet behandlar både erfarenhet och strategival; två aspekter som är väldigt svåra att kvantifiera. Arbetet använder sig av kvalitativa metoder för att samla så detaljrika data som möjligt, vilket gör det möjligt att fokusera på spelarnas resonemang och inte bara deras resultat. För att kunna testa *impasses* har en spelprototyp tagits fram, som är designad för att få *impasses* att uppstå för spelare av alla olika erfarenhetsnivåer. Förhoppningsvis kan arbetet ge en bättre förståelse för hur spelare av olika erfarenhetsnivåer tänker, vilket kan göra det lättare att designa spel som ger ordentligt stöd för spelare när de stöter på dödlägen och fastnar.

Spelprototypen är ett actionplattformspel i vilket spelaren måste lära sig olika mekaniker för att kunna nå och slå sönder måltavlor i sju banor. För att underlätta *EIR* innehåller spelet en lista på alla mekaniker, samt beskrivningar och demonstration-videor som visar hur de fungerar. Prototypens kontroller följer vissa konventioner och bryter mot andra för att medvetet påverka hur erfarna spelare interagerar med spelet.

Åtta spelare har deltagit i undersökningen. Alla fick självskatta sin erfarenhetsnivå av spel innan deras spelsession började, både generellt sett och av spel inom actionplattform-genren. Think-aloud-metoden användes under spelsessionerna för att samla in data om hur spelarna resonerade, och spelsessionerna spelades in för vidare gameplay-analys. Efter varje spelsession frågades deltagarna om de kände igen någon av mekanikerna från andra spel, eftersom detta är en faktor som kan ha påverkat deras förväntningar om hur mekanikerna fungerade.

2 Bakgrund

I bakgrunden redovisas forskning kring tre ämnen som är centrala för denna studie: *Impasse*-driven inlärning, hur inlärning och speltutorials är kopplade, samt hur spelstilar kan påverkas av spelerfarenhet.

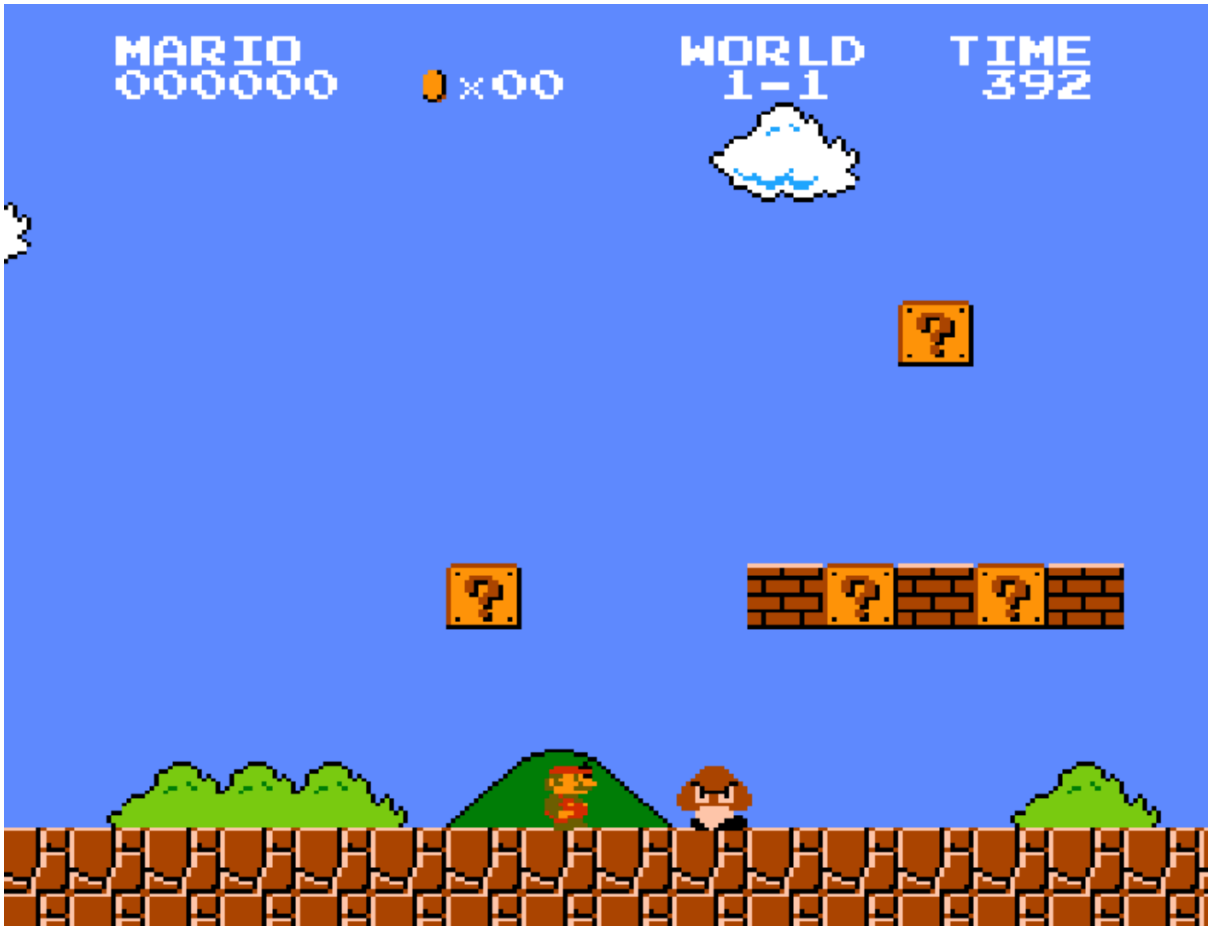
2.1 Impasse-driven inlärning

Impasse-driven inlärning är enligt VanLehn (1988) en inlärningsmetod som bygger på att använda misslyckande för att främja inlärning. Vad detta innebär i praktiken är att människor lär sig bättre när de misslyckas med en utmaning och tvingas att omvärdera sina metoder för att klara av utmaningen. Namnet kommer från engelskans *impasse*, vilket översätts till dödläge på svenska. Fördelen med *impasse*-driven inlärning är att man lär sig med ett mål i åtanke, vilket gör det lättare att leta efter relevant information och känna igen liknande problem senare. Konceptet av att använda misslyckanden för att främja inlärning har dykt upp i ett flertal andra studier (Darabi, Arrington och Sayilir, 2018), men de studierna har alla myntat sina egna begrepp för att beskriva sina metoder och deras steg. Den här studien kommer att utgå från VanLehns (1988) terminologi om *impasse*-driven inlärning då den definierar specifika strategier som är lätta att applicera på spel.

Det finns två strategier som människor använder när de ställs inför en *impasse*: antingen använder de *Repair* eller så söker de utomstående hjälp. *Repair* är vad VanLehn (1988) kallar det när någon försöker lösa en *impasse* på egen hand, genom att använda sin problemlösningsförmåga och sina tidigare erfarenheter för att resonera sig fram till en lösning. Ofta sker detta genom trial-and-error, men det är också fullt möjligt att lista ut lösningen helt i huvudet utan att prova olika strategier.

VanLehn (1988) menar att det finns en risk med att använda *Repair* för att lösa en *impasse*, då buggar som han kallar dem kan uppstå i den nya kunskapen. Han definierar en bugg som ett enstaka fel i den annars korrekta *impasse*-lösningen, ofta grundat i missförstånd och felaktiga antaganden, och kan orsaka fler *impasses* i framtiden. Risken att buggar uppstår minskar ju mer erfarenhet en person har med en aktivitet, då erfarenhet kan göra det lättare för personen att förstå en *impasse* och dess lösning.

Det bästa sättet att undvika buggar är att använda *External Information Retrieval* (*EIR*), alltså den andra strategin som VanLehn (1988) skriver om. *EIR* innefattar att man söker hjälp från utomstående resurser hellre än att lösa en *impasse* själv, vilket enligt VanLehn ger en mycket klarare bild av lösningen för en *impasse*. Så länge som den externa informationen i sig inte är felaktig minskas risken för buggar avsevärt. Ett exempel på *EIR* som VanLehn tar upp är att fråga en lärare om ett matteproblem eller att söka information från läroböcker, och ett mer modernt exempel skulle kunna vara att använda sökmotorer på internet.



Figur 1 Den första fienden i *Super Mario Bros.* (1985)

Den första banan i *Super Mario Bros.* (1985, se Figur 1) är ett välkänt exempel på hur *impasse*-driven inlärning kan användas i spel. Iwata och Miyamoto (2009), två spelutvecklare som arbetade på spelet, berättade i en intervju att spelare utan tidigare kunskap förväntas springa in i den första fienden, varpå de förlorar ett liv och måste börja om. Spelaren ställs alltså inför en *impasse*; de måste ta sig förbi fienden för att fortsätta spela. Spelets kontroller är enkla och spelaren har inte många alternativ, så det är logiskt att de snabbt provar att hoppa över fienden. På grund av lådorna som är placerade över fienden så är det troligt att spelaren landar på den hellre än hoppar över den, varpå de lär sig att man kan besegra fiender genom att hoppa på dem. Om spelaren slår i en av lådorna ovanför sig lär hen sig också att man blir belönad av att hoppa upp i lådor med frågetecken. All denna information går att lära ut genom en så enkel *impasse* som att en fiende står i vägen för spelaren.

2.2 Inlärningsmetoder i tutorials

Spelare behöver en grundläggande kunskap om ett spel för att veta vad som är möjligt att göra i det, eftersom spelare inte kan interagera med spelet på ett meningsfullt sätt om möjligheterna är helt okända. *Impasse*-driven inlärning som VanLehn (1988) skriver om kan användas för att lära ut den informationen, genom att låta spelare fastna och själva ta sig förbi hindren genom *Repair*. Detta är dock riskabelt eftersom buggar kan uppstå i den mest grundläggande kunskapen vilket gör det svårt för spelarna att lära sig de mer avancerade mekanikerna. Därför förser spel instruktioner till sina spelare genom tutorials, som är korta träningsmoment där spelare får lära sig om spelets mekaniker och regler (Suddaby, 2012).

Hedges (2017) tar upp olika positiva och negativa aspekter som tutorials kan ha. I regel menar han att tutorials bör undvika att tråka ut spelaren, och att de ska vara så integrerade med spelet som möjligt; tutorialen ska inte kännas som en separat del som man måste klara av innan man kommer till det riktiga spelet. Om tutorialen innehåller för många negativa aspekter uppstår risken att spelaren blir så uttråkad att hen slutar spela (Hedges, 2017; Suddaby, 2012), vilket motverkar syftet med att inkludera en tutorial till att börja med.

Kraven på vad som behöver läras ut till spelaren varierar mycket mellan olika spel, så det finns ingen enhetlig tutorial-metod som garanterat fungerar i alla spel. Det finns dock ett antal metoder som ofta förekommer i spel med olika variationer. Suddaby (2012) tar upp fyra exempel på sådana: expositionsbaserade tutorials, tutorial-rummet, kontextuella tutorials och tematiskt relevanta kontextuella tutorials.

Expositionsbaserade tutorials fungerar genom att berätta för spelaren hur man spelar utan att spelaren får interagera med tutorialen. Exempel på dessa är att visa spelaren en bild där alla kontroller står, eller att förklara spelets mekaniker genom text innan spelet börjar. Suddaby (2012) menar att detta är ett av de minst effektiva sätten att lära spelaren spela, eftersom det visas för mycket information för att spelaren ska kunna komma ihåg de relevanta delarna senare.

Palladino (2017) skriver att människors uppmärksamhet försämras när de utsätts för för mycket eller för lite stimulering, vilket stöder Suddabys (2012) påstående då expositionsbaserade tutorials ofta överbelastar spelaren med information. Kontext tas också upp som ett problem, eftersom spelare kan ha svårt att koppla informationen de lär sig med handlingar som kan utföras i spelet. Expositionsbaserade tutorials fungerar bäst i mindre komplexa spel, då risken att spelaren blir överbelastad med information inte är lika hög i ett enkelt spel. Suddaby menar att man bör undvika att använda expositionsbaserade tutorials eftersom de har så många svagheter, men han skriver även att det är den lättaste tutorialen att implementera. I vissa begränsade situationer, exempelvis om man visar upp sitt spel på en mässa, kan det vara bäst att ha en expositionsbaserad tutorial, då spelarna på mässan antagligen inte kommer ha tid att spela igenom en längre, bättre tutorial.

Vidare använder expositionsbaserade tutorials huvudsakligen inte *impasse*-driven inlärning, då spelaren inte har tid att stöta på en *impasse* innan de får informationen som de behöver lära sig. Tutorials av detta slag kan dock finnas tillgängliga under hela spelets gång, exempelvis genom att visa kontrollerna i laddningsskärmar, så de har potentialen att stödja *impasse*-driven inlärning.

Suddaby (2012) skriver även om tutorial-rum, som innebär att tutorialen tar plats i ett rum separat från andra delar av spelet. Tutorial-rum låter spelaren interagera med mekanikerna i en kontrollerad miljö, och kräver ofta att spelaren demonstrerar att de kan använda mekanikerna som tutorialen försöker lära ut. Enligt Suddaby fungerar Tutorial-rummet bättre än expositionsbaserade tutorials eftersom spelarna får instruktioner samtidigt som de interagerar med spelet, vilket tillåter dem att använda kunskapen de lär sig i praktiken. Ett flertal av nackdelarna som Hedges (2017) tar upp är dock applicerbara på tutorial-rum: de brukar ha för mycket handledning, de varar ofta för länge och de kan avbryta spelets flow, beroende på hur tidigt in i spelet tutorial-rummet kommer.

Suddaby (2012) skriver att tutorial-rum kan passa bra i strategispel där det finns mycket att lära sig, då spelaren får spela spelet själva samtidigt som de får reda på vad de borde göra

härnäst. Som med alla tutorials varierar effektiviteten av tutorial-rum beroende på hur väl de implementeras; en bra implementation kan agera som en rolig introduktion till spelet medan en dålig implementation kan upplevas som ett slöseri med tid.

Figur 2 Tutorial-rummet i *FTL: Faster Than Light* (2012)

FTL: Faster Than Light (2012) har ett intressant exempel på ett tutorial-rum, där spelaren får lära sig att kontrollera sitt rymdskepp och strida mot en enkel motståndare med hjälp av meddelanden som förklarar spelet (se Figur 2). Första gången spelet startas får spelaren svara på om de har spelat spelet tidigare eller om de vill spela tutorialen, för att undvika att redan erfarna spelare tvingas att spela om tutorialen. Tutorialen kan också startas när som helst från huvudmenyn om en ny spelare spelar spelet hos en kompis till exempel.

Det här tutorial-rummet innehåller några av de negativa aspekterna som Hedges (2017) nämner. Till exempel avbryts spelarens flow när nya meddelanden dyker upp eftersom spelet pausas konstant. En positiv sak är att spelaren får utföra handlingarna själv, hellre än att tutorialen bara demonstrerar vad som behöver göras. *FTL: Faster Than Light* (2012) är ett spel som kräver förståelse för alla grundmekaniker för att en spelare ska kunna kontrollera sitt skepp och slåss mot motståndare, så tutorial-rummet passar väldigt bra för att lära spelaren allt som krävs i en lagom takt. *Impasse*-driven inlärning sker sällan i tutorial-rum då spelet stannar upp och förklarar vad spelaren ska göra i varje steg, så det är osannolikt att spelaren kommer uppfatta en situation som ett dödläge vilket är ett krav för att *impasse*-driven inlärning skall kunna ske (VanLehn, 1988).

Figur 3 En tematiskt relevant kontextuell tutorial i *Spelunky* (2008)

Suddaby (2012) skriver även om kontextuella tutorials. De går ut på att tutorial-meddelanden visas under spelets gång när spelaren stöter på nya mekaniker, och kan ske utan att avbryta spelupplevelsen alls. De är därför väldigt effektiva i långa spel med många mekaniker, eftersom spelaren endast lär sig saker som är relevanta för situationen som de befinner sig i utan att riskera ett informationsöverflöd.

Tematiskt relevanta kontextuella tutorials bygger på samma princip som vanliga kontextuella tutorials och fungerar på samma sätt i praktiken (Suddaby, 2012); skillnaden ligger i hur informationen presenteras för spelaren. En vanlig kontextuell tutorial använder ofta små meddelanden eller bilder som inte tar mycket fokus, medan en tematiskt relevant sådan introducerar informationen sömlöst i spelvärlden. Ett exempel på detta är tutorialen för att hoppa i *Spelunky* (2008, se Figur 3), som visas genom en målning på väggen vid det första stället som spelaren behöver hoppa.

En brist som tematiskt relevanta kontextuella tutorials har är att de endast kan användas om en mekanik går att förklara på ett enkelt sätt, då abstrakta mekaniker som inte har direkta kopplingar till verkligheten kan vara svåra att förklara kortfattat i spelvärlden utan att det känns krystat (Suddaby, 2012). Men för övrigt strävar de flesta moderna tutorials efter att skapa så intuitiva och välintegrerade tutorials som möjligt, och då är denna metod en av de bästa.

Att lista ut hur ett spel fungerar kan vara del av det roliga, så det är inte alltid nödvändigt för ett spel att ha en tutorial (Suddaby, 2012). Ett spel utan tutorial måste vara lätt för den tänkta målgruppen att förstå, då mängden erfarenhet som krävs för att förstå hur ett spel fungerar ökar i takt med hur komplext det är. Det är viktigt för speldesigners att förstå sig på sin målgrupp, då det är svårt att göra en tutorial som är tillräckligt lättförstådd för nya spelare utan att få erfarna spelare att känna sig dumförklarade (Suddaby, 2012; Hedges, 2017). Tutorials används för att lära ut grundläggande mekaniker och detaljer till spelaren, men ofta innehåller spel mer komplexitet än vad tutorialen visar. Att förstå den tänkta målgruppens erfarenhetsnivå är centralt för att kunna utforma bra spel med fungerande tutorials.

2.3 Spelerfarenhet

Erfarenhet är enligt SAOL (2015) "kunskaper och färdigheter erhållna genom något som man varit med om". Människor blir utsatta för fler situationer och blir därmed mer erfarna när de spenderar tid på en aktivitet, vilket utökar deras kunskap och tillåter dem att prestera bättre med aktiviteten i framtiden. En person som är mycket erfaren med en given uppgift kommer överlag att prestera bättre när hen utför den uppgiften än en icke-erfaren person (McDaniel, Schmidt & Hunter, 1988), och spel är inte undantagna från denna regel. Det finns ett flertal studier inom spelvetenskap som berör erfarenhet och dess inverkan på hur spelare spelar.

Blumberg, Rosenthal och Randall (2007) utförde en studie som berör hur problemlösningsförmågan skiljer sig åt mellan spelare av olika erfarenhetsnivå när de stöter på *impasses* i spel. De lät sina testdeltagare spela *Sonic the Hedgehog 2* (1992) och bad dem att tänka högt medan de spelade, med syftet att förstå hur deltagarna tänkte när de stötte på *impasses*. Studien kom fram till att erfarna spelare var mer uppmärksamma på när *impasses* uppstod och evaluerade sina strategier snabbare än oerfarna spelare, som upplevde mer frustration då de inte var lika bra på att komma på framgångsrika strategier.

Elias, Garfield och Gutschera (2012) skriver om spelerfarenhet i ett kapitel om heuristiker, alltså tumregler som spelare tar hjälp av när de fattar beslut i ett spel. En heuristik är en metod för problemlösning som grundas i praktiska, användbara strategier hellre än teoretiskt optimala lösningar, och används av människor för att snabbt bedöma vad som behöver göras näst för att klara av en uppgift. Hjärnan har inte kapaciteten att bearbeta alla variabler och hela kontexten i en given situation, så därför skapas heuristiker för att begränsa mängden alternativ som finns tillgängliga.

Det finns heuristiker av olika nivåer som uppstår i takt med att en spelare blir mer erfaren med hur ett spel fungerar. Elias, Garfield och Gutschera (2012) kallar dessa för låg-, mellan- och högnivåheuristiker, och beskriver att heuristiker av högre nivå innefattar avancerade strategier medan lågnivåheuristiker är strategier som nybörjare kan använda. Ett exempel på en lågnivåheuristik kan vara att alltid attackera motståndarens pjäser när det är möjligt för att nå fram till kungen i *Schack* (u.å.). Det är inte en speciellt effektiv strategi, men det är lätt att förstå tanken bakom strategin och det låter spelaren veta vad hen borde försöka göra med varje drag. Ett exempel på en heuristik av högre nivå kan vara att analysera motståndarens drag för att förstå deras strategi, och sedan göra egna drag som slår den strategin. Detta är en heuristik som inte skulle fungera utan erfarenhet, då det är svårt att förstå motståndarens strategi om man aldrig stött på den tidigare.

Erfarenhet av ett spel är inte den enda faktorn som tillåter en spelare att skapa heuristiker av högre nivå, utan saker som talang och problemlösningsförmåga kan också påverka vilken sorts heuristiker en spelare använder (Elias, Garfield och Gutschera, 2012). Det är nivån av förståelse som spelaren har för spelet som bestämmer deras heuristiker, vilket kan variera även mellan personer med liknande erfarenhet.

Erfarenhetens inverkan på spelstilar som Elias, Garfield och Gutschera (2012) tar upp stöds av en studie utförd av Hong och Liu (2003). De lät 76 grundskoleelever spela spelet *Klotski* (u.å.), ett pusselspel där spelaren måste flytta på lådor i ett litet rum för att kunna flytta en större låda ut ur rummet, och undersökte hur spelarna tänkte med hjälp av think-aloud metoden. Deltagarna grupperades baserat på deras resultat: de som löste pusslet snabbast med minst antal drag delades in i expertgruppen och de som löste pusslet långsammast och

med flest antal drag hamnade i nybörjargruppen. Hong och Liu analyserade därefter tankestrategierna som spelarna använde och delade in dem i tre grupper: trial-and-error-strategin, heuristik-strategin och analogt tänkande-strategin.

Trial-and-error-strategin går ut på att välja ett drag på måfå, utföra det, och göra om processen flera gånger utan att lära sig från resultaten (Hong & Liu, 2003). Heuristik-strategin är som trial-and-error-strategin, men innebär också att spelare lär sig av resultaten för att inte göra samma misstag flera gånger. Analogt-tänkande-strategin går ut på att spelaren analyserar brädet för att försöka hitta det bästa draget innan de bestämmer sig för vad de ska göra. 42,4% av nybörjarspelarna använde trial-and-error, 37% använde heuristik-strategin och 20,6% använde analogt tänkande. Bland expertspelarna använde 16% trial-and-error, 35,5% heuristik-strategin och 48,5% analogt tänkande. Hong och Liu drar slutsatsen att spelare som presterade bättre tänkte mer på sina handlingar. Då prestanda är kopplat till erfarenhet (McDaniel, Schmidt & Hunter, 1988) kan detta tyda på att erfarna spelare använder sig av mer avancerade spelstilar.

Viktigt att nämna är att heuristik-strategin som Hong och Liu (2003) refererar till inte är samma sak som de heuristiker Elias, Garfield och Gutschera (2012) skriver om, trots att namnen liknar varandra. Elias, Garfield och Gutschera menar att en heuristik är en praktisk strategi som spelare lätt kan använda sig av för att fatta informerade beslut i ett spel. Heuristik-strategin som Hong och Liu (2003) skriver om betyder att spelare lär sig av sina misstag under trial-and-error, vilket är mycket mer begränsande och inte har något med specifika spelstilar att göra. Anledningen till att Hong och Liu valde att kalla det heuristik-strategin är antagligen för att det är en strategi fungerar likt hur heuristiker bildas; spelare lär sig av sina erfarenheter för att komma på bättre strategier. I det här fallet är det dock bara en enstaka strategi som utvecklas, och dess enda syfte är att undvika liknande fel.

Heuristiker som Elias, Garfield och Gutschera (2012) skriver om är inte begränsade till spel, och kan därför bildas för vilken aktivitet som helst. Detta medför att man kan utveckla heuristiker om själva spelandet i sig, inte bara om spelen som man spelar. Därför kan en person som är erfaren med att spela spel bland annat vara bättre på att använda kontroller, navigera menyer och hitta mekaniker. Spel i samma genre delar ofta många igenkänningsbara drag med varandra (Burn & Carr, 2006) vilket skapar ännu bättre förutsättningar för erfarenheter och förmågor att föras över mellan spel. Ett exempel på detta är förmågan att sikta i first-person-shooter-spel, som alltid implementeras med samma kontroller i moderna spel med ytterst få undantag. Andrew Dotsenko (2017) förklarar att detta görs då det är fördelaktigt för speldesigners att följa etablerade kontrollkonventioner, alltså att använda samma knappar som andra spel brukar använda för liknande funktioner. Det tillåter nämligen spelare med erfarenhet att använda tidigare kunskap för att förstå spelet, vilket i sin tur låter dem bilda heuristiker av högre nivå snabbare. I de fall där det inte finns några konventioner att följa föreslår Dotsenko (2017) att spel bör efterlikna verkligheten när det är möjligt, för att göra det så lätt som möjligt för spelarna att förstå hur mekaniker och funktioner fungerar i spelet. Om man bryter mot spelarens förväntningar löper man risken att oförväntade *impasses* och frustration uppstår.

3 Problemformulering

På grund av att speldesigners ofta strävar efter att använda sig av standardiserade kontrollkonventioner (Dotsenko, 2017) och att spel inom samma genre har liknande karaktärsdrag (Burn & Carr, 2006) är det troligt att erfarenheter som spelare bygger upp i ett spel kan appliceras i andra spel också. Speldesigners måste hålla detta i åtanke vid tutorial-design eftersom en bra tutorial inte bör tråka ut eller dumförklara spelaren (Hedges, 2017), vilket kan vara lättare sagt än gjort då ett koncept som är helt nytt för en icke-erfaren spelare kanske är en självklarhet för en erfaren spelare. Det bästa sättet att lära en icke-erfaren spelare att spela är därför inte nödvändigtvis den bästa metoden för en erfaren spelare.

Det finns tydliga skillnader i hur erfarna och icke-erfarna spelare spelar (Blumberg, Rosenthal & Randall, 2008), men tidigare forskning har inte undersökt huruvida dessa skillnader påverkar strategier som används vid inlärning.

VanLehn (1988) skriver om *Repair* och *External Information Retrieval* (*EIR*), som är två strategier där skillnader möjligtvis kan uppstå. Effektiviteten av *Repair* grundas i kunskap och gynnar därför erfarna spelare som har klarat av fler utmaningar, vilket enligt VanLehn (1988) sänker risken att buggar - alltså missuppfattningar och felbedömningar i den nya kunskapen - uppstår. Han menar att *EIR* inte har direkta kopplingar till erfarenhet som *Repair* gör, men det faktum att erfarna spelare är bättre på att utnyttja *Repair* skulle kunna påverka hur ofta *EIR* används.

Syftet med den här studien är att undersöka hur erfarna och icke-erfarna spelare använder sig av *Repair* och *External Information Retrieval*, med avsikt att hitta potentiella mönster och skillnader i hur erfarna och icke-erfarna spelare löser *impasses*. Förhoppningsvis kan arbetet ge ökad förståelse för skillnaden i tankesätt mellan spelare av olika erfarenhetsgrad, vilket kan hjälpa speldesigners med att ge bättre stöd till spelare som fastnar vid *impasses*. Frågeställningen är som följer:

> *Hur används strategierna Repair och External Information Retrieval av spelare av olika erfarenhetsgrad för att tackla impasse-drivna inlärningsmoment i actionplattformspel?*

Vi har avgränsat studien på två olika sätt för att begränsa studiens omfång. Den första avgränsningen är att vi endast undersöker *impasse*-drivna inlärningsmoment, både för att de ofta uppstår i spel, och även för att *impasse*-driven inlärning lägger fokus på spelarens personliga resonemang och problemlösningsförmåga vilket gör det lättare att undersöka skillnader i tankesätt mellan spelare.

För det andra har vi endast resurserna för att testa ett spel, så vi valde att skapa en prototyp inom actionplattform-genren då vi är mycket erfarna med dess konventioner. Det tillåter oss att skapa *impasses* för både nya och erfarna spelare, eftersom vi medvetet kan undvika genrekonventioner när vi skapar mekaniker.

På grund av spelprototypens genre kommer resultaten huvudsakligen vara relevanta för spel i actionplattform-genren. Däremot är vår förhoppning att studiens resultat kan användas för att underbygga diskussioner och vidare forskning inom ämnet tutorial-design i fler genrer.

4 Metodbeskrivning

4.1 Genomförande

Denna studie bestod av tre delar: i den första delen intervjuades deltagarna om deras erfarenhetsnivå, i den andra delen fick deltagarna spela spelet och i den sista delen intervjuades deltagarna om deras spelupplevelse. Resultaten av studien analyserades efter datainsamlingen i ett försök att besvara frågeställningen. Testdeltagarna hittades på campus vid högskolan i Skövde, och där utfördes även undersökningen av bekvämlighetsskäl. Varje deltagare fick spela spelet med minst en testdeltagare närvarande för att förklara syftet och målet med studien, samt för att se till att inga problem uppstod under genomförandet.

Innan den första intervjun blev deltagarna informerade om att hela spelsessionen skulle spelas in och att ingen information som kunde användas för att identifiera dem skulle sparas. De försäkrades även om att endast testledarna skulle ha tillgång till det inspelade materialet, med syftet att bevara deltagarnas anonymitet och inte inkräkta på deras personliga integritet.

Studiens första del var som tidigare nämnt en intervju med syftet att ta reda på hur erfaren varje deltagare var. Först fick deltagarna med egna ord beskriva hur erfarna de var med spel generellt sett, och sedan fick de berätta hur erfarna de var med actionplattformspel. Om en deltagare inte visste vad ett actionplattformspel är så förklarade en testledare att det är en spelgenre där man slåss och hoppar mellan plattformar. Den egenskapta spelprototypen visades även som exempel på ett actionplattformspel. Testdeltagarna ombads därefter att uppskatta hur många timmar de spelade per vecka, och hur mycket de spelat som mest tidigare. Intervjuerna spelades in och transkriberades, och analyserades senare för att ge en uppfattning av hur erfaren varje deltagare var.

Efter den första intervjun fick testdeltagarna spelet förklarat för sig. Först fick de veta grundläggande information om spelets struktur, alltså att det består av sju banor med hinder och att deras mål var att förstöra måltavlorna i varje bana. Alla deltagare fick samma information för att se till att ingen hade bättre eller sämre förutsättningar. De blev sedan informerade om att de behövde lista ut hur de skulle nå måltavlan i varje bana, och att de kunde använda mekaniklistan om de ville veta hur en mekanik fungerade. Deltagarna fick även reda på att de kunde be testledarna om hjälp om de inte kunde klara av en bana, men att det var tänkt som en sista utväg för att studien inte skulle fastna om en deltagare inte klarade av en bana.

Studiens andra del bestod av att deltagarna spelade igenom spelprototypen. De uppmanades att förklara hur de tänkte enligt Think Aloud-metoden (Desurvire & El-Nasr, 2013) medan de spelade, och spelsessionerna spelades in för att möjliggöra analys vid senare tillfälle. Om en testdeltagare inte sa något på ungefär tio sekunder så blev de ombedda att säga vad de tänkte för att säkerställa att testdeltagaren fortsatte förklara sin tankeprocess. Spelsessionerna resulterade i data som bland annat kunde visa vilken strategi deltagarna valde, vilka *impasses* som var svårast för dem att lösa samt vilka buggar som uppstod i deras kunskap. Spelsessionerna gav även en liten mängd kvantitativa data, som hur lång tid det tog för varje deltagare att klara varje bana.

Den tredje delen av studien var en intervju som handlade om testdeltagarnas spelupplevelse. En testledare öppnade mekaniklistan och deltagarna fick säga om de kände igen någon av

spelets mekaniker sedan tidigare, då detta skulle kunna förklara möjliga missförstånd som uppstått under spelsessionen. Om en deltagare kände igen en mekanik sedan tidigare skulle det kunna ge en förklaring till varför deltagaren spelade som hen gjorde, då det var sannolikt att mer erfarna deltagare skulle testa saker som de stött på i andra spel. Deltagarna fick även berätta vilka mekaniker de hade svårast att förstå, då detta möjligtvis skulle kunna indikera vart buggar uppstått.

I analysen grupperades deltagarna efter sina erfarenhetsbeskrivningar för att göra det lättare att jämföra hur spelare av liknande erfarenhetsnivå spelade. Testledarna observerade inspelningarna och antecknade när en deltagare gjorde eller sade något intressant, vilket sedan användes för att dra slutsatser.

4.2 Prototyp

Prototypen är ett actionplattformspel med sju banor, där spelarens mål är att slå sönder måltavlor i varje bana. För att *impasse*-driven inlärning skall uppstå måste spelaren möta en *impasse* och försöka lösa den, vilket är en svår situation att skapa om det finns ett sätt för spelaren att förbigå hindret. Därför var spelet linjärt; banorna kan endast spelas i en ordning. Det är lättare att observera skillnader i strategival om det finns färre faktorer som kan påverka spelarnas beslut.

Varje bana innehåller minst ett *impasse*-drivet inlärningsmoment. Spelaren måste lista ut vilka mekaniker de bör använda, samt hur mekanikerna fungerar för att nå måltavlorna i varje bana. Spelet har en inbyggd mekaniklista med videodemonstrationer för att möjliggöra *EIR*, då spelaren kan öppna listan när som helst för att söka efter extern information om mekanikerna. Spelsessionen är över när spelarna slår sönder den sista måltavlan i den sista banan.

4.3 Metoddiskussion

Eftersom vi valde att leta efter testdeltagare på Högskolan i Skövdes campus så var det större sannolikhet att testdeltagarna skulle vara unga vuxna. Vi letade på högskolan på grund av två huvudsakliga anledningar: för det första var platsen väl anpassad för studien, då det fanns avskilda, ljuddämpade rum som underlättade inspelningarna av spelomgångarna. För det andra är det troligare att en ung spelare är seriös i sitt spelande än äldre spelare, då de generellt sett har mer fritid, vilket innebär att det var lättare att hitta unga erfarna spelare än äldre. Det finns en risk att testgruppen blivit homogen då alla deltagare hittades på högskolan i Skövde, men att motverka denna risk hade krävt mer resurser och tid än vad vi har tillgängligt.

Anledningen till att vi använde självskattning i den första intervjun är för att erfarenhet inte är kvantifierbart, så det är omöjligt att sätta en icke-arbiträr gräns för vem som är erfaren och vem som inte är det. Vi lät testdeltagarna förklara hur erfarna de var med egna ord eftersom vi tänkte att det skulle vara lättast för dem att göra. Vi kunde ha låtit deltagarna bedöma sig själva med en skala, men vi valde att använda självskattning eftersom vi ville ha en så detaljerad beskrivning av deras erfarenhet som möjligt. Deltagarna fick även motivera sina självskattningar och förklara sina svar, vilket gjorde det lättare att jämföra dem med varandra. Frågorna om hur många timmar de spelade i veckan var till för att kunna sätta deras självskattningar i kontext.

Testdeltagarna blev informerade om att deras mål i spelet var att lista ut mekanikerna och att de kunde ta hjälp av mekaniklistan för detta, men vi sade inte explicit att de kunde använda sig av trial-and-error för att klara hindren. Vårt resonemang var man alltid kan prova sig fram i spel och att det därför skulle vara uppenbart för spelarna, och vi ville inte påverka deras strategival genom att antyda att trial-and-error var en speciell strategi i det här spelet. Vi ville dock försäkra oss om att ingen trodde att mekaniklistan var det enda alternativet som de hade, så vi sade ordagrant till deltagarna att de behövde "lista ut hur spelets mekaniker fungerar". Enligt vår mening antyder den formuleringen att det går att hitta lösningen på egen hand genom att testa sig fram.

Vi gav testdeltagarna alternativet att fråga oss om lösningen på en *impasse* om de verkligen fastnade, då vi ville undvika att testet inte kunde fortsätta om en spelare gav upp vid en *impasse*. Ett problem med detta är att spelarna inte behöver resonera själva om vi ger dem lösningen, vilket skulle göra det svårare att analysera deras strategival. Risken finns också att det skulle vara snabbare och lättare för spelarna att fråga oss än att försöka klara av *impasses* själva, vilket skulle göra det svårt att analysera strategival och göra resultaten oanvändbara. Østbye m.fl. (2003) skriver att testdeltagare ofta gör vad de tror att testledaren vill se, så vi bad deltagarna att försöka klara hindren på egen hand utan att ge upp för att låta dem veta vad vi ville se.

Ett alternativ till att berätta lösningen för spelaren hade varit att låta dem skippa en bana om de inte klarar en *impasse*, men då skulle de inte lära sig lösningen. Detta skulle vara ett problem eftersom flera av banorna bygger på kunskap som byggs upp genom hela spelet, så en *impasse* som spelaren gav upp på skulle kunna uppstå igen senare. Vi ville undvika detta då det skulle vara svårare att jämföra deltagarnas resultat med varandra om deras *impasses* uppstod på olika ställen.

Vi valde att använda ett kvalitativt förhållningssätt då vi har en relativt liten testgrupp, och vi undersöker ämnen som är svåra att kvantifiera. Vi väljer till exempel att observera hur deltagarna spelar för att kunna dra slutsatser om hur de väljer strategi, vilket är ett bra sätt att undersöka undermedvetna handlingar enligt Bryman (2018). Intervjuerna är semi-strukturerade vilket innebär att frågorna är flexibla och tillåter djupgående svar, samt förtydliganden som kan förhindra missförstånd och ge djupare förståelse för hur deltagarna tänker (Østbye m.fl, 2003).

Think-aloud-metoden har vid flera tillfällen använts inom spelforskning för att tydliggöra spelares tankeprocesser under studiernas gång (Blumberg, Rosenthal & Randall, 2007; Hong & Liu, 2003). Desurvire och El-Nasr (2013) menar att think-aloud metoden är en snabb och effektiv metod för datainsamling i kontexten av spelutveckling. Datan som produceras av think-aloud metoden är, enligt Cotton och Gresty (2006), "rich and exceptionally revealing". Detta medför att think-aloud metoden kan ge en stor mängd information från ett relativt litet dataset, vilket är fördelaktigt för en studie som denna.

5 Projektbeskrivning

5.1 Spelbeskrivning

Spelprototypen skapades med hjälp av spelmotorn *Unity* (2018), av anledningen att vi har haft möjlighet att arbeta i den sedan tidigare. All kod och grafik skapades specifikt för att använda i prototypen, med undantag för knapp-symbolerna i spelets tutorial som hämtades från en sida med gratis-assets på webben. Spelet är designat för att spelas med en Xbox 360-handkontroll. Syftet med detta är att begränsa mängden knappkombinationer som spelarna kan testa, då tangentbord har väldigt många fler knappar än vad handkontroller har. Spelet fungerar endast till PC.

Spelet är ett actionplattformspel med pusselelement. Genren actionplattformspel är en kombination av två olika genrer: action och plattformspel. Plattformspel definieras av att spelarkaraktären måste hoppa över olika hinder och ta sig igenom en bana för att klara spelet. Action-delen av termen innebär att det finns mekaniker i spelet som används för att strida, och med pusselelement menas att det finns delar i spelet som måste lösas med logiskt resonemang, vilket inkluderades för att underlätta skapandet av *impasses*.

Spelet har sex olika mekaniker som används för att skapa *impasse*-drivna inlärningsmoment för spelaren, vissa knutna till spelarkaraktärens rörelsesystem, andra till dess förmågor och några som påverkar spelvärlden omkring spelaren. Spelets kontroller är baserade på Fitt's Law (Dotsenko, 2017), med ett fåtal undantag som vi gjort medvetet för att bryta mot förväntningar och skapa *impasses* för erfarna spelare. Spelet har sju banor; sex stycken vars syfte är att lära spelaren hur varje mekanik fungerar genom *impasse*-driven inlärning, och en i slutet av spelet som implementerar spelets mekaniker i kombination med varandra. Spelet använder sig av en mycket stilren grafisk stil, delvis på grund av bristande resurser och tid, men främst för att undvika att spelets grafik för bort spelarens fokus från resterande spelelement. Spelet innehåller en mekaniklista som kortfattat beskriver spelets mekaniker för att låta spelarna använda *EIR*.

5.1.1 Spelmekaniker

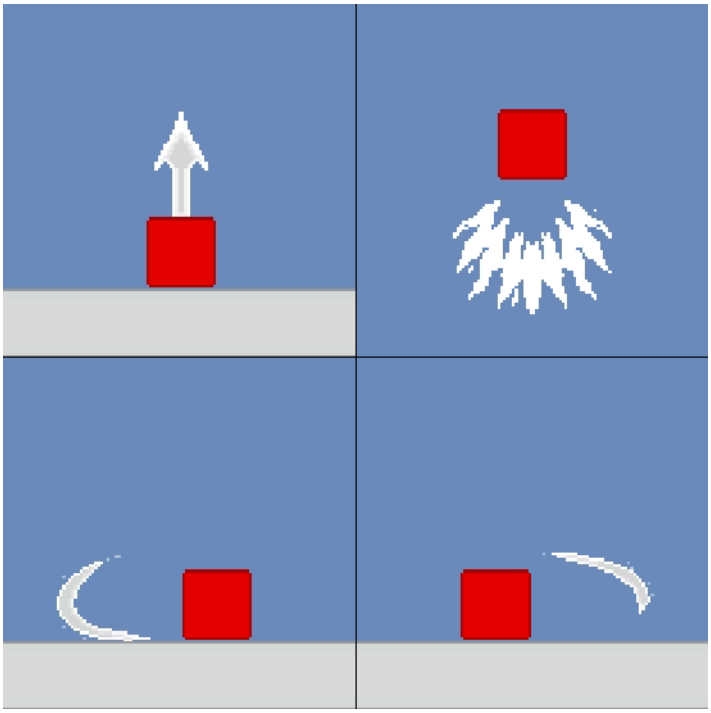
Det finns ett antal mekaniker i spelet, vissa som är designade för att kunna kännas igen av erfarna spelare, och vissa som är ämnade att vara främmande för både nya och erfarna spelare. Nedan följer en beskrivning av varje mekanik och deras tänkta syfte, samt resonemangen bakom att inkludera just de mekanikerna.

Rörelsesystem och Sprinting

Spelets grundläggande rörelsesystem består av genretypiska rörelse- och hoppmekaniker. Spelet innehåller även en *Sprint*-mekanik som ökar spelarens hastighet när de håller inne RB-knappen. Anledningen till att vi valde RB-knappen för *Sprint*-funktionen hellre än en knapp på framsidan av kontrollen var för att spelaren enkelt skulle kunna använda *Sprint* och hoppa samtidigt som i *Super Meat Boy* (2010), som använder denna knapplayout för att undvika att spelare blir trötta i fingrarna av att hålla inne *Sprint*-knappen under långa tidsperioder. Spelaren kan röra sig åt höger och vänster genom att dra den vänstra spaken åt sidorna, samt hoppa genom att trycka på A-knappen på spelkontrollen. Spelaren kan variera höjden av sitt hopp genom att släppa hoppknappen tidigt eller sent in i ett hopp, vilket är en standard inom nästan alla former av platformer-spel som ger spelaren ökad kontroll över sin karaktär.

Spelarkaraktärens grundläggande kontroller är inte tänkta att skapa *impasses*, då det skulle vara svårt att skapa mer avancerade *impasses* om alla spelare inte förstår spelets grunder. Därför är den första banan ett tutorial-rum vars syfte är att lära ut de grundläggande kontrollerna innan spelarna ställs inför svårare *impasses*.

Attacker



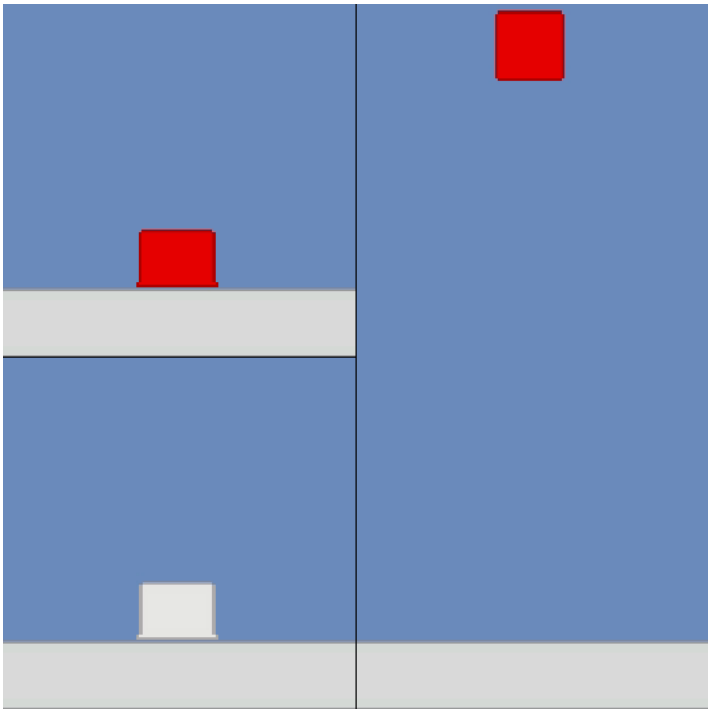
Figur 4 Spelarens attacker.

Spelaren har tillgång till tre attacker: en uppåt-attack som träffar måltavlor ovanför spelaren, en horisontell attack som kan skickas ut åt höger eller vänster och en nedåt-attack som knuffar spelaren uppåt (se Figur 4), men som endast kan användas i luften. Nedåt-attacken uppfyller samma funktion som ett dubbelhopp gör i genretypiska plattformspel och refereras till som *Downward Aerial* i det här spelet.

Vi fattade beslutet att slå ihop dubbelhoppet med en attack för att bryta mot erfarna spelares förväntningar och skapa mer komplexa *impasses*, då genretypiska plattformspel som *Hollow Knight* (2017) ofta knyter dubbelhopp-mekaniken till att trycka på hoppknappen när spelarkaraktären redan är i luften. Eftersom inga andra attacker har någon form av rekyl förväntades denna mekanik vara icke-intuitiv för spelare, då spelaren inte bör förvänta sig att en nedåtriktad attack kommer föra spelarkaraktären uppåt.

Med undantag för *Downward Aerial* förväntades attackerna vara lättförstådda av både erfarna och icke-erfarna spelare, eftersom attackerna är väldigt enkla och variationer kan hittas i nästan alla actionplattformspel.

Charge Jump



Figur 5 *Charge Jump* uppladdning och hopp.

Charge Jump är en variation på spelets hoppmekanik som tillåter spelaren att hoppa högre än vanligt. För att utföra ett *Charge Jump* behöver spelaren först stå stilla på marken, sedan hålla rörelsespaken nedåt tills spelarkaraktären börjar blinka vitt (Se figur 5), och därefter trycka på hoppknappen. Ett *Charge Jump* får spelaren att hoppa högre än ett vanligt hopp och en *Downward Aerial* gör i kombination, så uppladdade hopp är det bästa alternativet spelaren har för att nå höga höjder. *Charge Jumps* är dock inte ämnade att tillåta spelaren att hoppa långt i sidled, och därför är det inte möjligt att ladda upp ett hopp och röra sig samtidigt.

Förmågan att "ladda upp" ett hopp för att hoppa högre förekommer i många spel inom actionplattform-genren, exempelvis *Ori and the Blind Fores*t (2015) och *Super Mario Bros. 2* (1988). *Charge Jumps* förekommer inte lika ofta i spel som t.ex. dubbelhopp gör, så en av våra förväntningar med att inkludera den här mekaniken var att endast erfarna spelare skulle ha stött på den tidigare och att den skulle vara helt främmande för nya spelare.

Vi diskuterade idén att spelarkaraktären skulle kunna röra på sig medan de laddade upp ett hopp. Problemet med denna idé var att det uppladdningsbara hoppet hade haft för få nackdelar. Spelaren skulle kunna använda det även i horisontella hopp, vilket hade gjort det svårare att designa rörelse-baserade *impasses*. Därefter funderade vi på att tillåta spelkaraktären att röra sig långsamt medan hen laddade upp ett hopp, men den idén implementerades inte heller på grund av tekniska problem.

Clone

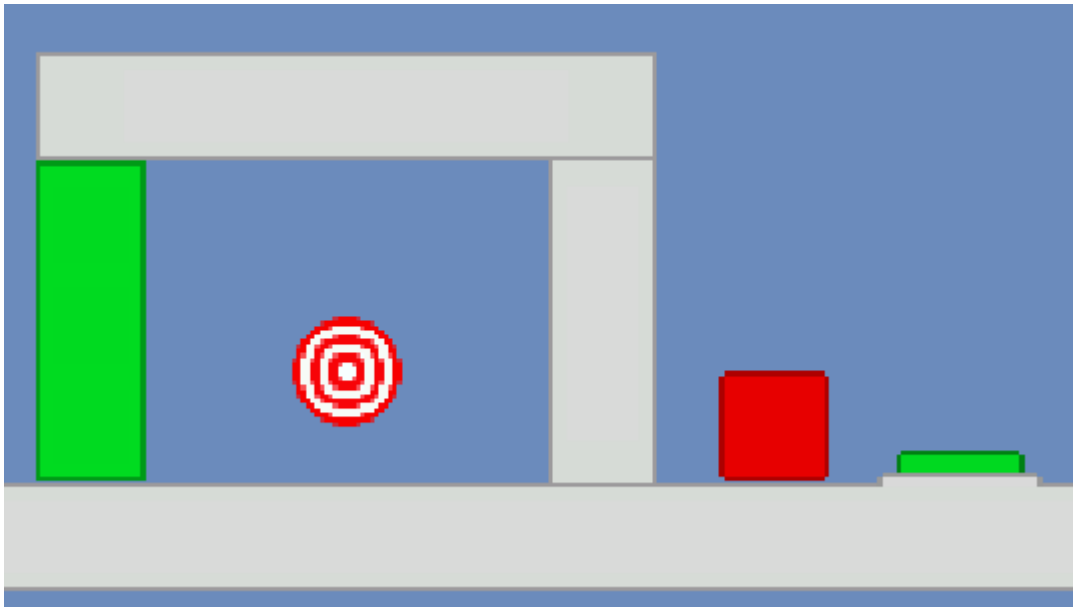
Figur 6 En klon som kopierar en attack.

Spelaren kan skapa en klon av spelarkaraktären genom att trycka på B. Klonen skapas på spelarkaraktärens position och är ett separat objekt som delar vissa egenskaper med spelarkaraktären. Spelaren kan inte flytta på klonen alls, men den påverkas av gravitation och den kan kopiera spelarens nästa attack (se Figur 6), varpå den försvinner. Om spelaren försöker skapa en klon när det redan finns en klon ute på spelfältet tas den gamla bort och ersätts med den nya.

Vi skapade *Clone*-mekaniken samtidigt som *Door*-mekaniken för att göra det möjligt att designa mer komplexa banor och *impasses*. *Clone*-mekaniken skapar inte *impasses* i sig, men den ger spelaren många fler alternativ att kombinera med de andra mekanikerna. Liknande mekaniker förekommer sällan inom actionplattform-genren enligt våra erfarenheter och förväntades därför vara främmande både nya och erfarna spelare.

Vi övervägde att låta spelaren styra klonen likt spelarkaraktären, men vi valde att inte göra det då spelaren hade fått problematiskt många alternativ vid varje *impasse*. Ju fler alternativ spelaren har, desto svårare blir det för hen att veta vilket alternativ som är rätt lösning. Vi funderade även på att låta spelarkaraktären och klonen interagera med varandra. Ett förslag var att spelarkaraktären skulle kunna ställa sig på klonen, men detta ansåg vi vara riskabelt då det skulle göra det svårare att designa höjd-relaterade *impasses*. Vi fattade beslutet att spelaren inte skulle kunna interagera med klonen alls för att hålla den lättförstådd. *Clone*-mekaniken saknar trots allt motsvarigheter i riktiga livet och är sällsynt i spel, vilket innebär att spelare kan missförstå dess funktionalitet om den görs för komplex.

Door-block och knappar



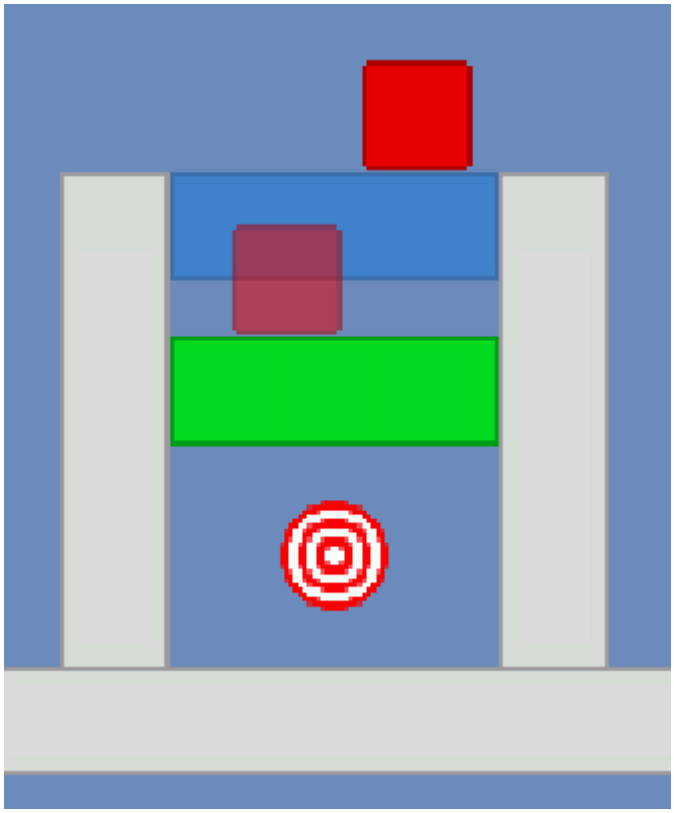
Figur 7 Ett *Door*-block med tillhörande knapp.

Door-block är gröna block som försvinner när spelaren ställer sig på en knapp med samma färg (se Figur 7). Knappen hålls nedtryckt tills det att karaktären flyttar sig bort från den, varpå blocket dyker upp igen. Kloner kan också tynga ner knappar, precis som spelarkaraktären. Liknande mekaniker förekommer väldigt ofta i spel, nästan oavsett genre, så de förväntades därför vara intuitiva för de flesta spelare. Färgkodningen mellan *Door*-blocken och knapparna gör att spelare naturligt drar en koppling mellan dem (Ware, 2008), vilket gör mekaniken mer intuitiv. *Door*-block kan användas i samband med *Clone*-mekaniken för att skapa *impasses*, då spelaren kan placera kloner på knappar för att hålla dem öppna.

Door-mekaniken lades till eftersom spelet behövde något som spelaren kunde interagera med och prova spelarkaraktärens förmågor på. Det faktum att *Door*-mekaniken är en intuitiv mekanik med motsvarigheter i verkligheten är en positiv faktor, då de flesta spelare kan förväntas förstå hur mekaniken fungerar snabbt (Dotsenko, 2017).

Vi funderade på att lägga till fler variationer av *Door*-mekaniken, men vi valde att inte göra det då det var högre prioriterat att designa banor istället. En variation som vi implementerade men aldrig använde var knappar som håller sig nedtryckta efter att spelaren kliver av dem, vilket skulle ge möjligheten att skapa mer komplexa pussel. Anledningen till att vi inte använde den här mekaniken var att den hade skapat pussel med många steg. Om en stor mängd steg krävs för att komma förbi en *impasse* tar studien längre tid att utföra, vilket kan påverka testdeltagarnas benägenhet att delta.

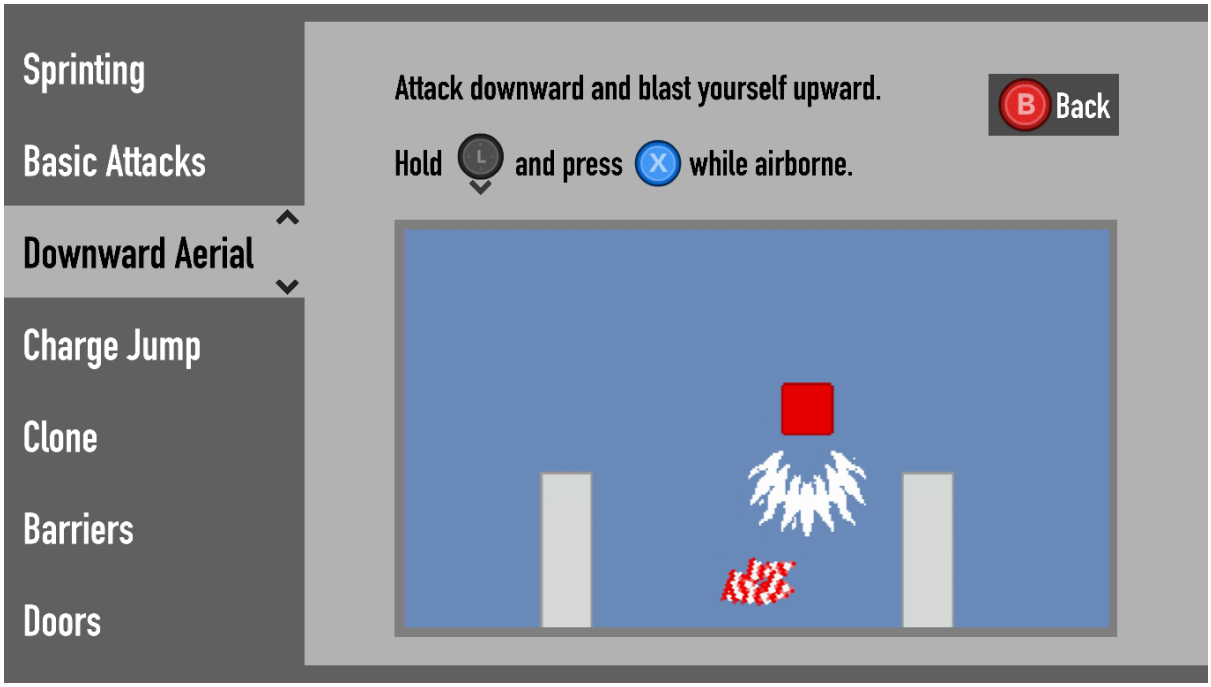
Barrier-block



Figur 8 *Barrier*-block med en klon inuti.

Barrier-block är block som endast spelaren kan stå på; kloner passerar igenom *Barrier*-block obehindrat (se Figur 8). Vi lade till dem för att ge *Clone*-mekaniken mer djup, då tidigare varianter av *Clone*-mekaniken endast var användbar för att hålla ned knappar. Med *Barrier*-block i spelet kunde vi designa *impasses* där måltavlor eller knappar endast kunde nås med hjälp av klonen. Vi förväntade oss att *Barrier*-block skulle vara både obekanta och lättförstådda för spelare oavsett erfarenhetsnivå eftersom deras funktionalitet är relativt unik, men saknar komplexitet; det finns bara ett sätt att interagera med dem.

5.1.2 Mekaniklista



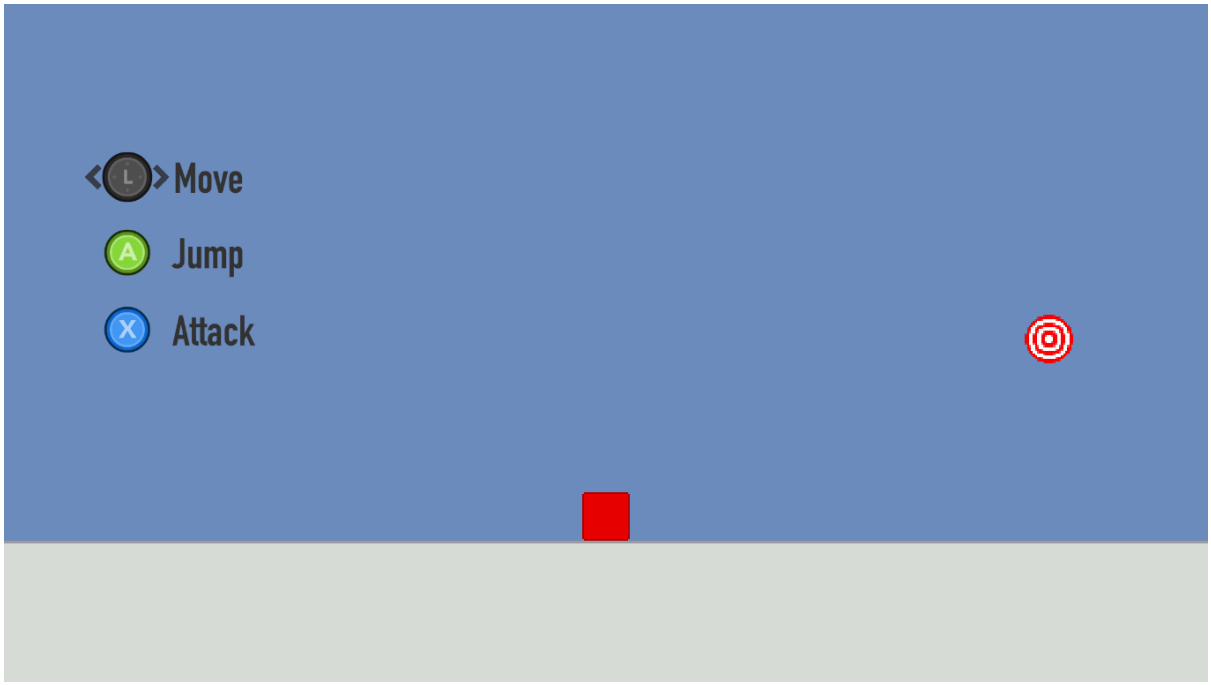
Figur 9 Utseendet av spelets mekaniklista.

Spelaren kan öppna mekaniklistan när som helst för att kolla upp information om spelets mekaniker, vilket tillåter deltagarna att använda *EIR* för att lösa *impasses*. Varje sida av mekaniklistan innehåller en kort beskrivning av varje mekanik, samt en förklaring om hur spelaren använder mekaniken själv. Utöver detta har varje sida ett videoexempel på hur spelmekaniken fungerar, för att visa mekanikerna hellre än att bara beskriva dem (se Figur 9). Det finns en översikt med alla mekaniker på den vänstra sidan av skärmen, som spelaren kan navigera mellan genom att flytta rörelsespaken eller trycka på upp och ner på styrkorset. Mekaniklistan refereras till som *Glossary* i spelets tutorials.

5.1.3 Banor

Nedan följer en bild och en kort beskrivning om varje bana i spelet, samt vilken eller vilka *impasses* som testas i varje bana.

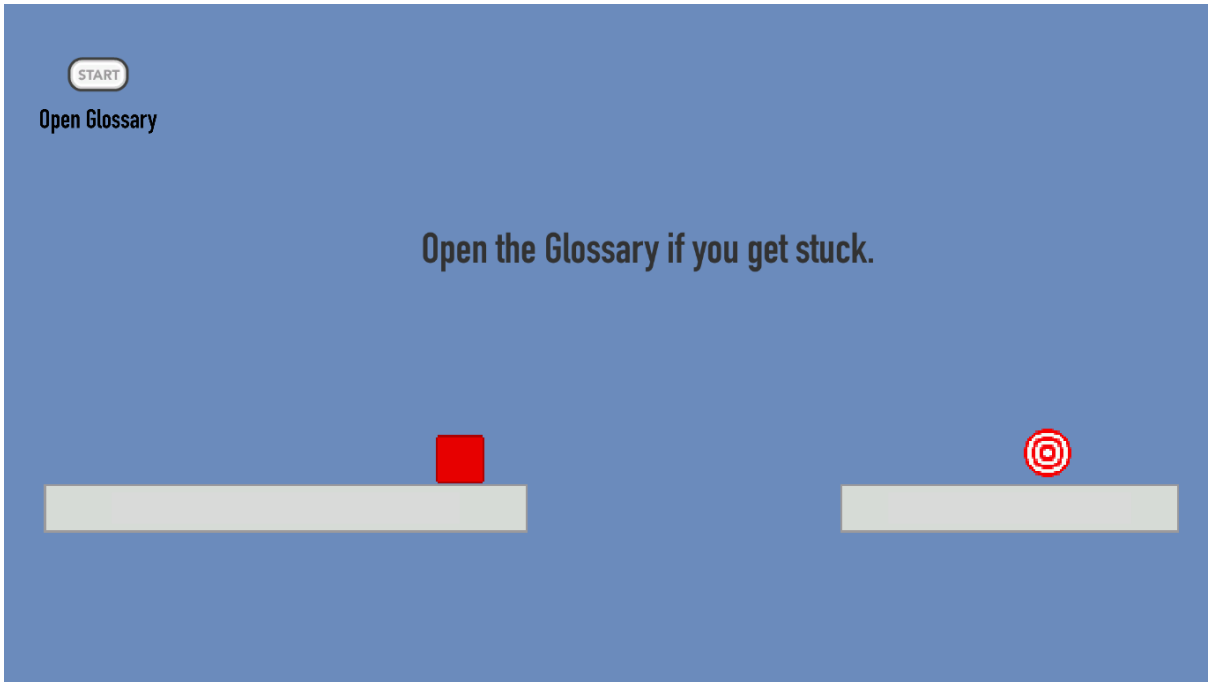
Bana 1: Tutorial



Figur 10 Tutorial-banan.

Den första banan är en tutorial för spelets grundläggande kontroller. Kontrollerna står i vänstra sidan av banan och banans måltavla står till höger (se Figur 10). För att slå måltavlan måste spelaren gå till höger, hoppa och slå, så det är omöjligt för spelaren att komma vidare utan att demonstrera att de förstår kontrollerna. Enligt definitionerna av Suddaby (2012) räknas detta som ett tutorial-rum, alltså en nivå där spelaren introduceras till spelets mekaniker innan spelet börjar på riktigt. Tutorialen är till för spelarens bekvämlighet och för att minska frustration innan spelet har börjat. Den anses därför vara nödvändig för spelbarheten av spelet.

Bana 2: Sprint

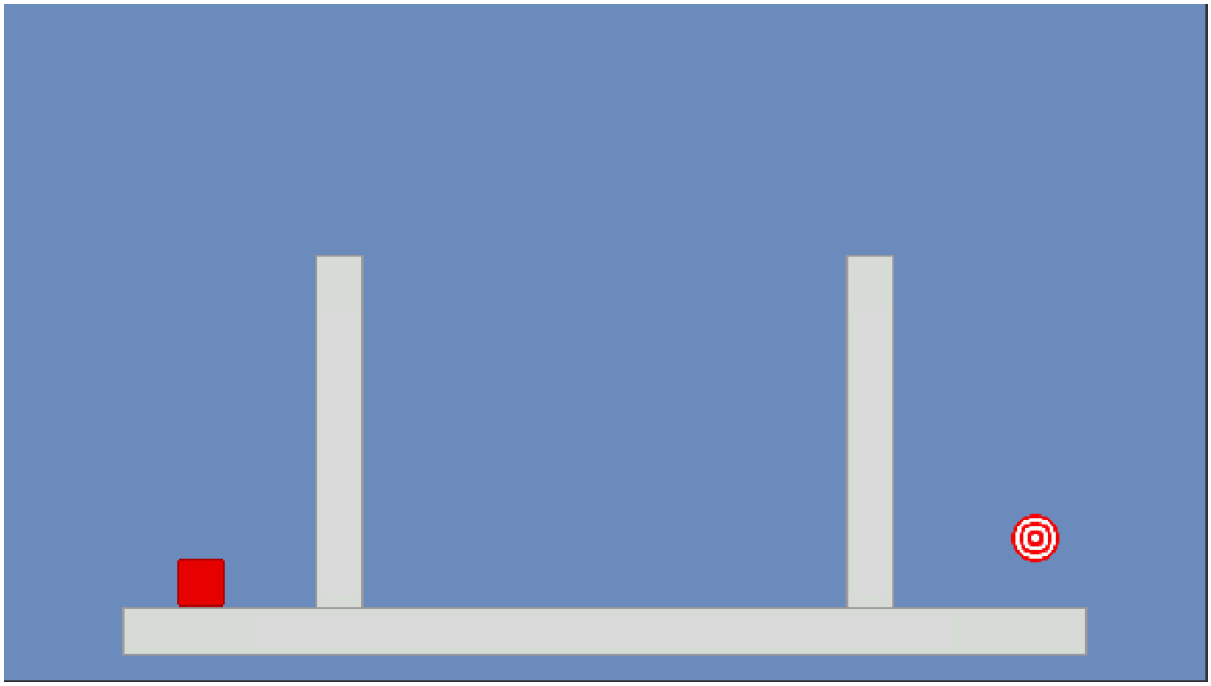


Figur 11 *Sprint*-banan.

Sprint-banan introducerar spelets första planerade *impasse*. Den består av två plattformar med ett stort hål mellan dem, och spelaren måste ta sig över hålet för att kunna slå måltavlan (se Figur 11). Avståndet är för långt för att spelaren ska kunna hoppa över det med ett vanligt hopp, så det är tänkt att spelaren ska använda *Sprint*-mekaniken för att hoppa längre. *Sprint*-mekaniken är den första mekaniken som visas i mekaniklistan, så spelaren förväntas öppna mekaniklistan och lära sig genom den. Alternativt kan spelaren hitta knappen själv genom *Repair*.

För att lära spelaren att de kan öppna mekaniklistan visas texten “Open the glossary if you get stuck” i mitten av banan. Det finns alltid ett tips längst upp till vänster på skärmen som påminner spelaren att detta görs genom att trycka på Start-knappen, och tack vare att banan är minimalistiskt designad finns det få element som kan dra fokus från detta.

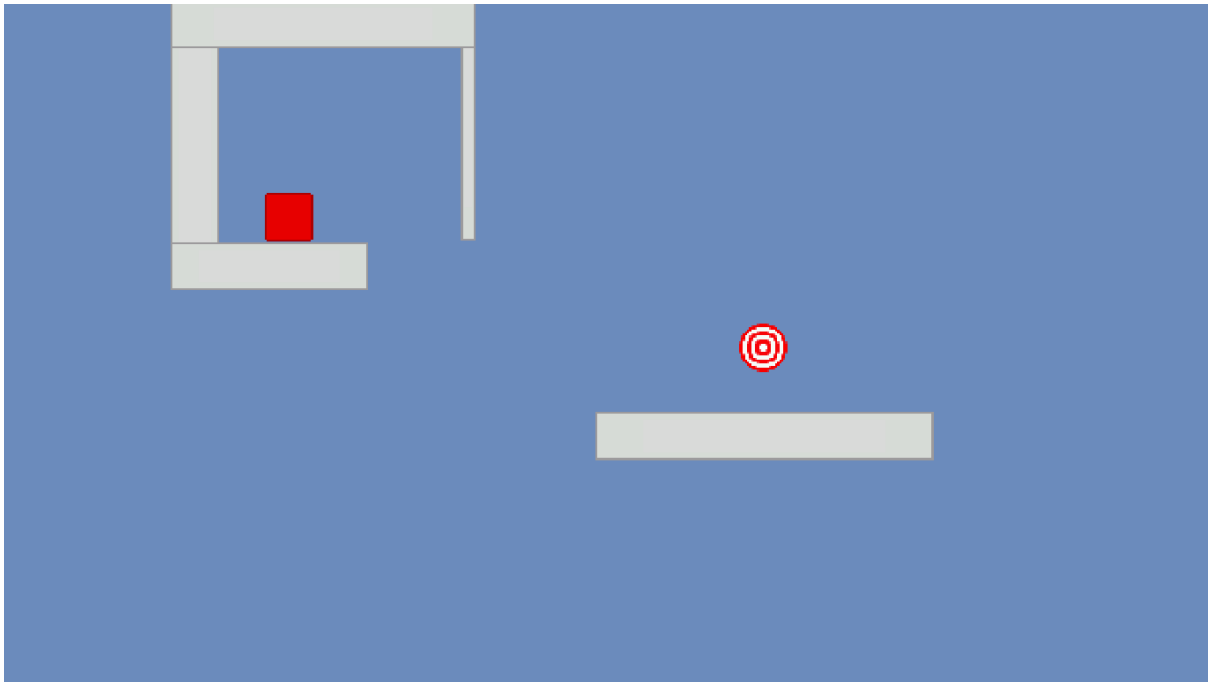
Bana 3: Charge Jump



Figur 12 *Charge Jump* banan.

I *Charge Jump*-banan möts spelaren av en *impasse* i formen av två höga pelare (se Figur 12). Det enda sättet att komma förbi denna *impasse* är att använda *Charge Jump*, då ett vanligt hopp och en *Downward Aerial* inte låter spelarkaraktären hoppa högt nog för att ta sig över pelarna.

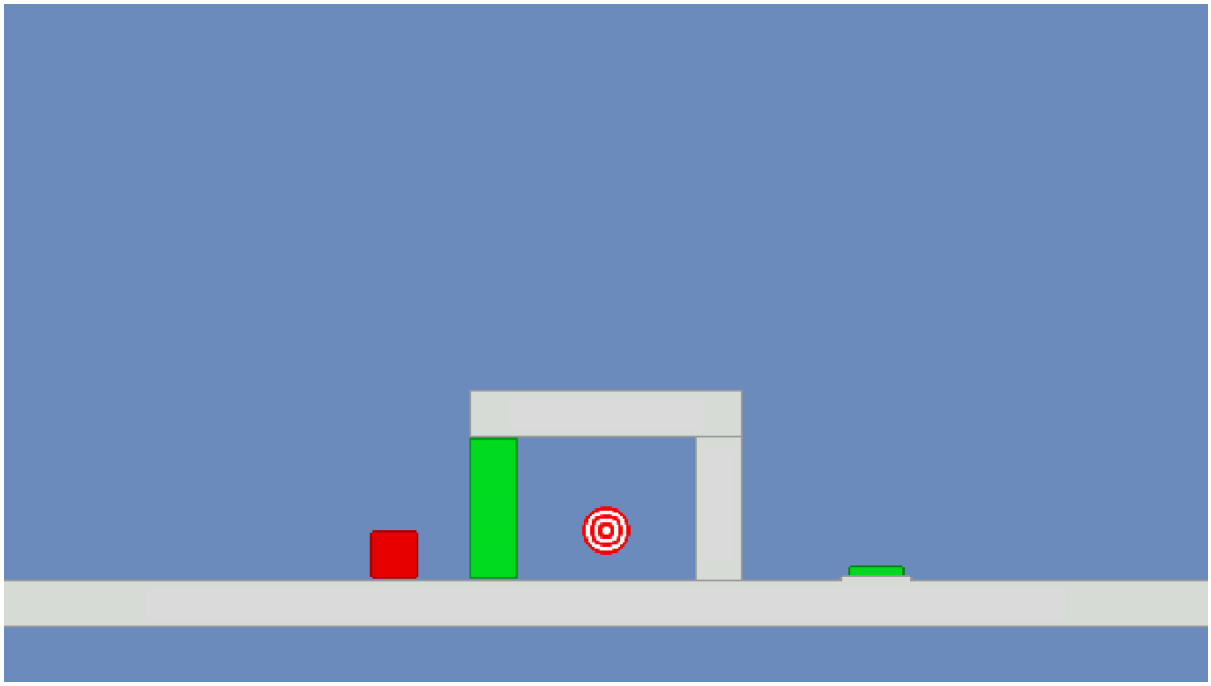
Bana 4: Downward Aerial



Figur 13 *Downward Aerial*-banan.

Den fjärde banan har en *impasse* som kräver att spelaren hoppar i luften. En tunn vägg blockerar spelaren från att hoppa till plattformen med måltavlan direkt, och den är placerad på ett sådant vis att det inte är möjligt att bara falla till plattformen (se Figur 13). Den enda lösningen på denär att använda sig av *Downward Aerial*-attacken för att studsa i luften. En alternativ design kunde ha varit att låta spelaren hoppa över en pelare som i *Charge Jump*-banan, fast högre den här gången för att kräva både ett *Charge Jump* och en *Downward Aerial* för att få mer höjd. Vi ville dock att banorna i regel skulle se annorlunda ut, för att göra det tydligt för spelaren att de skulle behöva använda olika mekaniker i varje bana.

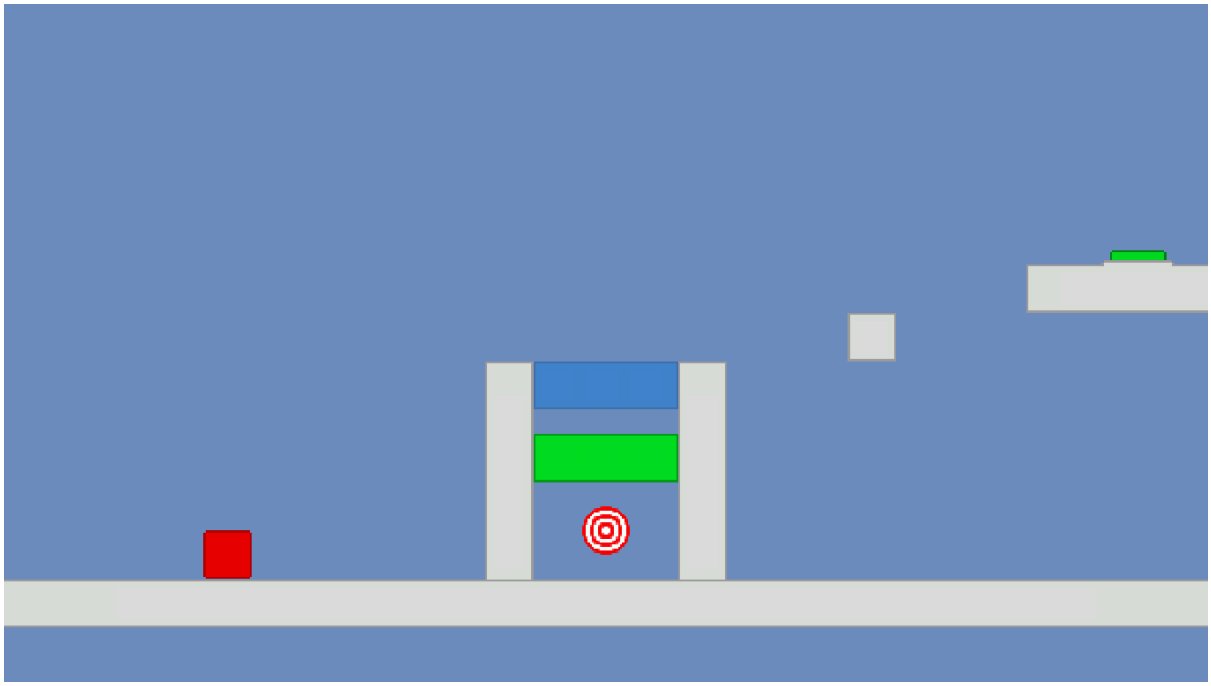
Bana 5: Door



Figur 14 *Door*-banan.

Den här banan består av en måltavla bakom ett *Door*-block och en knapp utanför som öppnar *Door*-blocket (se Figur 14). Spelaren kan inte interagera med något annat än knappen i början av banan, så de förväntas ställa sig på knappen och lära sig hur den fungerar. För att lösa denna *impasse* behöver spelaren ställa en klon på knappen för att hålla den nedtryckt. Återigen ser banan annorlunda ut från den förra och endast element som är relevanta för utmaningen visas. Det här är den enda banan som introducerar två mekaniker till spelaren samtidigt, vilket var nödvändigt då klonmekaniken inte kan användas till något på egen hand. Vi valde att visa klonens funktionalitet med hjälp av *Door*-mekaniken hellre än *Barrier*-mekaniken, då vi anser att *Door*-block är mer intuitiva än Barrier-block på grund av dess motsvarighet till verkligheten.

Bana 6: Barrier

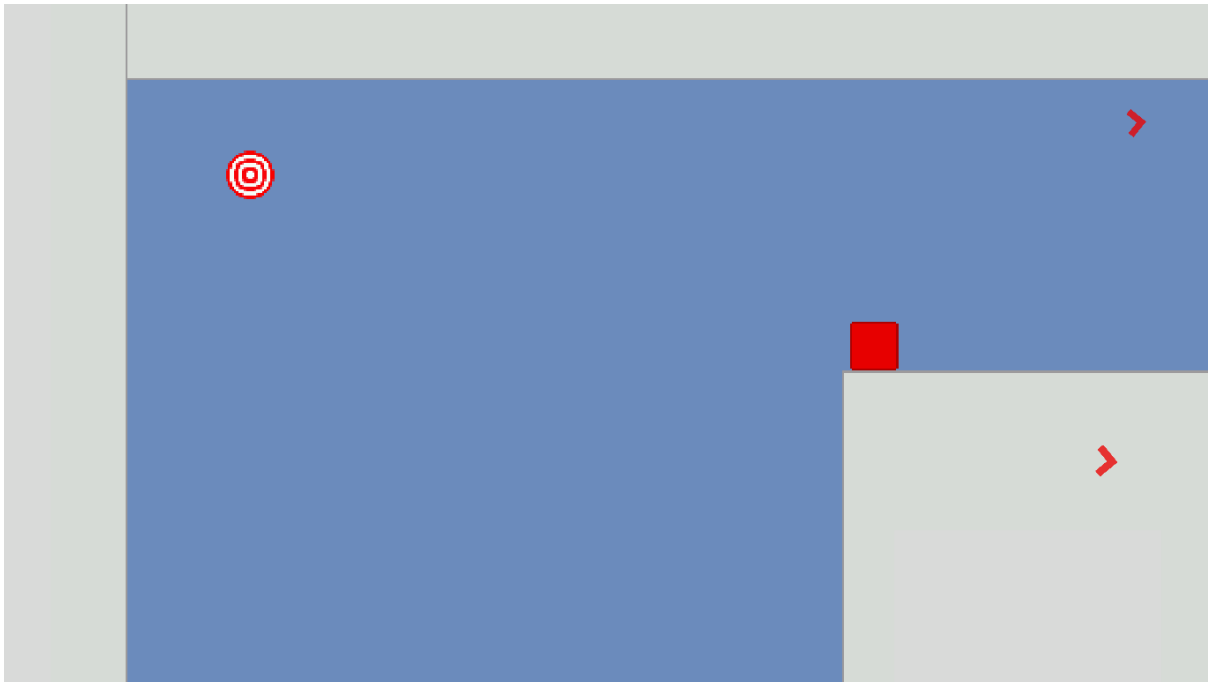


Figur 15 *Barrier*-banan.

Barrier-banan är väldigt lik den föregående banan, med ett *Door*-block som blockerar spelaren från målet och en knapp som måste tryckas ner först (se Figur 15). Här introducerar vi *Barrier*-block, och spelaren måste använda klonens attack-förmåga för första gången för att slå sönder måltavlan. Syftet med banans design är att bryta mot spelarens förväntningar, då likheterna med den förra banan står i kontrast till det faktum att banans lösning är motsatt - här måste spelaren stå på knappen medan klonen slår sönder måltavlan. Tanken bakom likheterna är att spelaren ska förstå att lösningen involverar *Clone*-mekaniken eftersom det finns ett *Door*-block och en knapp här.

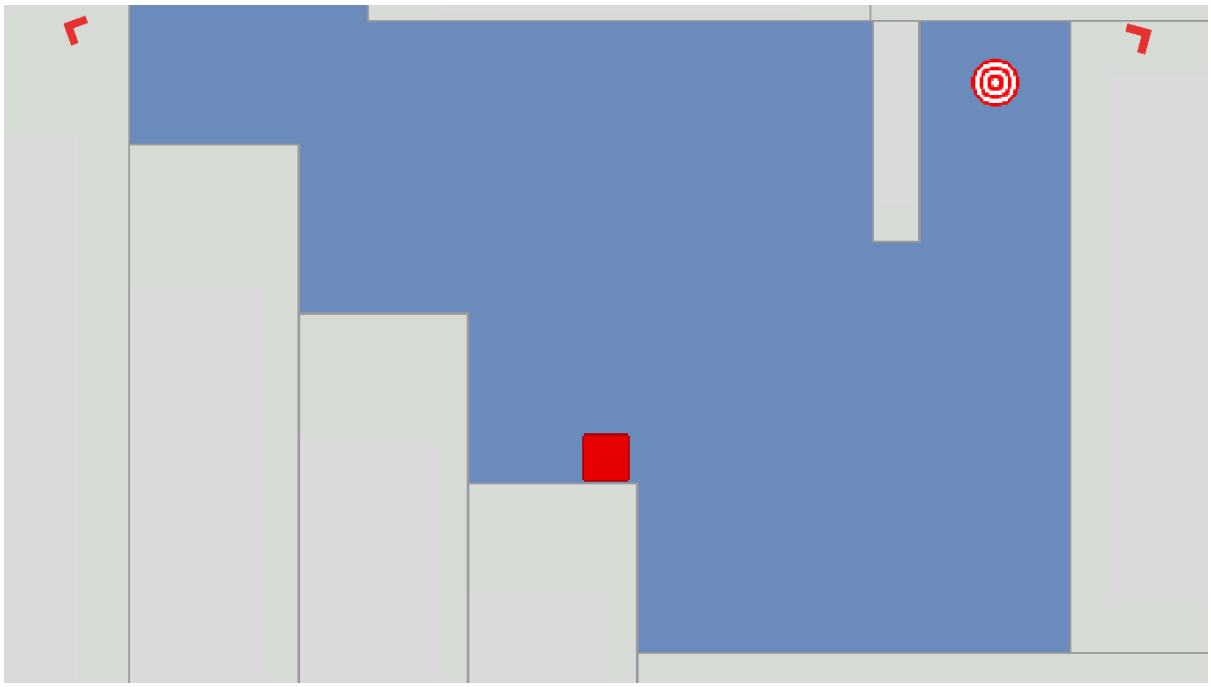
Bana 7: Final test

Den sista banan innehåller tre måltavlor som spelaren måste slå sönder för att vinna, och varje har varsin *impasse* som spelaren måste lösa. Det finns pilar som pekar mot måltavlorna för att informera spelaren om vart de måste gå för att hitta pusslen. För att klara av denna bana måste testdeltagarna kombinera vissa av spelets mekaniker. På grund av att lösningarna på dessa *impasses* inte står explicit i mekaniklistan tvingas deltagarna att använda sig av tidigare erfarenheter för att fylla i luckorna i sin kunskap.



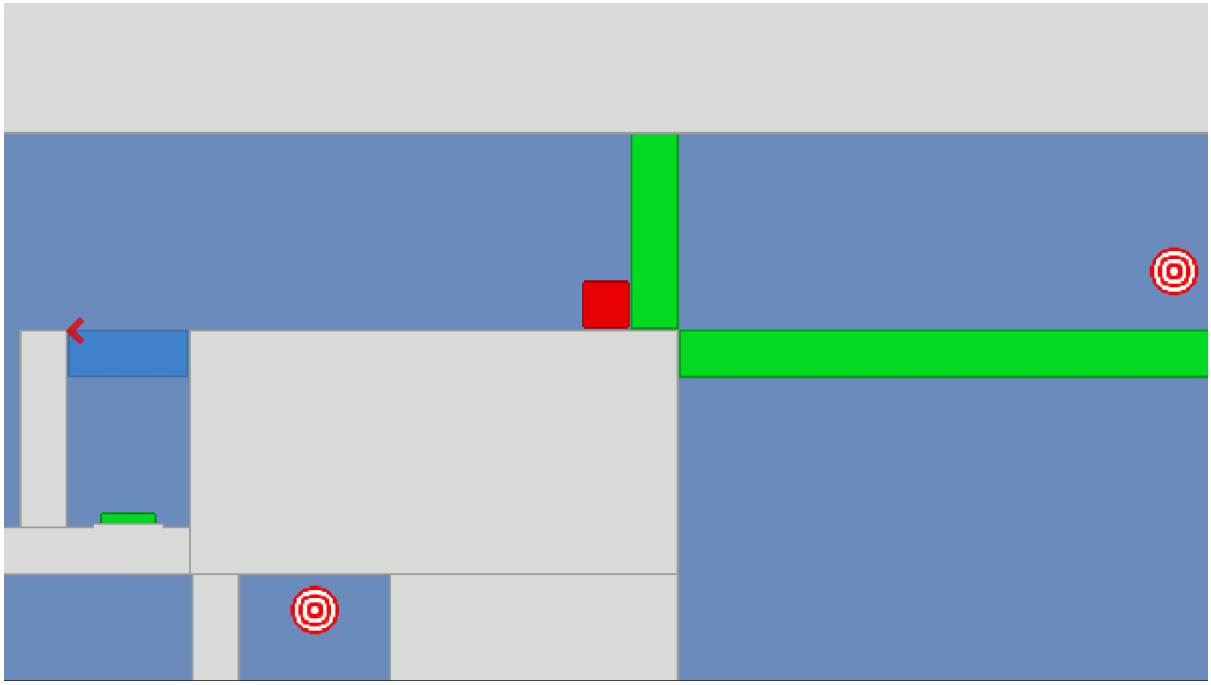
Figur 16 Sista banan, första pusslet.

Pusslet med lägst komplexitet är placerat direkt till vänster om spelarens startposition. Spelaren måste hoppa över ett stort hål för att slå en måltavla i luften (se Figur 16). Om spelaren trillar ner i hålet teleporteras de tillbaka till spelarens startposition i banan, så spelaren kan inte stå i botten av hålet och hoppa upp med ett *Charge Jump*. Lösningen på denna *impasse* är att använda *Sprint* och *Downward Aerial*, två mekaniker som inte tidigare använts i kombination för en *impasse*.

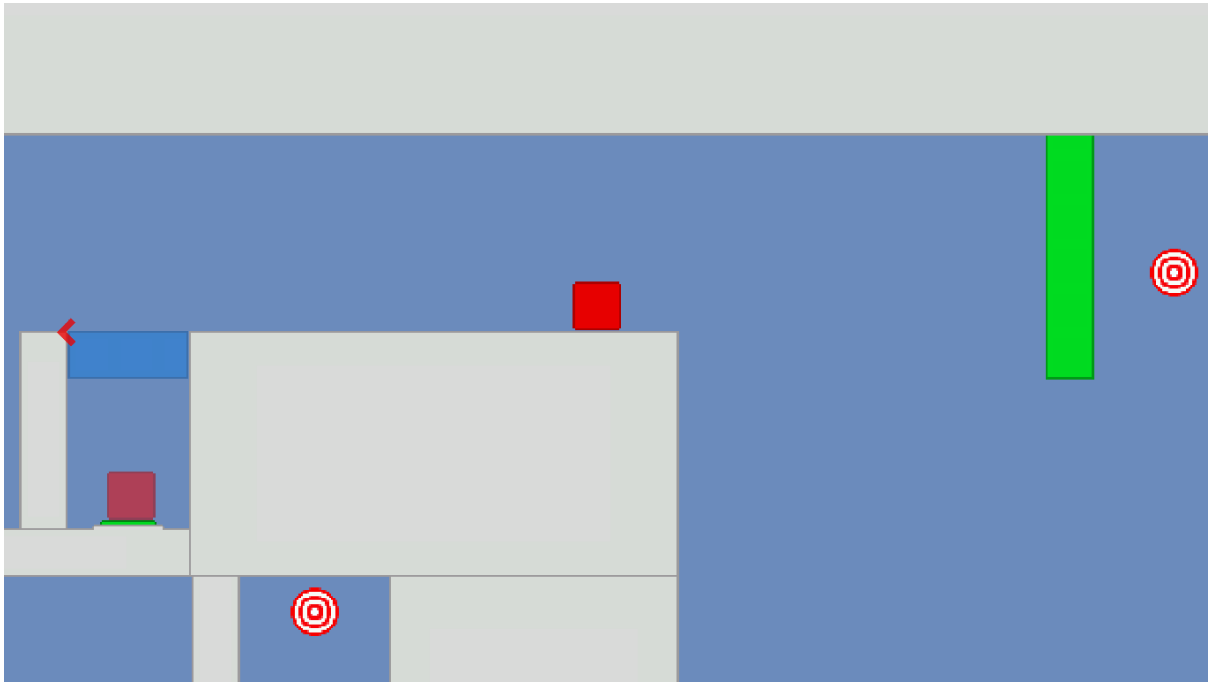


Figur 17 Sista banan, andra pusslet.

Nästa pussel hittas vid slutet av en trappa till höger om spelarens startposition. Detta pussel kräver att spelaren utför ett *Charge Jump*, sedan en *Downward Aerial* och sedan en uppåt-attack för att nå måltavlan som är väldigt högt upp (se Figur 17). Spelaren har inte behövt utföra uppåt-attacker tidigare, men de står med i spelets mekaniklista under Normal Attacks-sektionen och det är sannolikt att de har utfört en sådan attack tidigare eftersom den endast kräver att man drar rörelsespaken uppåt när man attackerar, som är lätt att utföra oavsiktligt.



Figur 18 Sista banan, tredje pusslet.



Figur 19 Sista banan, tredje pusslet, knappen nedtryckt.

Pusslet med störst komplexitet finns direkt till höger om spelarens startposition. Spelaren måste aktivera en knapp för att komma till måltavlan (se Figur 18), men att aktivera knappen

tar bort golvet (se Figur 19), så spelaren kan inte nå måltavlan när knappen är nedtryckt. Lösningen är att spelaren aktiverar knappen, hoppar ut över hålet, och sedan tar bort klonen från knappen, vilket kan göras genom att placera en ny klon eller att attackera. Golvet dyker då upp igen och måltavlan kan nås. Ett *Door*-block dyker upp framför måltavlan när knappen är nedtryckt för att förhindra spelaren från att hoppa till måltavlan och slå den i luften.

6 Utvärdering

I detta kapitel kommer skillnaderna i strategival mellan alla åtta testdeltagarna presenteras, analyseras och diskuteras. Resultaten kommer att gå igenom varje individuell deltagares spelbakgrund, beteenden och buggar som visades under spelsessionernas gång, samt deras åsikter angående hur svåra spelets mekaniker var att förstå. Testdeltagarnas resultat är sorterade efter hur lång tid det tog för dem att klara spelet. I analysen kommer vi att analysera resultaten för att sammanställa skillnader i testdeltagarnas spelsessioner. Frågeställningen kommer besvaras i slutsatsen.

6.1 Presentation av undersökning

Tabell 1 Tiden det tog för varje deltagare att klara varje bana.

	Deltagare							
Bana	1	2	3	4	5	6	7	8
Tutorial	0:11	0:12	0:14	0:33	0:13	0:39	1:11	1:12
Sprint	0:20	0:09	0:36	0:50	1:13	1:10	1:32	1:37
Charge Jump	0:18	1:16	1:08	0:30	0:34	0:53	0:58	2:29
Downward Aerial	0:37	0:06	0:20	0:47	2:04	1:40	4:38	3:56
Door	0:24	0:12	0:53	0:54	0:30	0:30	3:55	4:02
Barrier	0:24	0:31	0:39	0:59	0:43	2:06	5:14	4:44
Final Test	1:44	2:24	2:45	4:52	5:20	4:05	8:54	13:18
Total	**4:01**	**4:53**	**6:37**	**9:29**	**10:40**	**11:05**	**26:25**	**31:22**

Testdeltagare 1

Testdeltagare 1 är en 21 år gammal man. I den första intervjun beskrev han att han hade mycket generell spelerfarenhet, och att han var mycket erfaren med actionplattformspel. Han spelar omkring 14 timmar i veckan men han har varit för upptagen för att göra det på senare tid. Han sade att han som mest hade spelat runt 70 timmar i veckan under ett sommarlov.

Under spelsessionen kollade han upp information om spelets mekaniker via *EIR*, och arbetade sig igenom spelet metodiskt. Han började varje bana med att kolla upp informationen som han trodde skulle vara relevant till att lösa varje *impasse*, varpå han utförde lösningen med sin nya kunskap. Det uppstod en bugg i hans kunskap om *Downward Aerial*, då han tittade på den i mekaniklistan och sade “åh, fast det där kan jag inte göra, för det finns ju ingen måltavla…” Han antog alltså att han behövde träffa en måltavla med attacken för att studsa uppåt, antagligen eftersom attacken träffar en måltavla i videodemonstrationen. Han ifrågasatte om hans teori var korrekt och testade att utföra mekaniken, varpå han sedan insåg hur den fungerade. Han klarade spelets sista bana genom att använda *Repair* och fastnade inte under en längre tid på något av hindren där.

I den sista intervjun berättade han att han hade sett variationer på alla spelets mekaniker tidigare, och kunde ge exempel på spel där varje enskild mekanik dök upp. Till exempel kände han igen *Downward Aerial* från *Hollow Knight* (2017), men nämnde även att “Det är ju inte exakt samma, för man behöver ju inte ha något under sig.” (se Appendix B), vilket vidare förklarar buggen som uppstod. Han ansåg att *Clone* var den svåraste mekaniken att begripa,

då det tog honom ett kort tag att inse att klonen endast kopierar attacker, inte rörelsemekaniker. Total speltid: 4:02 (se tabell 1).

Testdeltagare 2

Testdeltagare 2 var en 26 år gammal man som beskrev sig själv som mycket erfaren i den första intervjun, då han har spelat spel sedan han var sex år gammal. Han har spelat ett fåtal actionplattformspel tidigare och är därför någorlunda bekant med spelgenren, men brukar inte spela sådana spel vanligtvis. Tiden han spenderar på spel varierar mellan 8 till 20 timmar i veckan, men tidigare hade han spenderat upp emot 50 timmar i veckan på spel.

Testdeltagaren började spelsessionen med att trycka på slumpmässiga knappar och se vad som hände. Han använde inte *Sprint* för att klara *Sprint*-banan, då han lyckades hitta *Downward Aerial* i *Tutorial*-banan och använde den istället. Detta ledde till en *impasse* i sista banan eftersom en av utmaningarna där kräver att man använder *Sprint*, och han visste inte att den mekaniken fanns. I *Charge Jump*-banan så ställde han frågan "finns det ett vägghopp?", och sedan testade han om spelet hade en mekanik som skulle låta honom klättra eller hoppa ifrån väggar. Under spelsessionen testade han ofta *Clone*-mekaniken som en potentiell lösning till *impasses*, exempelvis genom att försöka slå sin egen karaktär med klonen i ett försök att förflytta sig över ett hinder. Eftersom deltagaren endast experimenterade med mekanikerna för att lära sig mer om dem så är detta beteende inte ett tecken på att buggar i hans kunskap uppstod; det var avsaknaden av kunskap som ledde till beteendet, inte ett missförstånd eller ett fel i hans kunskap. Han använde sig av *Repair* för att lösa alla *impasses* i spelet utom den sista i spelet, som krävde att han använde sig av *Sprint*. Det var det enda tillfället i spelsessionen som han behövde söka extern information för att lära sig om en mekanik. Inga märkbara buggar syntes trots hans användning av *Repair*.

I den sista intervjun berättade han att han fann majoriteten av mekanikerna intuitiva, men han tyckte att det var ovanligt att *Sprint*-knappen var bunden till 'RB' knappen på kontrollen hellre än 'B' knappen. Han kände igen *Downward Aerial* från *Hollow Knight* (2017), men kunde inte minnas exakt hur den fungerade i det spelet. Han ansåg att mekaniken som var svårast att förstå var *Clone*-mekaniken, eftersom han tänkte att den kunde användas på många sätt och inte kände till dess begränsningar. Total speltid: 4:54 (se tabell 1).

Testdeltagare 3

Testdeltagare 3 är en 25-årig man som beskrev sig själv som en mycket erfaren spelare. Han berättade att han hade spelat många spel inom actionplattform-genren tidigare och beskrev den som sin favoritgenre. Han spelar omkring 10 timmar per vecka, men har som mest spelat 40 timmar per vecka under perioder då han haft mer fritid.

Testdeltagaren hittade *Downward Aerial* i början av spelsessionen, och använde den för att klara *Sprint*-banan utan att använda *Sprint*. Likt Testdeltagare 2 så visste han inte att *Sprint*-mekaniken fanns, så han lärde sig den genom *EIR* i slutet av spelet. Han provade ofta först med *Repair*, men övergick relativt snabbt till *EIR* om han inte hittade lösningen. Det verkade inte som att några buggar uppstod i hans kunskap.

I den sista intervjun sade deltagaren att han kände igen alla spelets mekaniker sedan tidigare, men sade även att *Downward Aerial* skiljde sig från hans förväntningar eftersom den förde spelkaraktären uppåt. Han anmärkte att mekaniken var lik ett dubbelhopp, och att dubbelhopp brukar vara knutna till hoppknappen. Total speltid: 6:38 (se tabell 1).

Testdeltagare 4

Testdeltagare 4 är en 25-årig spelare som sade att han brukade spela mycket när han var yngre, ungefär 40 timmar i veckan, men nuförtiden spelar han omkring 3 timmar per vecka. Han beskrev sig som "medium-erfaren" med spel över huvud taget, och sade att han inte spelat många actionplattformspel tidigare.

Han inledde testet med att testa kontrollerna, men han hittade ingen av de mer avancerade mekanikerna. I *Sprint*-banan öppnade han mekaniklistan och skummade igenom den, och använde sedan *Sprint* för att klara utmaningen. Han använde sig huvudsakligen av *EIR*, men i *Door* provade han *Repair* först. I den sista banan använde han endast *Repair*, och behövde inte öppna mekaniklistan för att veta vad han behövde göra. Han visade inte några tecken på buggar.

Deltagaren kände igen *Downward Aerial* från *Super Smash Bros. Brawl* (2008), *Charge Jump* från *Super Mario 64* (1996), *Door*- och *Sprint-mekaniken* från flera spel men nämnde inga exempel och *Barrier*-mekaniken kände han inte igen. Delen av spelet som han hade svårast att förstå var kombinationen av *Barrier*- och *Door*-block i den sista banan. Total speltid: 9:29 (se tabell 1).

Testdeltagare 5

Testdeltagare 5 är en man vid 22-års ålder som har spelat spel förut, men som knappt har spelat de senaste 4 åren. När han spelade förr brukade han spela ungefär 14 timmar i veckan. Han har ingen erfarenhet av spel inom actionplattform genren.

Han började med att experimentera lite med spelets kontroller, men hittade ingen av de mer avancerade mekanikerna. Han spelade varje bana på ungefär samma sätt: först försökte han lösa utmaningarna genom *Repair* och om han inte kunde hitta en lösning använde han *EIR*. Han bad om hjälp i *Downward Aerial*-banan eftersom han inte kunde komma på vad det var han behövde göra. Han försökte använda *EIR* för att klara utmaningen, men lyckades inte lista ut att han behövde använda *Downward Aerial*. När vi visat vilken mekanik han skulle använda så sade han: "jaha, jag läste ju inte ens, hade jag gjort det hade jag fattat,", varpå han snabbt klarade hindret. I den sista banan visade det sig att en bugg i hans inlärning hade uppstått, då han hade missförstått kraven för att använda *Downward Aerial*. Han hade fått intrycket av att karaktären var tvungen att vara på väg nedåt för att han skulle kunna utföra handlingen, vilket inte är fallet. Han fixade buggen genom att experimentera med mer *Repair*.

Han förklarade att han delvis kände igen *Downward Aerial* från spelet *Super Smash Bros. Brawl* (2008), och han kom ihåg *Door*-mekaniken från Portal 2 (2011). Han kände inte igen *Clone*, *Barrier* eller *Charge Jump*. Han anmärkte att *Downward Aerial* var svårast att förstå, och att han först trodde att attacken var tvungen att träffa något för att ge spelkaraktären mer höjd i luften. Han nämnde aldrig *Sprint* eller *Basic Attack*. Total speltid: 10:41 (se tabell 1).

Testdeltagare 6

Testdeltagare 6 är en kvinna som är 24 år gammal. Hon brukar spendera omkring 10 till 20 timmar i veckan på spel, men hon spelar främst simulationsspel och mobilspel. Som mest har hon spelat 45 timmar på en vecka, under en period av ledighet. Hon beskrev sig själv som icke-erfaren trots att hon spelar regelbundet, eftersom hon spelar väldigt få spel men spenderar mycket tid på de hon spelar. Hon har aldrig spelat ett actionplattformspel.

Testdeltagaren experimenterade inte med kontrollerna och mekanikerna under spelets gång, utan föredrog att använda *EIR* för att klara banorna. Hon använde *Sprint* för att ta sig förbi *Sprint*-banan och hade få problem med att förstå spelet. Vid *Charge Jump*-banan uppstod dock en bugg i hennes kunskap om *Charge Jump* mekaniken. Till en början insåg inte deltagaren att hoppet behövde laddas upp, men efter att hon experimenterat lite med mekaniken insåg hon att hon behövde hålla ned styrspaken tills karaktären började blinka. När detta skedde sade hon "Hur länge kan jag ladda det här då? Blir det längre ju mer jag håller in?". Hon gjorde alltså antagandet att styrkan av ett *Charge Jump* ökar med tiden, vilket inte är fallet då ett *Charge Jump* alltid hoppar lika högt. När hon fått använda mekaniken fler gånger lyckades hon rätta till sitt missförstånd. Utöver detta uppstod inga buggar under hela spelsessionen.

Hon kände igen *Sprint*-, *Door*- och *Basic Attack*-mekanikerna från spelet *The Elder Scrolls V: Skyrim* (2011), men resten av spelets mekaniker var nya för henne. Mekaniken som hon tyckte var svårast att använda sig av *Downward Aerial*, på grund av att det var svårt att utföra knappkombinationen som mekaniken är knuten till. Total speltid: 11:05 (se tabell 1).

Testdeltagare 7

Testdeltagare 7 är en 51-år gammal man med mycket begränsad spelerfarenhet. Senaste gången han spelade spel var ungefär 10 år sedan, och han anser sig själv vara en icke-erfaren spelare. Han har spelat spel inom plattform-genren tidigare, men har ingen erfarenhet med 2D-actionplattformspel.

I *Tutorial*-banan förstod han vad målet med spelet var, men han hade svårt att få karaktären att slå sönder måltavlan. I *Charge Jump*-banan så förstod han vilken mekanik han behövde använda sig av, men lyckades inte utföra mekaniken och var tvungen att be om hjälp. Han hade svårt med att klara av banorna rent fysiskt sett, även när han demonstrerade förståelse för vilken mekanik som behövde användas. Särskilt i den sista banan lyckades han inte utföra handlingarna som krävdes för att nå måltavlorna.

Han kände igen *Sprint* och *Basic Attacks* från spel som han spelat tidigare, och han ansåg att *Downward Aerial*-mekaniken var svårast att förstå eftersom han hade problem med att utföra den. Total speltid: 26:25 (se tabell 1).

Testdeltagare 8

Testdeltagare 8 är en 29-årig kvinna med mycket lite spelerfarenhet. Hon förklarade att hon ibland spelade spel med sin bror i sin barndom, men att hon inte spelat spel sedan dess. Hon har därför ingen erfarenhet av spel i actionplattform-genren.

Under spelsessionens gång hade hon svårt att styra spelkaraktären, och hon hade problem med att utföra flera handlingar samtidigt. Hon använde *EIR* vid varje *impasse*, och vid två tillfällen bad hon om hjälp då hon inte förstod vad som behövde göras. Det tog henne flera försök att fysiskt utföra handlingarna som krävdes för att klara banorna, och hon föredrog att stå stilla och fundera ut en lösning innan hon agerade. Hon lyckades lista ut hur man skulle gå tillväga för att klara majoriteten av spelets banor, men hon behövde hjälp med att navigera *impasses* mot slutet av spelet som krävde fingerfärdighet. Många buggar i lärandeprocessen uppstod under spelsessionens gång: Testdeltagaren missförstod rörelsesystemet och snärtade styrspaken istället för att hålla den i en riktning. Hon trodde att *Downward Aerial* skulle utföras på marken och missförstod målet med spelet eftersom hon missade att det fanns

attacker. I alla dessa fall så fastnade hon till den grad att testledarna var tvungna att rätta till hennes missförstånd för att testet skulle kunna fortsätta.

Hon kände igen spelets grundläggande kontroller, men hade inte sett någon av spelets mer avancerade mekaniker i något annat spel. När hon ombads beskriva vilken mekanik som var svårast att förstå svarade hon att *Downward Aerial* var svårast, på grund av att det var mekaniken som var mest komplex i sitt utförande. Total speltid: 31:22 (se tabell 1).

6.2 Analys

Tabell 2 Deltagarnas strategival i varje bana.

	Deltagare							
Bana	1	2	3	4	5	6	7	8
Sprint	EIR	Repair	Repair	EIR	Repair & EIR	Repair & EIR	EIR	EIR
Charge Jump	EIR	Repair	Repair & EIR	EIR	EIR	EIR	EIR	EIR
Downward Aerial	EIR	Repair	Repair	EIR	Repair & EIR	EIR	EIR	EIR
Door	EIR	Repair	Repair & EIR	Repair & EIR	EIR	EIR	EIR	EIR
Barrier	EIR	Repair	Repair	EIR	Repair & EIR	EIR	EIR	EIR
Final Test	Repair & EIR	Repair & EIR	Repair & EIR	Repair	Repair & EIR	Repair	EIR	EIR

Tabell 2 visar strategierna som varje deltagare valde i varje bana. Celler markerade med *EIR* innebär att deltagaren klarade en *impasse* med hjälp av mekaniklistan utan att försöka använda *Repair* alls. *Repair* betyder att testdeltagaren löste banan utan att öppna mekaniklistan. *Repair* & *EIR* innebär att deltagaren försökte använda *Repair*, men behövde öppna mekaniklistan för att hitta lösningen.

Testdeltagarna har delats in i tre separata grupper för att underlätta analysen av resultaten. Den första gruppen består av deltagare 1, 2 och 3. Dessa deltagare beskrev sig själva som väldigt eller mycket erfarna med spel och var bekanta med spelgenren och dess konventioner.

Den andra gruppen består av deltagare 4, 5 och 6. Deltagare 4 och 5 beskrev sig själva som någorlunda erfarna med spel och hade spelat spel tidigare. Deltagare 6 beskrev sig själv som icke-erfaren, men hennes testtid (se tabell 1), spelfrekvens och svar på efterhandsintervjun hade mer gemensamt med de mellan-erfarna spelarna än de icke-erfarna. På grund av detta placerades hon i den andra gruppen.

Den tredje gruppen består av deltagare 7 och 8. Dessa deltagare beskrev sig själva som icke-erfarna med spel. Båda hade spelat spel för länge sedan, men båda ansåg sig själva vara överlag obekanta med spel. Ingen av dem hade spelat actionplattformspel tidigare.

Hur de mest erfarna spelarna spelade

Testdeltagare 1, 2 och 3 var mycket erfarna spelare som spelar spel varje vecka. Deltagare 1 förlitade sig mycket på *EIR*, deltagare 2 använde sig nästan enbart av *Repair*, och deltagare 3 använde sig av en blandning av *EIR* och *Repair* (se tabell 2). De erfarna deltagarna bad aldrig om hjälp, och de fastnade väldigt sällan längre än en minut vid någon *impasse*. De pratade mest om vad de behövde göra härnäst, och funderade ofta högt om hur spelmekanikerna fungerade och vilken mekanik som passade bäst för att lösa varje *impasse*. När deltagarna använde sig av *Repair* så provade de ofta strategier som involverade mekaniker som finns i andra actionplattformspel, exempelvis vägghopp, vilket tyder på att de utgick från sina tidigare erfarenheter för att lista ut en möjlig lösning.

De erfarna spelarna hade överlag lätt för att förstå hur spelet fungerade och vad som förväntades av dem, vilket reflekteras i deras testtider som var 4:02, 4:54 och 6:38 (se tabell 1). Hong och Liu (2003) fann att spelare som presterade bättre i deras studie tänkte mer på sina handlingar, vilket tydligt kunde ses hos deltagarna i här studien med. När buggar uppstod var de erfarna testdeltagarna snabba med att rätta till sin kunskap, och kunde ofta göra det genom *Repair* hellre än att leta efter den korrekta informationen med *EIR*. Det är svårt att säga om erfarna spelare föredrog *EIR* eller *Repair* eftersom testgruppen var så liten, men de använde definitivt *Repair* oftare än de mindre erfarna spelarna gjorde. Det är dock svårt att ge en konkret anledning för detta, då många faktorer kan ha påverkat vilken metod som användes.

Hur de mellan-erfarna spelarna spelade

Testdeltagare 4 och 5 var spelare som inte spelar aktivt, men som gjort det tidigare. Testdeltagare 6 spelar aktivt och mer än vad de andra mellan-erfarna deltagarna gjorde, men hon saknar erfarenhet inom spelgenren som undersöks. Tiderna för deras spelsessioner låg på 9:29, 10:41 och 11:05 (se tabell 1), vilket är inom två minuter av varandra och skulle kunna tyda på liknande erfarenhetsnivåer. De mellan-erfarna deltagarna funderade högt över vad de behövde göra vid varje *impasse*, men de verkade ha färre idéer än de erfarna spelarna och förlitade sig alla huvudsakligen på mekaniklistan (se tabell 2).

Testdeltagare 5 försökte använda *Repair* i början av många banor, men använde i slutändan *EIR* då han ofta inte lyckades hitta lösningen på egen hand. Liknande beteende kunde ses i hur deltagare 4 och 6 spelade också, då de förlitade sig mer på *EIR* än de erfarna spelarna gjorde. De mellan-erfarna spelarna försökte dock använda sig av *Repair* vid flera tillfällen, och lyckades ibland klara av *impasses* utan *EIR* (se tabell 2). De var dock inte lika effektiva på att använda *Repair* som de erfarna spelarna var.

Intressant nog uppstod buggar även när de mellan-erfarna spelarna använde sig av *EIR*, vilket inte skedde för de mest erfarna spelarna. Det är möjligt att de mellan-erfarna spelarna inte hade stött på liknande mekaniker så ofta, och att de därför hade sämre referensramar vilket gjorde det svårt för dem att tolka informationen som stod. Alla deltagare hade tillgång till exakt samma information, så det faktum att *EIR*-buggar endast uppstod för de mindre erfarna spelarna måste bero på skillnader i deltagarna.

De mellan-erfarna spelarna tog längre tid på sig att lösa buggar än vad de mest erfarna spelarna gjorde. Ett exempel på detta kan ses i *Downward Aerial*-banan, där buggar uppstod både för deltagare 1 och 5.

Hur de minst erfarna spelarna spelade

Testdeltagare 7 och 8 spelar nästan aldrig spel och saknade därför lågnivå-heuristiker som de mer erfarna spelarna hade tillgång till. De kände inte till konventioner som de flesta spelare är bekanta med, exempelvis att man rör på spelkaraktären med vänster styrspak. De använde sig endast av *EIR* (se tabell 2), vilket skulle kunna bero på att de var för ovana med att spela spel för att kunna experimentera och utföra *Repair*. *Repair* bygger på tidigare erfarenheter för att hitta en lösning (VanLehn, 1988). Detta kan ha gjort det svårt för de minst erfarna spelarna att använda sig av strategin, eftersom de inte visste vad man brukar kunna göra i spel. Testdeltagare 7 och 8 tog mycket längre tid på sig än de mer erfarna spelarna, med spelsessioner som varade i 26:25 respektive 31:22 minuter (se tabell 1). De fastnade ofta på

ställen som mer erfarna spelare klarade utan problem, exempelvis när de skulle hoppa på några plattformar för att komma fram till knappen i *Barrier*-banan. De icke-erfarna spelarna var även de enda deltagarna som bad om hjälp flera gånger i en spelsession, och både deltagare 7 och 8 behövde hjälp med att utföra handlingar i den sista banan.

De minst erfarna deltagarna stötte på fler buggar under inlärningsprocessen än någon av de andra testdeltagarna, trots att de använde sig mest av *EIR*. Detta kan ha berott på att informationen som mekaniklistan innehåller inte var beskrivande nog för att de skulle förstå hur mekanikerna fungerade utan förkunskap, men det skulle kunna finnas andra faktorer som påverkade detta.

6.3 Slutsatser

Resultatet av studien visar att spelarna som hade relativt låg erfarenhetsnivå föredrog att använda sig av *EIR* (se tabell 2). Detta kan ha berott på att erfarenhet krävs för att använda *Repair* på ett effektivt sätt (VanLehn, 1988), och att de mindre erfarna spelarna inte trodde att de skulle kunna lista ut lösningen själva. En annan möjlig anledning är att de mindre erfarna spelarna inte ens försökte lista ut spelet själva, utan gjorde exakt som instruktionerna på skärmen sade i *Sprint*-banan: “Open the glossary if you get stuck” (se figur 11). Man kan också se det som att de flesta spelare använde *EIR* som standard och att endast de mer erfarna spelarna valde att använda *Repair*, kanske för att utmana sig själva eller för att se om de kunde. Det är svårt att säga varför deltagarna använde den strategi som de valde, men är uppenbart att de mer erfarna spelarna använde mer *Repair* medan de mindre erfarna spelarna använde mer *EIR*.

Datan visar även att buggar uppstod när mindre erfarna spelare använde *EIR*. En möjlig förklaring till detta kan vara att de korta beskrivningarna i mekaniklistan inte gav tillräckligt detaljerad information för att de mindre erfarna spelarna skulle förstå hur mekanikerna fungerade ordentligt. Det är också möjligt att informationen presenterades på ett sätt som skapade förvirring, trots att informationen i sig kanske var tillräcklig.

Deltagare 1 och 5 gjorde påståenden i *Downward Aerial*-banan som kan stödja denna teori. Mekaniklistans beskrivning för *Downward Aerial* är “attack downward and blast yourself upward”, och videon för mekaniken visar att en spelkaraktär slår sönder en måltavla med attacken och studsar uppåt. Deltagare 1 verkade titta på videon hellre än att läsa beskrivningen, då han sade “fast det där kan jag inte göra, för det finns ju ingen måltavla under” när han tittade i mekaniklistan. Deltagare 5 berättade också att han inte läste beskrivningen, och därför missförstod mekaniken: “jaha, jag läste ju inte ens, hade jag gjort det hade jag fattat.” Det är alltså möjligt att videorna tog upp så mycket av deltagarnas fokus att de missade beskrivningarna helt, vilket speciellt är ett problem med *Downward Aerial* eftersom videon verkade vara otydlig. Rimligtvis hade alla sidor i mekaniklistan samma problem eftersom de följde samma upplägg. Det är möjligt att resultaten hade varit annorlunda om videorna tog upp mindre fokus.

På grund av att studien endast utfördes på åtta deltagare så är det möjligt att sambanden som observerats mellan erfarenhet och strategival endast är sammanträffanden, och inte har någon gemensam orsak. Därför är slutsatserna som dragits i denna studie inte applicerbara utanför deras ursprungliga kontext, och bör hellre användas som en utgångspunkt för vidare forskning inom ämnet.

7 Avslutande Diskussion

7.1 Sammanfattning

Syftet med detta arbete var att undersöka potentiella skillnader i hur spelare med olika erfarenhetsnivå använder sig av *Repair* och *External Information Retrieval* (*EIR*) vid *impasses*. För detta syfte skapades en artefakt av ett actionplattformspel, som var designad för att spelare skulle stöta på och lösa *impasses*.

Undersökningen utfördes på åtta personer som själva fick bedöma sin erfarenhet av spel, både generellt sett och inom actionplattform-genren. Därefter fick de spela prototypen samtidigt som de redogjorde för sina tankar. Detta spelades in tillsammans med en video av deras spelsession. Efter spelsessionen fick testdeltagarna berätta vilka mekaniker de kände igen sedan tidigare, och de fick säga vilken mekanik de hade svårast för att förstå.

Resultatet av undersökningen visade att de mest erfarna spelarna använde sig av *Repair* oftare än de icke-erfarna spelarna, som huvudsakligen förlitade sig på *EIR*. Buggar uppstod i inlärningen för spelare av alla erfarenhetsnivåer, men de erfarna spelarna löste sina buggar snabbare än de icke-erfarna spelare och demonstrerade en bättre förståelse för spelmekanikerna. Det uppstod flest buggar för de mindre erfarna spelarna trots att de huvudsakligen använde *EIR*, som brukar förhindra att buggar uppstår (VanLehn, 1988).

Det är svårt att säga varför spelarna valde den strategi som de gjorde, men det finns tydliga skillnader i hur strategierna användes beroende på spelarnas erfarenhetsnivå. Studiens slutsatser är att buggarna som uppstod vid *EIR* kan ha berott på kvalitén av informationen, samt sättet som den presenterades på. Resultaten är svåra att applicera i en bredare kontext på grund av studiens begränsade omfång.

7.2 Diskussion

Trovärdigheten av studiens resultat påverkas av ett antal faktorer. Först och främst har studien få deltagare från många olika spelbakgrunder, vilket gör resultaten svåra att analysera då det är omöjligt att förstå den fullständiga kontexten bakom spelarnas beslut att använda strategierna som de gjorde. Även med en större testgrupp hade det varit svårt att fullständigt förstå spelarnas resonemang, men det hade gett mer data att analysera vilket skulle göra det lättare att dra slutsatser med säkerhet.

Erfarenhet är svårt att bedöma oavsett vilken metod man använder, så vi valde att låta testdeltagarna självskatta sin erfarenhet för att få en så bra beskrivning som möjligt. Med detta faller ansvaret på oss att jämföra vem som är mer erfaren, vilket vi gjorde genom att gruppera spelare som vi upplevde gav liknande beskrivningar. Detta kan ha försämrat studiens trovärdighet, då det är fullt möjligt att vi kategoriserade spelare felaktigt och gjorde antaganden baserade på en falsk gruppering.

Vissa aspekter av spelprototypens design hade en oförväntad inverkan på studiens resultat. Till exempel var det möjligt för spelarna att klara *Sprint*-banan utan att använda Sprint, vilket var ett förbiseende som kunde ha lösts genom att sätta ett tak över spelaren för att förhindra att *Charge Jump* eller *Downward Aerial* kunde användas för att klara banan. I den sista banan hade två av åtta spelare inte använt *Sprint*-mekaniken tidigare. Tanken bakom spelets design var att alla spelare skulle uppleva samma *impasses* på samma ställen, men på grund av

Sprint-banan så det blev inte alltid så. Resultaten bör inte ha påverkats mycket av detta, då det inte var kritiskt viktigt att deltagarna stötte på *impasses* på samma ställen. Det finns en andra *Sprint-impasse* i den sista banan, så alla spelare stötte på samma mängd *impasses* även om två deltagare stötte på en *impasse* på ett annat ställe. Misstaget bidrog dock till onödig variation vilket gjorde resultaten aningen svårare att analysera.

Ett annat problem med prototypen var att *Final Test* innehöll tre *impasses* i en och samma bana. Utmaningarna är intressanta i sig, och testar appliceringar av mekanikerna som inte visats tidigare. Men det gick inte att analysera vilken strategi som användes vid varje impasse, då flera deltagare gick fram och tillbaka mellan hindren och använde *Repair* och *EIR* blandat. I efterhand är det uppenbart att *Final Test* borde ha varit tre banor för att tillåta bättre analys.

Spelare som fastnade under testet kunde be oss om hjälp med en *impasse*. Detta användes huvudsakligen av de minst erfarna spelarna, samt en gång av deltagare 5 som var mellan-erfaren. Beslutet att tillåta detta var inte tillräckligt genomtänkt, då förklaringarna av lösningarna som spelarna fick under spelsessionen varierade och inte följde någon mall. Informationen kan därför ha varierat i kvalitet och kvantitet, vilket kan ha bidragit till variation i resultaten.

En svaghet som studien hade var att det inte gick att härleda anledningen till att deltagarna valde den strategin som de gjorde. Det finns många möjligheter, t.ex. att det var snabbare att kolla upp informationen eller att någon såg det som en utmaning att klara spelet helt själv, men ingen data som talar för eller emot dessa teorier har samlats in. Ett sätt att förbättra studien hade därför varit att fråga deltagarna varför de spelade som de gjorde, då detta skulle kunna ge mer insikt i spelarnas tankesätt.

Mycket av det som VanLehn (1988) skriver om kan även ses i resultaten av denna studie. Det var uppenbart att erfarna spelare lyckades använda *Repair* bättre än mindre erfarna spelare, då de mycket oftare kom fram till en lösning utan att använda *EIR* (se tabell 2). Dock observerades att buggar uppstod oftare vid *EIR* för mindre erfarna spelare än det gjorde för mer erfarna spelare, vilket VanLehn inte nämner. Anledningen för detta skulle kunna vara att informationen som fanns tillgänglig för deltagarna skiljde sig åt mellan studierna.

Vi tog med alla personer som ville delta i studien, och ställde inga krav på att testgruppen skulle ha hög mångfald. Detta resulterade i att män utgjorde sex av åtta av studiens deltagare. Utöver detta hittade vi majoriteten av testdeltagarna på Högskolan i Skövdes campus, vilket ledde till en överrepresentation av unga vuxna. Det är möjligt att faktorer som ålder och kön kan ha haft en inverkan på studiens resultat, men på grund av bristande tid och resurser så var vårt enda val att fråga de som var tillgängliga och villiga att delta. Vi tror däremot att studien hade givit liknande resultat även om testgruppen var mer varierad, eftersom erfarenhet byggs när en person spenderar på en aktivitet (SAOL, 2015). Ålder, kön och socioekonomisk bakgrund till exempel skulle kunna påverka hur lätt det är för en spelare att bli mer erfaren, men två spelare med liknande erfarenhetsnivå är enligt vår mening jämförbara oavsett bakgrund.

Det är tydligt att icke-erfarna spelare har större problem med att lösa *impasses* än vad erfarna spelare har, både genom *Repair* och *EIR*. För att undvika att de fastnar behöver de därför tydligare instruktioner, och om en *impasse* skulle uppstå bör det finnas information tillgänglig i spelet för att tillåta *EIR*. Det är även viktigt att vara tydlig när man introducerar spelaren till nya mekaniker, då det är lätt för buggar i spelarens kunskap att uppstå när en spelmekanik

förklaras på ett vagt sätt. Ett bra sätt att undvika missförstånd är att låta spelaren demonstrera sin kunskap med ett praktiskt exempel, vilket många spel redan gör (Suddaby, 2012).

7.3 Framtida forskning

Syftet med det här arbetet har varit att belysa hur spelare av olika erfarenhetsnivå använder *Repair* och *EIR* vid *impasse*-drivna inlärningsmoment, för att öka förståelsen för forskningsområdet och underlätta framtida forskning. En fortsättning på detta arbete skulle huvudsakligen beröra tre aspekter: problem med spelprototypen skulle behöva åtgärdas, metoden för att genomföra undersökningen skulle kunna förbättras och undersökningen skulle behöva utföras på fler personer.

Om arbetet skulle fortsätta i en vecka till hade det varit möjligt att utföra studien på fler personer och möjligtvis att förbättra metoden, men det hade varit svårt att rätta till alla problem med prototypen. Det viktigaste hade varit att åtgärda problemet med *Sprint*-banan, och att göra så att spelarna måste använda *Sprint* för att klara den. Det skulle också vara intressant att fråga deltagarna om varför de valde strategierna som de använde.

Om studien skulle pågå i några månader till så skulle en mer avancerad undersökning av *Repair* och *EIR* kunna utföras. Detta skulle exempelvis kunna göras genom att skapa två spelprototyper: en som endast testar *Repair* och en som endast testar *EIR*. Detta hade varit intressant då det kan ge djupare inblick i hur strategierna används, även om det inte tillåter undersökning av varför strategierna väljs. Studiens omfattning hade fördubblats och det hade krävts många fler testdeltagare för att få användbara data, men det hade givit en mycket djupare inblick i hur spelare av varierande erfarenhetsnivå använder *Repair* och *EIR*. Det skulle även kunna vara intressant att undersöka den inverkan som spelerfarenhet har på spelarbeteende i en annan kontext än *impasse*-driven inlärning.

I den här studien observerades att buggar uppstod oftare vid *EIR* för icke-erfarna spelare, trots att de icke-erfarna deltagarna hade tillgång till samma information som de mer erfarna spelarna. Det skulle vara intressant att undersöka vad detta berodde på, och vilken inverkan kvalité och presentation av information har på *EIR*. Det hade också varit intressant att utföra fler studier som denna på spel i andra genrer, för att se hur applicerbara den här studiens slutsatser är i en bredare kontext.

Referenser

Blumberg, F. C., Rosenthal, S. F. och Randall, J. D. (2008) *Impasse*-Driven Learning in the Context of Video Games, Computers in Human Behavior, 24(4), pp. 1530–1541. doi: 10.1016/j.chb.2007.05.010.

Bryman, A. (2008) *Social research methods. 4:e utgåvan*. Oxford: Oxford Univ. Press.

Carr, D. och Burn, A. (2006). *Computer Games: Text, Narrative and Play*. Cambridge: Polity.

Cotton, D. och Gresty, K. (2006) Reflecting on the Think-Aloud Method for Evaluating E-Learning, *British Journal of Educational Technology*, 37(1), pp. 45–54.

Darabi, A., Arrington, T. L. och Sayilir, E. (2018) Learning from Failure: A Meta-Analysis of the Empirical Studies, Educational Technology Research and Development : *A bi-monthly publication of the Association for Educational Communications & Technology*, 66(5), pp. 1101–1118. doi: 10.1007/s11423-018-9579-9.

Desurvire, H. och El-Nasr, M. S. (2013) Methods for Game User Research: Studying Player Behavior to Enhance Game Design, *IEEE Computer Graphics and Applications*, 33(4). doi: 10.1109/MCG.2013.61.

Dotsenko, A. (2017) Designing Game Controls. *Andrew Dotsenko's Blog* [blogg], 29 mars, 2017. https://www.gamasutra.com/blogs/AndrewDotsenko/20170329/294676/Designing_Game_Controls.php [2019-03-12].

Elias, G.S., Garfield, R. och Gutschera, K.R. (2012). *Characteristics of games*. Cambridge, Mass: MIT Press Ltd.

FTL: Faster Than Light. (2012) [spel] Utvecklare: Subset Games. Shanghai, China: Subset Games.

Hedges, N. (2017) Video Game Tutorials: How Do They Teach?. *Nathan Hedges's Blog* [blogg], 13 oktober, 2017. https://www.gamasutra.com/blogs/NathanHedges/20171013/307378/Video_Game_Tutorials_How_Do_They_Teach.php [2019-02-07].

Hollow Knight. (2017) [spel]. Utvecklare: Team Cherry. Adelaide, South Australia: Team Cherry.

Hong, J. och Liu, M. (2003) A Study on Thinking Strategy between Experts and Novices of Computer Games, *Computers in Human Behavior*, 19(2), pp. 245–258. doi: 10.1016/S0747-5632(02)00013-4.

Iwata, S (2017) Letting Everyone Know It Was A Good Mushroom. *Iwata Asks* [blogg], 13 november, 2009. http://iwataasks.nintendo.com/interviews/#/wii/nsmb/0/3 [2019-05-17].

Klotski. (u.å.) [spel].

McDaniel, M. A., Schmidt, F. L. och Hunter, J. E. (1988) Job Experience Correlates of Job Performance, Journal of Applied Psychology, 73(2), pp. 327–330. doi: 10.1037//0021-9010.73.2.327.

Ori and the Blind Forest. (2015) [spel]. Utvecklare: Moon Studios. Washington, USA: Xbox Game Studios.

Palladino, L. J. (2007) Find Your Focus Zone: An Effective New Plan to Defeat Distraction and Overload, *Library Journal*, 132(10).

Portal 2. (2011) [spel]. Utvecklare: Valve Corporation. Bellevue, USA: Valve Corporation.

Schack. (u. å.) [spel].

Sonic the Hedgehog 2. (1992) [spel]. Utvecklare: Sonic Team. Tokyo, Japan: Sega.

Spelunky. (2008) [spel]. Utvecklare: Mossmouth, LLC. San Francisco, USA: Mossmouth, LLC.

Suddaby, P. (2012) *The Many Ways to Show the Player How It's Done With In-Game Tutorials*. https://gamedevelopment.tutsplus.com/tutorials/the-many-ways-to-show-the-player-how-its-done-with-in-game-tutorials--gamedev-400 [2019-02-08]

Super Mario 64. (1996) [spel]. Utvecklare: Nintendo. Kyoto, Japan: Nintendo.

Super Mario Bros. (1985) [spel]. Utvecklare: Nintendo. Kyoto, Japan: Nintendo.

Super Mario Bros. 2. (1988) [spel]. Utvecklare: Nintendo. Kyoto, Japan: Nintendo.

Super Meat Boy. (2010) [spel]. Utvecklare: Team Meat. Asheville, USA: Team Meat.

Super Smash Bros. Brawl. (2008) [spel]. Utvecklare: Nintendo. Kyoto, Japan: Nintendo.

Svenska Akademin (2015). *SAOL*. Stockholm: Norstedts Förlag.

The Elder Scrolls V: Skyrim. (2011) [spel]. Utvecklare: Bethesda Softworks. Rockville, USA.

Unity. (2018) [programvara]. Unity technologies. Tillgänglig på internet: https://unity3d.com

VanLehn, K. (1988). *Toward a theory of impasse-driven learning*. New York: Springer-Verlag.

Ware, C. (2008). *Visual Thinking for Design*. Burlington, Mass: Morgan Kaufmann.

Østbye, H., Knapskog, K., Helland, K., Larsen, L.O. (2003) *Metodbok för medievetenskap*. Malmö: Liber AB.

Appendix A - Beskrivning av spelet / studien

”Under detta test kommer du få spela sju stycken banor. Målet i varje bana är att förstöra alla måltavlor. För att komma vidare i spelet måste du lista ut hur spelets mekaniker fungerar. Du kan ta hjälp av spelets mekaniklista om du inte vet hur man tar sig förbi ett hinder. Vi vill att du berättar vad du tänker när du fastnar vid ett hinder. Vi är intresserade av att ta reda på hur du tar dig förbi hindren, så det är det som du borde tala om. Om du känner att du inte klarar av ett hinder så kan du fråga oss om hjälp, men helst skall du klara dem själv så försök själv innan du frågar.”

Appendix B - Deltagare 1 intervjuer

Förhandsintervju
Ledare: Hur gammal är du?

Deltagare: Jag är 21 år gammal.

Ledare: Okej, skulle du kunna beskriva hur erfaren du är med spel, generellt sett?

Deltagare: Jag har spelat aktivt sedan jag var fem år, så väldigt.

Ledare: Och hur mycket brukar du spela aktivt per vecka?

Deltagare: Ja, nu har jag ju inte tid för någonting, men om du menar annars... jag skulle säga runt fjorton timmar, liksom två timmar per dag.

Ledare: Okej, så hur mycket har du spelat som mest tidigare, under till exempel sommarlov?

Deltagare: Bra fråga, över hur lång tid?

Ledare: På en vecka, då alltså.

Deltagare: Jag vet inte, jag håller inte räkningen när jag är i ett sånt mode, men kanske tio timmar per dag.

Ledare: Okej. Hur erfaren är du med actionplattform-genren? Typ spel som Hollow Knight eller Rogue Legacy?

Deltagare: Ja, jag har ju spelat Hollow Knight. Jag har ju fått true ending. Så jag skulle säga att jag är mycket erfaren.

Efterhandsintervju
Ledare: Så, kände du igen några av mekanikerna sedan tidigare?

Deltagare: Ja, hoppa och sprinta det är ju liksom Mario så det är ju basic. Attackerna, själva animationen kändes ju väldigt Hollow Knight med kurvan och downward grejen som man hoppar med. Det är ju inte exakt samma, för man behöver ju inte ha något under sig. Jag undrar om jag har spelat något som har det så... Inte som jag kommer på, men det är fortfarande likt Hollow Knight eller Shovel Knight. Klon-grejen... vad heter det där spelet? The Swapper, heter det det? Då kan man klona sig själv och sedan byta mellan varandra så klonings-delen av det var ju med liksom ungefär, fast här är det bara att när man slår att det är det den härmar liksom, inte rörelsen. Knappar som man behöver stå på, ja det är ju Portal, typiskt pusselspel liksom. Barriärerna... Vet jag inte, för det har ju med klonerna att göra så då skulle det ju behöva finnas kloner också i spelet och det kommer jag inte på något.

Ledare: Okej, och vilken av de här mekanikerna tyckte du var svårast att förstå, och varför?

Deltagare: Ingen var ju svår att förstå, så då försöker jag tänka vilken var svårast... Jag skulle nog säga klon kanske bara för att jag inte visste - det kanske var för att jag inte läste förresten, stod det? Ah, jag läste inte det, det hade nog varit tydligt men annars så var det så att jag förstod inte riktigt att det var bara attacken som härmades, eftersom inget annat härmades liksom i klonen, så det skulle väl vara den då.

Ledare: Okej, tack så mycket!

Appendix C - Deltagare 2 intervjuer

Förhandsintervju
Ledare: Okej, lite frågor här innan vi börjar då. Först och främst, beskriv hur erfaren du är med spel.

Deltagare: Väldigt erfaren, skulle jag säga. Jag har lagt ned väldigt mycket tid på spel.

Ledare: Hur länge har du spelat?

Deltagare: Sedan jag var ungefär sex år gammal och spelade datorspel, skulle jag säga. Stor del av hela livet.

Ledare: Okej, hur mycket brukar du spela?

Deltagare: I timmar, eller?

Ledare: Ja, i timmar i veckan.

Deltagare: Jag snittar på... Det brukar bero på, men mellan 8 till 20 timmar i veckan skulle jag säga.

Ledare: Okej, hur mycket har du spelat som mest tidigare? Alltså under en period, typ ett sommarlov eller så.

Deltagare: Mest spelade perioden kan ha varit... ja nära 50 timmar i veckan kanske.

Ledare: Okej, hur erfaren är du med actionplattform-genren? Som typ Hollow Knight, Rogue Legacy, Dead Cells-ish?

Deltagare: Ja... Ganska erfaren, jag har spelat lite Hollow Knight, inte spelat klart det.

Ledare: Har du kört många metroidvanias?

Deltagare: Ja, en hel del. Jag har spelat några metroids hela vägen.

Ledare: Alright.

Efterhandsintervju
Ledare: Nu har vi några till frågor här. Först och främst, kände du igen några av de här mekanikerna från spel som du spelat tidigare?

Deltagare: Uh, jag försöker komma ihåg om Hollow Knight hade det där att man slår nedåt för att komma uppåt men jag minns inte riktigt, de kanske hade ett dubbelhopp snarare.

Ledare: Okej, okej. Charge jump?

Deltagare: Den var inte vanlig, faktiskt.

Ledare: Så den kände du inte igen?

Deltagare: Nä.

Ledare: Clone, något i den stilen?

Deltagare: Nä, det hade jag inte kört med förut.

Ledare: Okej, dörrar och barriärer?

Deltagare: Ja, dörrar kändes ganska logiska och när man bara prövade så funkade det med klonen för genomskinlig kändes som något som man borde kunna gå igenom.

Ledare: Okej, sprinting och vanliga attacker, det har du sett i spel innan?
Deltagare: Ja, jag är lite förvånad över att sprint var på den knappen just, för jag är van vid att den är på B eller någonting så då antog jag att det inte fanns.

Ledare: Japp, och vilken mekanik tyckte du att du hade svårast att förstå av de här?

Deltagare: Ja, det var väl klonen som var… riktigt hur den fungerade. Jag visste inte om man kunde använda den i luften på något vis eller så.

Ledare: Okej, tack så mycket!

Appendix D - Deltagare 3 intervjuer

Förhandsintervju
Ledare: Så, först och främst då. Hur erfaren är du med spel?
Deltagare: Jag skulle säga att jag är mycket erfaren.

Ledare: Okej, och hur mycket spelar du i veckan?
Deltagare: Tio timmar i veckan, i genomsnitt,

Ledare: Okej, så om du beskriver hur mycket du spelat som mest? I tidigare perioder av ditt liv?

Deltagare: Ja du... Har säkert varit mer än 40 timmar i veckan... Ja, vi säger 40 timmar i veckan, så ett heltidsjobb liksom.

Ledare: Hur erfaren är du med actionplattform-genren, typ spel som Hollow Knight, Rogue Legacy?

Deltagare: Det är nog min favoritgenre, metroidvania spel och så vidare, så jag är mycket erfaren med dem.

Ledare: Okej.

Efterhandsintervju
Ledare: Nu ska jag fråga dig några frågor här. Först och främst, kände du igen några av de här mekanikerna från tidigare?

Deltagare: Ja, att springa för att hoppa längre, och dubbelhopp. Jag vet inte om jag har sett dubbelhopp på det här sättet, alltså att man ska attackera nedåt för att dubbelhoppa, men dubbelhoppet i sig har jag sett innan.

Ledare: Har du sett sprinting och basic attacks?

Deltagare: Ja, basic attacks har jag sett, och sprinting också. Att klona sig själv har jag sett på lite olika sätt. Och jag har sett att man interagerar med dörrar och andra sådana grejer genom att hålla ned knappar, och barriärer.

Ledare: Okej, har du sett *Charge Jump* innan i något spel?
Deltagare: Ja, det har jag. Jag minns inte exakt vilket spel just nu, men jag känner igen den.

Ledare: Okej, så den känner du igen. Känner du igen barrier-mekaniken då? Alltså väggar som bara låter vissa objekt åka igenom eller så?
Deltagare: Ja... allmänt sett alltså, ganska övergripande...

Ledare: Har du något exempel på det?

Deltagare: Jag tror inte att jag har sett just en klonad version av spelarkaraktären komma igenom vissa väggar. Det kanske är, alltså tidsbaserade pussel ifall det är typ time travel så har jag kanske väl sett det, ja.

Ledare: Okej, och vilken mekanik tyckte du var svårast att förstå?

Deltagare: Ja, jag var ju tvungen att kolla upp mekaniker tre gånger tror jag. Eftersom att jag inte experimenterade med dem så mycket som jag borde... Ja, egentligen så tror jag att sättet som man dubbelhoppar på i spelet är det mest säregna som jag inte sett, så jag hade

definitivt förväntat mig att man tryckte på hoppknappen två gånger, så jag skulle säga att det var det mest ointuitiva.

Ledare: Okej, tack så mycket. Då stoppar vi.

Appendix E - Deltagare 4 intervjuer

Förhandsintervju
Ledare: Så, först och främst då. Beskriv hur erfaren du är med spel.

Deltagare: Jag är medium erfaren skulle jag säga. Jag har spelat mer när jag var yngre än vad jag gör nuförtiden.

Ledare: Mhm, så hur mycket brukar du spela nuförtiden då? I timmar i veckan?

Deltagare: Oj, inte mycket, typ tre timmar i veckan kanske.

Ledare: Okej, hur mycket har du som mest spelat tidigare då? I veckan då alltså. Typ under ett sommarlov eller något.

Deltagare: Kanske… 40 timmar?

Ledare: Okej, hur erfaren är du med actionplattform-genren? Typ spel som Hollow Knight eller Rogue Legacy? Sådär 2D spel där man hoppar omkring och slåss mot grejer.

Deltagare: Inte jätte. Medium.

Ledare: Okej.

Efterhandsintervju
Ledare: Känner du igen några av de här mekanikerna sedan tidigare?
Deltagare: Innan det här testet, då alltså?

Ledare: Ja.

Deltagare: Sprint, ja den känner jag igen. Basic attack, ja den känner jag igen.

Ledare: Har du några exempel?
Deltagare: Basic attacks, det är väl typ alla spel som har att man ska slåss med combat eller sådär. Sprint känns också som en vanlig sådär 2D-platformer mekanik typ, och även 3D också antar jag. Bara man kontrollerar en karaktär så känns det som att man kan nästan alltid sprinta. *Charge jump* känner man ju igen till exempel från… typ Super Mario 64 fast det är ju inte riktigt samma sak fast typ när man gör ett bakåthopp eller sådär. Downward aerial, typ Super Smash Bros. kan man ju slåss nedåt, men det är inte exakt samma som det här i och med att man får lite boost sådär. Clone känner jag också igen, men den är ju lite som i Zelda typ, att man kan lägga objekt på switchar, och så kan man gå igenom en dörr när man aktiverat switchen, fast man använder sig själv istället. Switchar känner jag också igen, det var ju samma exempel som jag sa där med Ocarina of Time. Barriers… Jag vet inte exakt något exempel på spel som gör det här, jag kanske ska säga att jag inte känner igen den här då.

Ledare: Okej, och slutligen då. Vilken mekanik tyckte du var svårast att förstå, och varför?

Deltagare: Av de här då? Hmm… Den som var svårast att förstå var nog… Måste man säga någon?
Ledare: Om du inte har något svar på det som kan du svara på vad du tyckte var svårast att förstå i hela spelet istället.

Deltagare: Det svåraste att förstå var den där sista leveln, men om jag ska säga en mekanik… Alltså just den kombinationen av barriers och doors som var i den sista leveln. Att man kunde ta bort klonen genom att slå och hoppa med sin riktiga gubbe och så.

Ledare: Okej, tack!

Appendix F - Deltagare 5 intervjuer

Förhandsintervju
Ledare: Så, okej. Så innan vi börjar så vill vi ställa lite frågor då. Beskriv hur erfaren du är med spel, generellt sett.

Deltagare: Ja, Xbox har jag kört mycket, men de senaste fyra åren så har jag knappt spelat alls, alltså dator eller Xbox.

Ledare: Vilken sorts spel brukade du spela när du spelade?

Deltagare: Främst, det är väl First Person Shooter, heter det väl. Alltså CoD, GTA. Sedan så har man ju kört lite mobilspel också.

Ledare: Spelet som du kommer spela nu är ett actionplattformspel. Hur bekant är du med actionplattform-genren, typ spel som Hollow Knight, Rogue Legacy. Känner du igen någon av dem?

Deltagare: Nä... Alltså är det 2D eller?

Ledare: Ja.

Deltagare: Ah, väldigt lite. Jag har kört lite det där kända Super Smash, men ja... Väldigt lite.

Ledare: Okej, och när du brukade spela, hur mycket spelade du då ungefär? I timmar i veckan.

Deltagare: Ja, det var ju ett tag där där det var liksom varje dag lite grann. Ja, kanske två timmar per dag?

Ledare: Okej.

Efterhandsintervju
Ledare: Okej, nu har vi några till frågor här bara. Först och främst, kände du igen några av de här spelmekanikerna från tidigare spel som du spelat.

Deltagare: Ja, det gör jag... Ja det skulle jag säga. Inte just så här med en fyrkant men ja.

Ledare: Okej, mer specifikt, vilka av de här känner du igen? Kände du igen klonmekaniken från någonting annat som du spelat eller?
Deltagare: Det gör jag inte, den var bra, den var ny.

Ledare: Hur är det med nedåt-attacken i luften?

Deltagare: Ja... och det var ju det som jag var så himla dålig på i Smash, kom jag på. Då kunde man ju i luften byta riktning eller det var någonting sånt där som jag var väldigt dålig på, så ja, det var ju inte helt okänt för mig. Men den var också ny i den här typen av spel, skulle jag säga. Dörrar är ju klassiskt, men det där med tryckplattor är ju mer en grej kanske i ett spel där man är flera. Åh, vad hette det där spelet där man... Åh, kommer ju inte hjälpa att beskriva tror jag, men det handlade mycket om att man var två stycken. Det var två robotar.

Ledare: Ah, Portal 2.

Deltagare: Portal! Ja, precis. Då har jag för mig att det var lite sådär, men nu har ni ju clone-grejen så då kan man göra det själv. Och barrier har ju med clone-grejen att göra. Och både attacker och sprint är ju klassiska.

Ledare: Okej, en sista fråga då. Vilken av de här mekanikerna tyckte du var svårast att förstå, och gärna varför också.

Deltagare: Ja, jag fastnade ju på ett ställe, men det var ju egentligen för att jag inte läste texten utan bara tittade.

Ledare: Ah, på nedåt-attacken tänker du då eller? Jag minns att du försökte slå ned väggen.

Deltagare: Ja, just det, precis. Så ja, för det jag tänkte var att man bara kunde göra det på måltavlor. Så det var också det som jag tänkte såhär: “okej, men det är som att slå, fast i luften”. Jag tänkte inte vidare, så det lurade mig faktiskt. Men igen, efter att jag läste texten så fattade jag direkt att det har inte med måltavlan att göra utan det är bara “blast upward”. Men annars så var det ju lätt att förstå, det var ju bara att gå in här och kolla på grejerna om man inte hängde med, och så tror jag säkert att hade man testat knapparna själv utan menyn så hade man ju till slut klurat ut det också.

Ledare: Okej, toppen! Det var det.

Appendix G - Deltagare 6 intervjuer

Förhandsintervju
Ledare: Okej, först så ska vi ställa några frågor här då. Skulle du kunna berätta ditt kön och hur gammal du är?

Deltagare: Jag är en kvinna och jag är 25 år.

Ledare: Alright. Hur erfaren skulle du säga att du är med spel, generellt sett?
Deltagare: Jag skulle väl säga att jag är… Jag har ju spelat spel, men jag spelar ju inte jättefrekvent. Jag skulle nog säga att jag inte är särskilt erfaren.

Ledare: Okej, hur många timmar spelar du i veckan?

Deltagare: Ja, det är väl… Jag spelar ungefär 10 till 20 timmar i veckan.

Ledare: Vad spelar du?

Deltagare: Jag spelar Sims för det mesta.

Ledare: Hur mycket har du spelat som mest tidigare då? Under perioder av ledighet eller sommarlov eller så?

Deltagare: Ah, då spelade jag ju i princip hela dagen, i princip varje dag. Inte riktigt, men jag spelade typ bara tre olika spel.

Ledare: Okej, vilka spel var det?

Deltagare: Det var Dragon Age serien, det var Skyrim och det var Sims.

Ledare: Så, skulle du kunna beskriva hur erfaren du är med actionplattform-genren då? Typ spel som Hollow Knight eller Rogue Legacy där man hoppar omkring och slåss?

Deltagare: Alltså, inte alls. Jag suger på såna spel, jag har ju provat dem tidigare och kom fram till att "nä, fy". Jag kan liksom inte styra, och jag trillar ned hela tiden. Jag har inte gjort några större försök att bli bra på dem.

Ledare: Okej.

Efterhandsintervju
Ledare: Så nu har jag några fler frågor här. Vilka av spelets mekaniker kände du igen?

Deltagare: Ja, jag kände igen sprint från Skyrim. Attacker också, så klart.

Ledare: Okej, kände du igen någon annan mekanik?

Deltagare: Nej, jag tror inte det… Jag har för mig att vissa dörrar fungerar på samma sätt i Skyrim.

Ledare: Alright. Vilken mekanik tyckte du var svårast att förstå, och varför?

Deltagare: Det var nog nedåt-attacken. Jag funderar på om jag det var något som jag tyckte var svårt med klon… Nej, det tyckte jag nog var rätt så intuitivt. Ja, det var nog nedåt-attacken, den var svår att göra.

Ledare: Okej, tack så mycket.

Appendix H - Deltagare 7 intervjuer

Förhandsintervju
Ledare: Så, skulle du kunna börja med att säga ditt kön och ålder?

Deltagare: Jag är en man, jag är 51 år gammal.

Ledare: Okej, hur erfaren skulle du säga att du är med spel i allmänhet?

Deltagare: Nja, inte jätte, jag har ju spelat med mina söner när de var små men inte annars.

Ledare: Okej, så du brukar inte spela regelbundet då eller?
Deltagare: Nej, det brukar jag inte.

Ledare: Okej. Hur länge sedan var det du spelade då?
Deltagare: Ja, vad kan det ha varit? Det var väl 10 år sedan, nu ungefär.

Ledare: Vad spelade ni då, då?

Deltagare: Det var väl mest Mario och den där Zelda till Nintendo 64:an, tror jag.

Ledare: Har du spelat några spel inom actionplattform-genren tidigare? Typ spel där man hoppar på plattformar och slåss mot fiender?

Deltagare: Nja, det tror jag inte att jag har.

Ledare: Okej, nu kommer jag berätta hur spelet fungerar.

Efterhandsintervju
Ledare: Så nu har jag ett par frågor till att ställa dig, okej?

Deltagare: Ja, kör på.

Ledare: Först och främst, kände du igen några av de här mekanikerna sedan tidigare? Alltså från spel som du spelat tidigare då.

Deltagare: Ja, att man kan springa och hoppa känner man ju igen från Mario.

Ledare: Kände du igen attacker?
Deltagare: Ja, det gjorde jag. I Zelda så svingar han ju med svärdet.

Ledare: Okej, kände du igen någon annan mekanik?

Deltagare: Nja, det gjorde jag nog inte.

Ledare: Alright, och vilken mekanik i spelet tyckte du var svårast att förstå?

Deltagare: Ja du... Det var väl det där hoppet i luften.

Ledare: Okej, varför skulle du säga att du tyckte det?
Deltagare: Det var svårt att göra det, man vill ju hålla uppåt för att komma upp men man skulle hålla ner.

Ledare: Var det någon annan som du tyckte var svår?

Deltagare: Nej, det var det inte.

Ledare: Okej, det var nog allt då. Tack så mycket!

Deltagare: Ja då, inga problem.

Appendix I - Deltagare 8 intervjuer

Förhandsintervju
Ledare: Okay, so first we need you to answer some questions. So, could you please describe how experienced you are with games in general.

Deltagare: I'm bad. I'm really, really bad.

Ledare: So you don't play games in general?

Deltagare: No, nothing.

Ledare: Have you played before?

Deltagare: I mean, I remember playing this Ratchet and Clank when i was young. Other than that... I mean, I play some games on my phone, of course. Like word games, Word Star. But I rarely play.

Ledare: Alright. So, you said you've played in the past. Have you played for extended periods of time?
Deltagare: No, it's just like a friendly, playing with my brother kind of thing.

Ledare: Okay, so I take it you're not very experienced with the action platformer genre?

Deltagare: No. No, no, no, no, no. No.
Ledare: Alright, now I am going to describe the test to you.

Efterhandsintervju
Ledare: So, now we have a few closing questions. Did you recognize any of these game mechanics from previous games?

Deltagare: Yes, yes, yes. Oh, what's it called... Angry Birds? When you have to go past the obstacles?

Ledare: Flappy Bird?

Deltagare: Yes, Flappy Bird. I recognize Flappy Bird.

Ledare: Okay, which part of this game was like Flappy Bird to you?

Deltagare: The part when I had to jump.

Ledare: Oh, the Downward Aerial?
Deltagare: Yeah.

Ledare: Any other parts of the game that you recognized from other games?
Deltagare: The attacking one... That's like, every game, they attack. Like Ratchet and Clank, he attacks.

Ledare: That's a pretty common mechanic in most games. Did you recognize Sprinting? Like a button to go fast?

Deltagare: Maybe not exactly in this way, but like, the going faster button, maybe one time when i played some game it was like "run fast". I didn't know what I was doing, but yes.

Ledare: Alright, is that all of them?
Deltagare: Yes, that's all.

Ledare: Okay. Which mechanic did you find hardest to understand out of these ones?

Deltagare: Downward… Yes, this one. I don't like this one too much. The other ones were easy, the Downward Aerial was a lot.

Ledare: Yeah, okay.

Deltagare: That was it?
Ledare: Yup, that's it. Thank you!

Deltagare: Wow. This was a fun survey, I would do this experiment every day.