

TouchHockey

Ett Air-hockeyspel för touchbord

Jennifer Bedhammar
Aniisa Bihi
Malin Ejdbo
Elias Elmquist
Emilie Ho
Pontus Söderqvist
Peilin Yu

Examinator: Karljohan Lundin Palmerius

Sammanfattning

Denna rapport behandlar ett medietekniskt kandidatarbete inom spelutveckling. Syftet och målet med projektet var att skapa ett Air-hockey spel till ett touchbord som motsvarar ett verkligt Air-hockeybords storlek. I enlighet med kundens krav utvecklades spelet för att ställas ut i en utställningsmiljö. Liknande spel finns redan på marknaden, men dessa riktar sig främst mot mobiler med mindre skärmar. På ett touchbord finns möjligheten att utvidga spelupplevelsen med 3D-grafik och bygga vidare på spelmekaniken. Störst fokus i projektet låg i att skapa en unik spelupplevelse med tydlig återkoppling som ökar spelarnas förståelse av spelets funktionalitet.

Med önskemål från kursledningen genomfördes projektet i enlighet med Scrummetoden. Metoden byggde på fördelningen av arbetsuppgifter i samband med tid, prioriteringar och kontinuerliga förändringar hos kundkraven. Det agila arbetssättet utgick från uppdelningen av projektet i sprintar. Under projektets gång utfördes totalt åtta sprintar. Projekthanteringen skedde med hjälp av projekthanteringsprogrammet HacknPlan.

Spelet utvecklades i spelmotorn Unity med Microsoft Visual Studio som IDE och GitHub som versionshanteringsverktyg. Spelfysiken utvecklades av projektgruppen efter att ha undersökt olika lösningar, bland annat Unitys egna kollisionshantering. Designen och användargränssnittet inspirerades av det klassiska Air-hockeysspelet med utställningsmiljön och målplattformen i åtanke. Målplattformen var viktig då touchbordets storlek påverkade bland annat placeringen av objekten. För att förbättra spelarnas förståelse för spelet användes sedan olika former av återkoppling.

Resultatet av projektet är ett fungerande Air-hockey spel som spelas på ett touchbord. Likt vanlig Air-hockey kan två spelare spela mot varandra. Utöver den vanliga funktionaliteten av ett Air-hockey spel finns även power-ups som kan tas upp av spelarna, vilka antingen förenklar eller försvårar spelet.

En svårighet som projektgruppen stötte på var konflikter i GitHub mellan gruppmedlemmarna. Unitys metafiler skapade problem och konflikter när flera arbetade i samma scen och pushade upp ändringar. Den största svårigheten med projektet var dock kollisionshanteringen mellan klubborna och övriga föremål. Två olika versioner som löste detta problem utvecklades där det kollisionshanteringssystem som projektgruppen skapade från grunden valdes som det bättre alternativet. Unitys inbyggda kollisionshantering valdes bort på grund av begränsningarna i hur klubborna skulle styras av spelarna.

Touchbordet medförde begränsningar på grund av dess storlek och responsivitet vid touchinput. Storleksbegränsningen lösades genom att speldesignen utformades i hänsyn till målplattformen. För att göra spelet lättförståeligt användes återkoppling i form av färgkodning och text samt en design som liknar ishockey och det klassiska Air-hockeysspelet. Projektgruppen kunde inte påverka responsiviteten hos målplattformen.

Innehåll

| | |
|---|-------------|
| Sammanfattning | i |
| Figurer | v |
| Tabeller | vii |
| Typografiska konventioner | viii |
| 1 Inledning | 1 |
| 1.1 Bakgrund | 1 |
| 1.2 Syfte | 1 |
| 1.3 Frågeställning | 2 |
| 1.4 Avgränsningar | 2 |
| 2 Relaterat arbete | 3 |
| 2.1 Verklig Air-hockey | 3 |
| 2.2 Air-hockeyapplikationer | 4 |
| 2.3 2D Air-hockeyspel | 4 |
| 2.4 Användargränssnitt | 5 |
| 3 Utvecklingsprocess | 6 |
| 3.1 Utvecklingsmetodik och tidsplan | 6 |
| 3.2 Mötesrutiner | 7 |
| 3.3 Dokumentationsprinciper | 8 |
| 3.4 Versionshantering | 8 |
| 3.5 Kravhantering | 9 |
| 3.6 Projekthantering | 9 |
| 3.7 Testningsrutiner | 10 |
| 3.8 Användartester | 10 |
| 4 Teknisk beskrivning | 11 |
| 4.1 Grundläggande krav | 11 |

| | |
|---|------------|
| INNEHÅLL | iii |
| 4.2 Målplattform | 11 |
| 4.3 Utvecklingsmiljö | 12 |
| 4.4 Klasshierarki | 13 |
| 4.5 Spelfysik | 14 |
| 4.5.1 Rörelse | 14 |
| 4.5.2 Kollision | 15 |
| 4.6 Grafik | 16 |
| 4.7 Ljud | 16 |
| 4.8 Power-ups | 17 |
| 4.9 Optimering | 19 |
| 5 Användarupplevelse | 20 |
| 5.1 Användargränssnitt och menyer | 20 |
| 5.1.1 Startmeny | 22 |
| 5.1.2 Omstartsmeny | 22 |
| 5.2 Teman | 23 |
| 5.3 Resultat av användartester | 25 |
| 6 Resultat | 26 |
| 7 Analys och diskussion | 29 |
| 7.1 Metod | 29 |
| 7.2 Resultat | 30 |
| 7.2.1 Positionsbaserad styrning | 30 |
| 7.2.2 Power-ups | 30 |
| 7.2.3 Teman | 31 |
| 7.3 Arbetet i ett vidare sammanhang | 31 |
| 8 Slutsatser | 33 |
| 8.1 Svar på frågeställningarna | 33 |
| 8.1.1 Är Unitys inbyggda kollisionshantering ett fullgott alternativ för att implementera beteendet hos en simulering av Air-hockey, för spelbarhet och realism, eller krävs externa alternativ och i så fall varför? | 33 |
| 8.1.2 Hur uppnås, genom användargränssnitt och med hjälp av design och återkoppling, att en användare intuitivt kan uppfatta hur den ska interagera med det digitala Air-hockeyspelet? | 34 |
| 8.1.3 Vilka begränsningar för interaktion och speldesign medför ett touchbord som målplattform för ett digitalt Air-hockeyspel? | 34 |
| 8.2 Slutsats | 34 |
| Litteraturförteckning | 35 |

| | |
|--|-----------|
| A Ansvarsfördelning och individuella bidrag | 37 |
| B HacknPlan statistik | 38 |
| C Användartester - Frågor | 39 |
| D Användartester - Sammanställning | 40 |
| D.1 Första intrycket | 40 |
| D.2 Responsivitet | 40 |
| D.3 Generellt om grafiken | 40 |
| D.4 Godistemat | 41 |
| D.5 Fotbollstemat | 41 |
| D.6 Neontemat | 41 |
| D.7 Generellt om användargränssnittet | 41 |
| D.8 Ljud | 41 |
| D.9 Omstartsmenyn | 42 |
| D.10 Power-ups | 42 |
| E Användartester - Svartsformulär | 43 |

Figurer

| | | |
|-----|---|----|
| 2.1 | Ett klassiskt Air-hockeybord [3]. | 3 |
| 2.2 | Klassiskt Air-hockey spel anpassat till en iPad [4]. | 4 |
| 2.3 | Instruktionsvideo för ett tvådimensionellt Air-hockey spel. | 5 |
| 3.1 | Tidsplan över projektet i form av ett Gantt-schema. | 7 |
| 3.2 | Bild på hemsidan HacknPlan som användes för projekthantering. | 9 |
| 4.1 | Touchbordet som var målplattformen. | 12 |
| 4.2 | En översikt på spelets arkitektur. | 13 |
| 4.3 | Översikt på spelets hierarki. | 14 |
| 4.4 | De 3D-modeller som användes till pucken, klubban och spelplanen. | 16 |
| 4.5 | Skölden dyker upp vid spelarens mål när denne tar upp en sköld-power-up. I detta fall har den blå spelaren plockat upp en power-up som skapar en sköld som skyddar den blå spelarens mål från den röda spelaren. | 17 |
| 4.6 | En power-up som minskar klubban är aktiv på den röda spelaren och en power-up som förstorar klubban är aktiv på den blå spelaren. | 18 |
| 4.7 | Den slutgiltiga designen på de tre power-up objekten. Den vänstra är en förstora-power-up, mittersta är en sköld-power-up och den högra är en minska-power-up. | 18 |
| 5.1 | Spelplanen när spelet startar. En timer räknar ner i tre sekunder innan pucken släpps. I bilden syns att det är två sekunder kvar innan spelet startar. | 20 |
| 5.2 | En timer räknar ner de sista fem sekunderna. I bilden syns att det endast är två sekunder kvar innan spelet avslutas. | 21 |
| 5.3 | Startmeny. | 22 |
| 5.4 | Tidig skiss på omstartsmenyn. | 22 |
| 5.5 | Det klassiska temat. Inspirerad av det klassiska spelet. | 23 |
| 5.6 | Neontemat. Klubborna och pucken är självlysande. | 24 |
| 5.7 | Fotbollstemat. | 24 |
| 5.8 | Godistematz med mer pastellfärgar. | 25 |
| 6.1 | Översikt över spelplanen och dess komponenter. 1. Klubbor som spelarna kan kontrollera. 2. Nedräkning till när pucken släpps. 3. FPS-räknare. 4. Poäng för röd spelare. 5. Poäng för blå spelare. 6. Återstående tid tills spelet avslutas. 7. Mål. 8. Antal touchinputs. | 26 |

| | | |
|-----|---|----|
| 6.2 | Pucken, klubban och spelplanen med texturer applicerade i det klassiska temat. Pucken byter sedan färg till rött eller blått beroende på vilken spelare som sist slog till den. | 27 |
| 6.3 | Exempel på några power-ups under en spelomgång. 1. Power-up objekt. 2. Puck träffad av blå spelare och har bytt färg. 3. Aktiv minska-power-up effekt på den blå spelaren. 4. Sköld som skyddar blå spelarens mål. 5. Aktiv förstora-power-up effekt på den röda spelaren. | 27 |
| 6.4 | Omstartsmeny efter ett oavgjort spel. | 28 |
| B.1 | Statistik i HacknPlan. De färgade områdena representerar de olika kategorierna som fanns under projektet. Det orangea området symbolisera dokumentation och rapportskrivning, lila är programmering, röd är buggar, ljusblå är design, grön är modellering, grön-blå är användartester, rosa är ljuddesign och brun är idéer. | 38 |
| E.1 | Svar på fråga 1. | 43 |
| E.2 | Svar på fråga 2. | 43 |
| E.3 | Svar på fråga 3. | 44 |
| E.4 | Svar på fråga 4. | 44 |
| E.5 | Svar på fråga 5. | 44 |

Tabeller

| | | |
|-----|--------------------------------------|------|
| 1 | Ordlista | viii |
| 3.1 | Lista över milstolparna i projektet. | 7 |

Typografiska konventioner

Engelska ord som inte finns i given ordlista kommer förklaras inom parentes i texten och markeras i kursivt. Detta då ingen direkt översättning finns på dessa ord och förklaras därmed tydligare på engelska. En ordlista över återkommande ord kan ses i tabell 1.

Tabell 1: Ordlista

| Ord | Betydelse |
|--------------------------|---|
| <i>Touch</i> | Beröra/Vidröra |
| <i>Touchinput</i> | Pekskärmsinmatning |
| <i>Touchbord</i> | Interaktivt bord som reagerar vid beröring |
| <i>Air-hockey</i> | Luft-hockey |
| <i>Power-up</i> | Bonusobjekt |
| <i>Frame</i> | Bildruta |
| <i>FPS</i> | Frames per second - bilder per sekund |
| <i>Scrum</i> | Utvecklingsmetod |
| <i>Sprint</i> | Kort tidsperiod där uppgifter utförs |
| <i>Backlog</i> | Lista på uppgifter som väntar på att utföras/prioriteras |
| <i>Stand-up</i> | Kort möte under dagen där varje projektmedlem ställer sig upp och berättar kort vad de jobbar med |
| <i>Masterbranch</i> | Huvudgrenen med den senaste fungerande och väldokumenterad version |
| <i>Push</i> | Lägger upp ändringar på Masterbranchen |
| <i>Global Repository</i> | GitHub-mapp som innehåller alla filer i projektet på Githubs server |
| <i>Pull Request</i> | En begäran att få uppdatera masterbranchen, som måste godkännas av en annan projektmedlem |
| <i>Metafil</i> | En fil som innehåller information om en annan fil |
| <i>Script</i> | En Unity komponent som innehåller kod för att kontrollera element i scenen |
| <i>GameObjects</i> | En Unity komponent som defineras som ett element i en scen |

Kapitel 1

Inledning

Inom ramarna för kursen TNM094 har ett medietekniskt kandidatarbete genomförts. Projektgruppen bestod av sju studenter som läste sin sjätte termin på utbildningen Civilingenjör i Medieteknik vid LiU på Campus Norrköping. Projektet hade en intern kund som var en del av kursens fakultet. Kunden ställde inledande krav på den sluttgiltiga produkten.

Denna rapport beskriver planeringen och utformandet av en produkt som utvecklats av tidigare nämnd projektgrupp. Det som behandlas i detta kapitel är bakgrund, syfte, frågeställningar och avgränsningar.

1.1 Bakgrund

I en allt mer teknikberoende värld utformas projekt och produkter som ständigt tar tekniken till en ny nivå. Det ingår i det vardagliga livet att äga en smarttelefon eller pekplatta, dessa enheter är bara några av de få som använder sig av funktionen touch. Det behövs inte längre knappar eller pekpinnar när teknik som touch existerar.

Med hjälp av touch återskapas allt fler sällskapsspel i digitalt format. Delvis för att tekniken fortsätter utvecklas men även för att spelens blir mer lättåtkomliga och nya möjligheter uppstår. Spelen får nya funktioner, ljudeffekter och bättre grafik. Detta gör det möjligt att modifiera funktioner samt spelregler på ett ögonblick. Vilka spel kan tänkas digitaliseras på grund av dessa anledningar? I detta fall är det Air-hockey, som beskrivs mer i avsnitt [2.1](#).

Kundens begäran var att projektgruppen skulle skapa ett Air-hockey spel som skulle vara spelbar i utställningssyfte. Målet med den begäran var att slutprodukten ska kunna stå i någon form av utställningslokal öppen för allmänheten. Spelet skulle köras på ett så kallat touchbord, en pekskärm som motsvarar storleken av ett verkligt Air-hockey spel. Utöver detta ska spelet erbjuda olika grafiska teman och spellägen. Den aspekt i spelet som var viktigast var att spelet gav bra återkoppling så att användarna kände sig involverade i spelet.

1.2 Syfte

Syftet med projektet är att undersöka utvecklingsmöjligheter för ett Air-hockey spel till touchbord. Air-hockeyspelet ska efterlikna egenskaperna hos ett verkligt Air-hockey spel, bland annat i spelfysik, design, interaktion och ljud. Spelet ska även bygga vidare på konceptet och skapa en mer varierad spelupplevelse med användargränssnitt och grafik.

1.3 Frågeställning

Följande frågeställningar låg till grund för projektet:

- Är Unitys inbyggda kollisionshantering ett fullgott alternativ för att implementera beteendet hos en simulering av Air-hockey, för spelbarhet och realism, eller krävs externa alternativ och i så fall varför?
- Hur uppnås, genom användargränssnitt och med hjälp av design och återkoppling, att en användare intuitivt kan uppfatta hur den ska interagera med det digitala Air-hockeyspelet?
- Vilka begränsningar för interaktion och speldesign medför ett touchbord som målplattform för ett digitalt Air-hockeyspel?

1.4 Avgränsningar

Eftersom utställningsmiljöer är anpassade för mäniskor i alla åldrar är produkten inte skapad för någon specifik målgrupp, utan anpassad till en bred publik och ska därför vara lättförståelig oavsett ålder. Ingen tidigare kunskap eller erfarenhet av ett Air-hockeyspel ska krävas av användaren, vilket krävde att spelets gränssnitt skulle vara tillräckligt enkelt för att en förbipasserande gäst på utställningen ska kunna sätta sig in i spelet direkt. Spelet beslutades av kunden att vara på svenska, vilket är en begränsning för ej svenska talande spelare. Då projektet handlar om ett spel i en utställningsmiljö finns specifika avgränsningar för tillgänglighet. Spelet kommer ej släppas som en självständig applikation, utan fokus ligger på utställningen.

En fysisk klubba anpassad för touchbord är inte nödvändig då spelet reagerar på antingen finger- eller handberöring. Spelet utgår från två spelare åt gången. Dessa spelare kan spela matcher under ett bestämt tidsintervall utan att behöva hålla poängen själva. För att minska kraven på spelaren kommer de inte kunna ändra tema själva utan det kommer ändras automatiskt vid omstart. Eftersom detta spel utvecklats i utställningssyfte utan funktioner som utesluter någon samhällsgrupp eller liknande, saknas koppling till etiska eller samhälleliga aspekter.

Kapitel 2

Relaterat arbete

Air-hockey är inte ett nytt spel utan har funnits sedan 1969 [1]. Det är detta klassiska spel som projektet bygger på. Detta avsnitt tar upp liknande spel som redan finns och som projektgruppen tog inspiration från under utvecklandet av produkten.

2.1 Verklig Air-hockey

Det spel som projektgruppen tog mest inspiration av var det klassiska och fysiska Air-hockeyspelet. Spelet uppfanns 1969 när det var väldigt populärt med bordsspel och släpptes sedan för allmänheten 1970 [1]. Det blev en succé och redan 1973 bildades de första lokala förbunden [2].

Air-hockey är ett spel där två personer spelar mot varandra på ett stort rektangulärt luftbord, se figur 2.1. Spelet ska efterlikna ett ishockey spel i miniatyr som kan spelas mellan två spelare. Båda spelarna har en varsin ”klubba” som används till att slå till en platt ”puck” ner i motståndarens mål. Målen är utformade som hålöppningar på vardera kortsida av bordet. Bordsytan har hundratals små hål som med hjälp av en fläkt tvingar luft att komma ut ur hålen. Med hjälp av luften under pucken minskar friktionen, vilket ökar hastigheten på pucken.

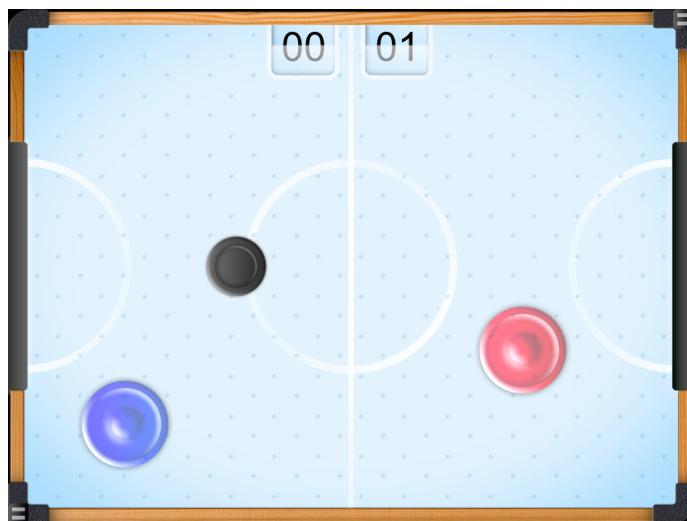


Figur 2.1: Ett klassiskt Air-hockeybord [3].

Air-hockey är ett spel som många vuxit upp med och har spelat i arkadhallar. Dock är arkadhallar inte längre lika populärt att besöka idag, detta på grund av att det nu är möjligt att ladda ner spel gratis eller för några fåtal kronor på telefonen i bekvämligheten av sitt egna hem. Nu förtiden kan det finnas Air-hockeybord på restauranger, barer och spelarkader, men det är mer populärt att spela spel på datorer, spelkonsoler eller mobila enheter som har stöd för 3D-grafik.

2.2 Air-hockeyapplikationer

Att återskapa klassiska arkadspel till mobila enheter är populärt och Air-hockey är inget undantag. Det finns många olika Air-hockeyspel som idag är anpassade till mobila enheter och utvidgar det klassiska spelets spelmekanik, ett exempel på en sådan app kan ses i figur 2.2. De olika apparna inriktar sig inom olika aspekter av spelet och har utvidgat dessa. I en app kan spelarna till exempel plocka upp power-ups som kan vara till en fördel eller en nackdel i spelet, medan i en annan app kan det finnas ett nivåsystem med olika banor som läses upp under spelets gång, och olika teman blir tillgängliga.

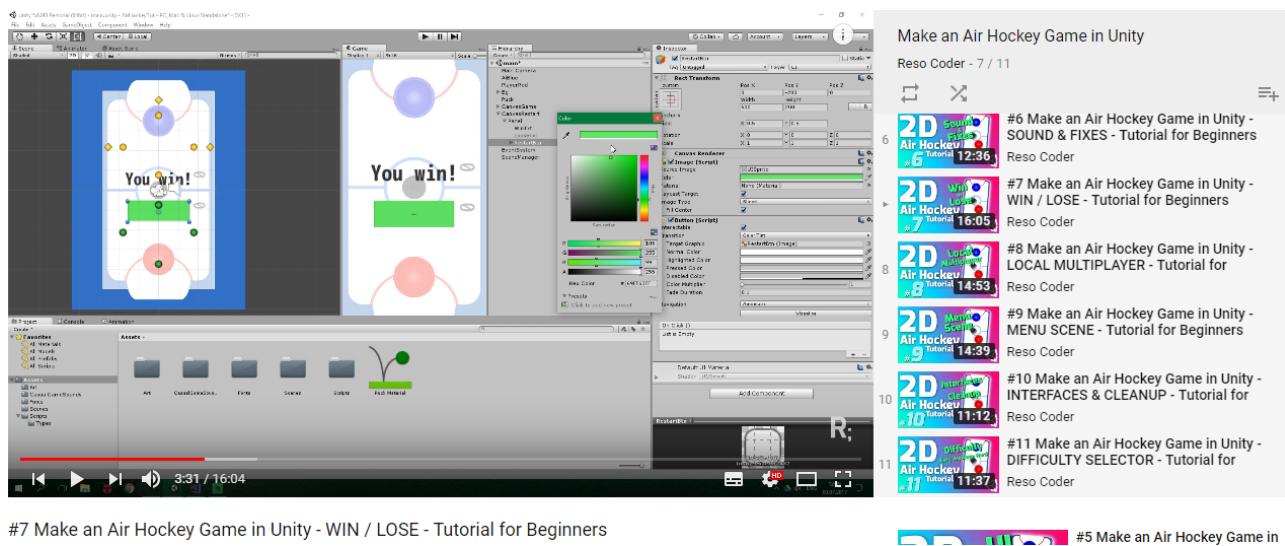


Figur 2.2: Klassiskt Air-hockey spel anpassat till en iPad [4].

Innan projektarbetets start hade några gruppmedlemmar erfarenhet av dessa applikationer, och inspirerades av att implementera liknande lösningar i projektet. Visionen var att inte bara simulera ett Air-hockey spel, utan att även bygga vidare på konceptet för att göra en spelomgång mer dynamisk.

2.3 2D Air-hockey spel

Bland det första som gjordes i projektet var att göra en förstudie på vilken spelmotor som skulle användas. En viktig aspekt var att det skulle vara möjligt att skapa ett fungerande tredimensionellt Air-hockey spel med spelmotorn. De spelmotorer som jämfördes var Unreal Engine 4 [5] och Unity [6], mycket på grund av att några av projektmedlemmarna tidigare hade arbetat med dessa. En stor anledning till att Unity valdes var på grund av att bra dokumentation återfanns. Den dokumentation som hittades var en videoserie, se figur 2.3, där programmeraren Resocoder [7] förklarar i detalj hur ett tvådimensionellt Airhockey-spel kan utvecklas i Unity.



Figur 2.3: Instruktionsvideo för ett tvådimensionellt Air-hockey spel.

De olika stegen för implementeringen av spelet förklarades och skriven kod återfanns i samband med videon för att enkelt följa utvecklingen. De olika aspekterna av spelet var även uppdelad i olika videoklipp, vilket underlättade att implementera de önskade delarna av spelet. Instruktionerna i videoklippens följdes, vilket resulterade i ett komplett spel som projektgruppen kunde ta inspiration från. Det fastställdes också att det var möjligt att skapa ett Air-hockey spel i Unity.

2.4 Användargränssnitt

Förutom att ha studerat användargränssnitt på andra appar och undersökt det verkliga Air-hockeybordet och touchbordet, tittade projektgruppen tillbaka på en gammal kurs, TNM040 Kommunikation och användargränssnitt [8]. Föreläsningsanteckningar och föreläsningsbilder från kursen, samt rekommenderad kurslitteratur [9] användes för att undersöka designprinciper som kunde hjälpa i designen av användargränssnittet och bidra till den intuitiva förståelsen som gruppen sökte.

Från kursdokumenten och litteraturen framgår det bland annat att vissa saker är viktiga att ha i åtanke vid design av gränssnitt. En god design ska ha klar affordans, tydliga indikatorer, god synlighet, mappning och återkoppling [9]. Användargränssnittet designades med detta i åtanke för att skapa en bra spelupplevelse för användaren.

Kapitel 3

Utvecklingsprocess

I ett stort projektarbete med uppsatta krav och mål är struktur viktigt. Detta avsnitt beskriver den agila arbetsmetoden som användes under projektet, vilka rutiner som fanns och vilka hjälpmittel som användes.

3.1 Utvecklingsmetodik och tidsplan

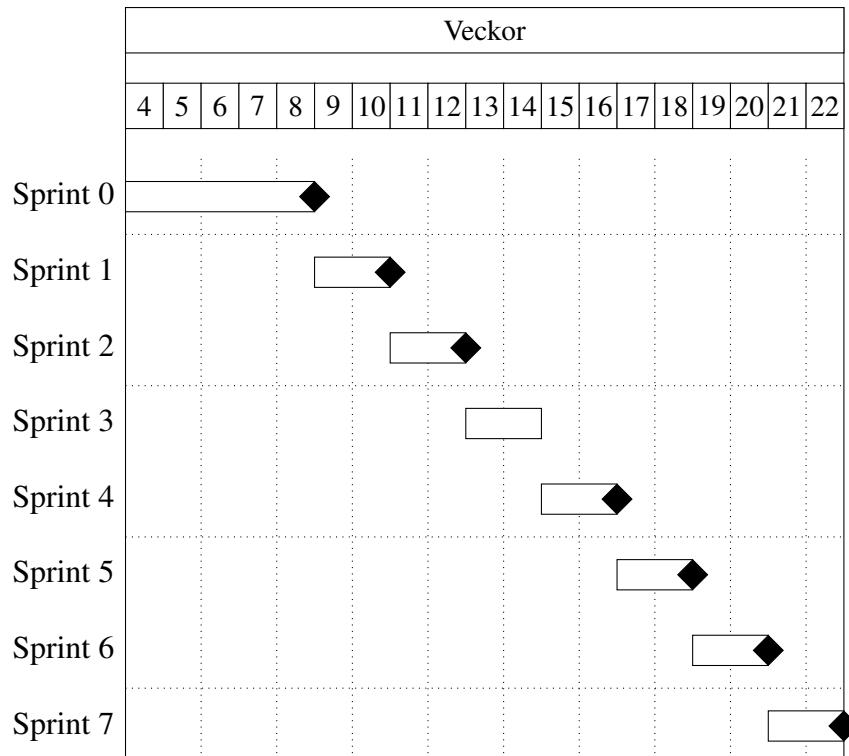
Detta projekt utgick från en agil [10] systemutvecklingsprocess. Agil utveckling skapar en nära kontakt och samarbete mellan kunden och projektgruppen. Med kontinuerliga kundmöten genom hela utvecklingsprocessen motverkades eventuella missförstånd och projekthinder. Den nära kundkontakten under projektet skapar även en positiv inställning till förändringar och nya idéer. I detta projekt användes den agila utvecklingsmetoden Scrum [11], som fokuserar på möjligheten att strukturera förändringar kan ske. Scrum utgår från en levande lista med önskemål där tidigare prioriterade önskemål kan ersättas med nya [11]. Projektgruppen arbetade mestadels individuellt men satt tillsammans, parprogrammering tillämpades när det var nödvändigt. I bilaga A beskrivs ansvarsfördelningen i projektgruppen och individuella bidrag för varje gruppmedlem.

Projektgruppen använde sig som tidigare nämnts av Scrum [11] med start från sprint 0. I sprint 0 skapades den grundläggande planeringen över projektet. Gruppen gjorde även efterforskningar om de metoder och tekniker som skulle användas samt tog fram kravspecifikationer efter möte med kunden. Utifrån dessa krav kunde övergripande uppgifter bestämmas och en planering över de kommande sprintarna togs fram. Detta blev grunden för tidsplanen med milstolpar, leverabler och slutdatum för projektet. Varje sprint, förutom sprint 0, varade i två veckor.

I tabell 3.1 visas de milstolpar som förekom i projektet. De är markerade i form av svarta romber i Gantt-schemat i figur 3.1, där alla planerade sprintar för projektet har illustrerats.

Tabell 3.1: Lista över milstolparna i projektet.

| Nr. | Mål med milstolpen |
|-----|---|
| 1 | Kravspecifikationer, övergripande planering över sprints. |
| 2 | Grundläggande touchfunktioner och grundfysiken för Air-hockey. |
| 3 | Första prototyp klar. |
| 4 | Grundläggande grafik, speldesign och användargränssnitt. |
| 5 | Slutgiltig spelprototyp redo för användartester. |
| 6 | Justeringar efter tester klart. |
| 7 | Projektrapport och produkt med uppnådd kravspecifikation redo för leverans. |



Figur 3.1: Tidsplan över projektet i form av ett Gantt-schema.

Leverabler i projektet var de olika spelprototyperna som levererades i samband med sprintarna och milstolparna, samt projektplanen och projektrapporten.

3.2 Mötesrutiner

På första måndagen i varje sprint skedde först en kort återblick av föregående sprint, därefter planerades den aktuella sprinten. Google Kalender [12] användes under planeringen för att schemalägga sprintens kommande möten och arbetstillfällen. Kalendern integrerades med gruppmedlemmarnas individuella kalendrar vilket gjorde det enkelt att ha bra struktur i upplägget av arbetet. Projektgruppen tog även upp vilka uppgifter som fanns och beslutade vilka som skulle utföras i den aktuella sprinten. De uppgifter som valdes ut flyttades till sprintbackloggen och delades upp bland gruppmedlemmarna.

En sprint avslutades med ett retrospektivt möte där resultatet av sprinten utvärderades och vad som kunde förbättras inför nästa sprint diskuterades.

Under varje arbetsdag förkom ett, ibland två, informella möten. Första mötet hölls på morgonen och var återkommande, det andra hölls på eftermiddagen vid behov. Mötestiderna varierade beroende på schemat och varade i genomsnitt i en halvtimme. På morgonmötet gick gruppen igenom vad alla medlemmar hade för uppgift för dagen och lite kort om vad som gjorts föregående arbetsdag. Eftermiddagsmötet bestod av ett stand-up möte eller ett avslutningsmöte där rapportering av arbetet samt eventuella hinder och problem som förekommit togs upp. Detta var ett bra tillfälle att be om hjälp eller diskutera viktiga punkter.

Projektgruppen hade även möten med kunden under projektet, dessa kundmöten förekom ungefär varannan sprint. Mötesprotokoll eller skriftliga sammanfattningar på vad som diskuterades och beslutades fördes av sekreteraren.

3.3 Dokumentationsprinciper

Det fanns flera kodprinciper som projektgruppen följde. Koden skulle ha en viss struktur där olika stycken hade olika funktioner. Ordningen av koden skulle dessutom vara enkel att följa och variabel-, fil- och funktionsnamn skulle följa en viss standard. Denna standard var att variabler, filer och funktioner skulle ha ett namn som beskrev vad syftet med variabeln, filen eller funktionen var på ett tydligt sätt. Namnen skulle hellre vara långa och förståeliga än korta och obegripliga.

Dokumentation av koden skrevs översiktligt under tiden som koden skrevs. En fullständig dokumentation skrevs när projektgruppen var nöjd med koden och resultatet. Den fullständiga dokumentationen granskades sedan av en annan person i projektgruppen, som inte varit delaktig i koden, och återkoppling gavs. När den som granskade koden tyckte det såg bra ut godkändes den.

Mötesprotokoll och andra dokument som inte var relaterade till kod dokumenterades via en Google Drive-mapp [13] som delades med alla i projektgruppen. Dokumentansvarig och sekreterare hade ansvar för att strukturera denna mapp och se till att alla andra gruppmedlemmar upprätthöll den strukturen. Mötesprotokoll sparades i pdf-format efter mötets slut för att fastställa mötet. De möten som protokolfördes var sprint-startmöten, -avslutsmöten, kundmöten och första morgonmötet för veckan. Större beslut relaterade till projektet som togs utanför mötestid lades till i en punkt på nästkommande mötesdagordning så att det dokumenterades officiellt.

3.4 Versionshantering

Den spelmotor som användes i projektet var Unity. Unity har ett inbyggt versionshanteringsverktyg som gruppen övervägde att använda. Eftersom projektgruppen hade tidigare erfarenhet av GitHub [14] valdes detta som versionshanteringsverktyg istället för Unitys inbyggda. På GitHub fanns ett global repository som alla gruppmedlemmar hade tillgång till.

Eftersom både Unity och Github skulle användas under projektet behövde GitHub ställas in att ignorera Unitys metafiler [15]. Detta var nödvändigt då det annars kunde skapa konflikter mellan olika gruppmedlemmars metafiler. När fler än en person försökte pusha upp ändringar i projektet till GitHub samtidigt kunde dessa konflikter uppstå. Dessa konflikter kunde undvikas genom att ställa in Github och Unity att fungera tillsammans och ställdes in enligt en guide av Philipp, Mark [15].

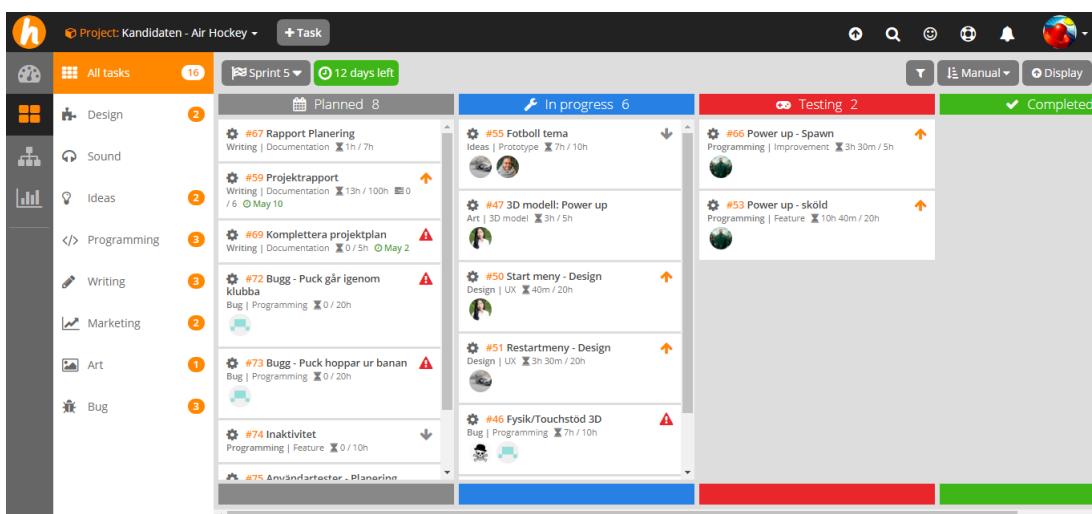
De rutiner som fanns för versionshantering var att gruppmedlemmarna varje dag hämtade ner de senaste ändringarna från GitHub. Alla kommentarer på GitHub skulle vara skrivna på engelska, detta på grund av att det var smidigast om koden skulle vidareutvecklas. Om en konflikt uppstod var det versionshanteringsansvariges ansvar att detta löstes innan projektet pushades upp till det globala repository som gruppen jobbade mot. Innan någon i projektgruppen pushade upp ändringar skulle denne meddela detta till resterande i projektgruppen. Detta skapade en bättre kommunikation mellan gruppmedlemmarna och minskade antalet konflikter i GitHub.

3.5 Kravhantering

För att få fram krav på systemet skedde kundmöten. Kraven och önskemålen från kunden förvarades i produkt-backloggen, där alla krav omsattes till uppgifter. För en bättre och tydligare struktur kunde kraven även delas upp i olika områden och i flera deluppgifter. I backloggen var de högst listade uppgifterna de som kunden ansåg vara viktigast, därför var dessa först med att hämtas till sprint-backloggen. De återkommande kundmötena såg till att kunden kunde komma med synpunkter under utvecklingens gång samt justera kraven om så önskades efter demonstration av produkten. Kundens åsikter och önskemål kunde eventuellt ändra kravspecifikationen och behövde därmed kunna uppdateras. Projektgruppen hade ansvaret att ställa alla nödvändiga frågor som krävdes till kunden för att få en djupgående förståelse av dennes önskemål.

3.6 Projekthantering

HacknPlan [16] användes för projekthantering och länkades samman med GitHub för spårning. HacknPlan valdes då det är ett projekthanteringsprogram anpassad för spelutveckling. På HacknPlan skapades relevanta uppgifter som publicerades innan varje sprint påbörjades. Uppgifterna kategoriseras enligt HacknPlans färdiga kategorier och underkategorier, till exempel ”*Design | UX*” och ”*Programming | Improvement*”. Under sprinten markerades uppgifterna som planerade, pågående, i testning och klara beroende på vilket tillstånd de befann sig i, se figur 3.2. Gruppmedlemmarna kunde även skriva upp sig och lägga in arbetad tid på de olika uppgifterna för att visa vem som arbetade på vad och därmed underlätta kommunikationen. När en uppgift var klar kunde alla i projektgruppen på så sätt se vem som slutfört den och hur mycket tid som lagts ned på den specifika uppgiften. Statistik över hur mycket tid som lades ner på de olika kategorierna kan ses i bilaga B.



Figur 3.2: Bild på hemsidan HacknPlan som användes för projekthantering.

På sprintavslutsmöten gick projektgruppen igenom alla uppgifter i sprintbackloggen, sammanfattade arbetet och såg till att alla uppgifter var rätt markerade i HacknPlan. Om en uppgift varit i testning kunde den exempelvis flyttas till klar om projektgruppen ansåg att den klarat testerna. Uppgifterna som var markerade rätt i planerade, pågående eller i testning fördes över till nästa sprint i HacknPlan.

3.7 Testningsrutiner

För att säkra kvaliteten i projektet krävdes kontinuerlig testning. Tester som utfördes var regressions-tester, prestandatester och användartester. Även kodgranskning användes för att säkerställa kvalitén på koden.

Regressionstester genomfördes varje gång nya moduler sammankopplades med systemet, testerna genomfördes av utvecklaren själv när spelet testkördes i Unity. Detta säkerställde att varje ny ändring var relevant och inte skapade problem för resten av systemet. Ett prestandatest utfördes även i samband med regressionstester med hjälp av Unitys profileringsverktyg [17] där utvecklarna bland annat kunde undersöka processor- och minnesanvändning.

Testningsansvarig ansvarade för att skapa användartester och såg till att dessa genomfördes. Använtartester genomfördes när en fungerande produkt tagits fram för att få förslag på möjliga förbättringar, detta genom att testa hur spelet upplevdes av användaren. En fungerande produkt innebar i detta fall att alla essentiella funktioner var implementerade och stabila under körning. Använtartesterna bestod av speltest samt digitala formulär.

Kodgranskning skedde varje gång en funktionalitet ansågs vara färdig. Den specifika uppgiften flyttades då till testningsloggen i HacknPlan och en pull request gjordes i GitHub. Denna granskning genomfördes av en annan gruppmedlem för att säkerställa att koden var lättförståelig och buggfri. Pull requests implementerades sent i projektet och tidigare granskades koden direkt på plats av en annan gruppmedlem. Gruppen arbetade större delen av tiden tillsammans, vilket gjorde detta möjligt. Om koden ansågs hålla förväntad kvalité efter granskning och återkoppling kunde den pushas upp till GitHub. Storleken på varje uppgift i sprintarna skattades och parprogrammering användes där det ansågs nödvändigt som en form av kontinuerlig testning.

3.8 Använtartester

Ett Google-formulär skapades för personer inom målgruppen att svara på. Detta formulär ställde frågor om de förväntningar användarna hade på ett Air-hockey spel, hur det skulle se ut och fungera. Detta utfördes för att kunna ta beslut om viss funktionalitet och designfrågor, till exempel om en spelare skulle ha möjlighet att gå över till den andra spelarens planhalva med sin klubba. Men även när det ansågs att det var ett mål och när spelet var slut. När tillräckligt många svar hade kommit in tog projektgruppen beslut om viss funktionalitet och design. En kort sammanfattning över svaren på detta formulär kan ses i avsnitt 5.3, och en fullständig sammanställning finns i bilaga E.

Även användartester i form av att användare fick spela spelet och säga vad de tyckte förekom. Under dessa användartester var det endast två gruppmedlemmar närvarande och det fanns en fast lista med frågor som projektgruppen ställde till användaren. Frågorna som ställdes under användartesterna presenteras i bilaga C och svaren sammanfattas kort i avsnitt 5.3. En fullständig sammanställning av alla användartester presenteras i bilaga D.

Kapitel 4

Teknisk beskrivning

Detta avsnitt beskriver mer detaljerat den tekniska aspekten av produkten, vilka systembegränsningar som fanns för systemet och hur vissa tekniska aspekter i produkten implementerades.

4.1 Grundläggande krav

Dessa var de krav som kunden ställde till projektgruppen:

- Minst två spelare
- 60 FPS uppdateringshastighet
- 2,5D grafik
- Två teman
- Stöd för ljud
- Spelet ska vara på svenska
- Ingen tidigare spel erfarenhet krävs av användaren

De grundläggande kraven för produkten var att det skulle ha stöd för två spelare likt ett verkligt Air-hockey spel, och det skulle vara på svenska. Grafikmässigt skulle spelet köras optimalt på målplattformen (60 FPS) samt innehålla 2,5D Grafik, 3D-grafik sett från ett tvådimensionellt perspektiv. Två stycken teman skulle inkluderas i spelet för att demonstrera hur en digital version av ett verkligt spel kan anta flera olika former. Ljud skulle även förekomma i spelet för att ge återkoppling av olika moment i spelet.

4.2 Målplattform

Målplattformen var ett touchbord som under projekttiden befann sig på Campus Norrköping. Touchbordet bestod av en stationär dator, kopplad till en ca 40 tum-TV med touch-möjligheter. Allt detta var sammanbyggt i en metallkonstruktion. Bildförhållandet på skärmen var 16:9, vilket förekommer på majoriteten av TV-skärmar. Detta vanligt förekommande bildförhållande liknar även formen på ett Air-hockeybord, vilket underlättade utvecklandet.

Två varianter av touchbord fanns att tillgå. Ett bord hade en kapacitiv pekskärm likt dagens mobiltelefoner, där kapacitansen på ytan påverkas av fingerrörelse. Detta touchbord visas i figur 4.1.



Figur 4.1: Touchbordet som var målplattformen.

Det andra touchbordet hade infraröda sensorer längs skärmramen. Ett problem som märktes tidigt under projekttiden var att det fanns fördräjning när det gällde touchinput på båda touchborden. På grund av att touchbordet med infraröda sensorer hade högre fördräjning valdes målplattformen till den kapacitativa touchbordet som ses i figur 4.1. Konsekvenserna av fördräjningen diskuteras vidare i avsnitt 7.2.

4.3 Utvecklingsmiljö

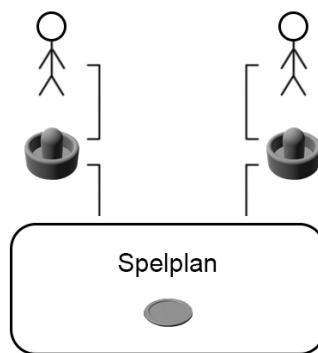
Spelet utvecklades i spelmotorn Unity [6] med Microsoft Visual Studio [18] som *IDE* (Integrated development environment) och kodspråket som användes var C#. Unity valdes bland annat då spelmotorn stödjer både MacOS och Windows, och har en omfattande dokumentation med intruktionsvideos för spelutveckling. Unity användes även under förstudierna för att implementera en 2D-version av spelet, se avsnitt 2.3, och omvandlingen till 3D kunde således implementeras smidigt i samma spelmotor. Visual Studio valdes då den medför bra verktyg för kompilering och felsökning av koden. Unity erbjuder även utvecklaren att installera Visual Studio under installationen av spelmotorn.

Systemet som byggdes var uppdelat i tre områden, dessa utvecklades oberoende av varandra. De tre områdena var användarinput, spelutveckling och användargränssnitt. Delarna byggdes ihop i spelmotorn med dess inbyggda funktioner. Området gällande användarinput relaterade till hur användaren interagerar med programmet, så som touchfunktionalitet och kontroll av spelobjekten.

Spelutvecklingsområdet innehöll arbete som har med grafik, ljuddesign och speldesign att göra. Att arbeta med grafiken innebar att skapa 3D-modeller, texturer och animera. 3D-modellerna skapades i Autodesk 3DS Max [19] samt Blender [20] och texturer skapades i Adobe Photoshop [21], som samtliga projektmedlemmar använt i tidigare kurser. Ljuddesign innebar ljudeffekter och dessa redigerades i Audacity [22]. Speldesign-området innehöll arbete med teman av olika miljöer och power-ups, vilket betydde att fysiken kunde manipuleras beroende på temat, exempelvis att friktionen ändrades.

4.4 Klasshierarki

Systemarkitekturen var inte komplicerad då spelet endast bestod av några få komponenter. Det fanns huvudsakligen två spelare som kontrollerar en varsin klubba, allt skedde lokalt och ingen data skickades via något nätverk. En översikt på spelets arkitektur illustreras i figur 4.2.



Figur 4.2: En översikt på spelets arkitektur.

Då spelet utvecklades i Unity strukturerades kod efter hur spelmotorn ville ha den. I Unity förekom en scen som innehöll olika GameObjects som till exempel ljuskällor, primitiv geometri, 3D-modeller och användargränssnitt. För att kunna manipulera och modifiera GameObjects i spelet skrevs egen kod i så kallade scripts som kunde exekverades vid start, varje frame eller i en funktion som kördes då vissa krav uppfylldes som till exempel vid ett knapptryck eller en kollision.

Scripts placerades på de GameObjects som de skulle påverka, detta så att de enkelt kunde komma åt dess parametrar och variabler. Scripts kunde även referera till andra scripts och GameObjects för att kunna komma åt deras variabler och funktioner. För mer generella script som hade med spelet som helhet att göra, och inte individuella GameObjects, lades de på ett Gameobject som inte hade någon synbar representation utan vars enda syfte var något att fästa scripten på i scenen. Programmeringen gjordes alltså med modulär tillvägagångssätt. En översikt på hur spelets hierarki byggdes upp av GameObjects illustreras i figur 4.3.



Figur 4.3: Översikt på spelets hierarki.

Med Unitys inbyggda filstruktur sparades alla filer i en mappstruktur som gjorde det enkelt att hitta och navigera i olika filer. Varje scen hade även i sin tur undermappar med relevanta material, texturer och dyligt som användes för att avskilja olika teman.

4.5 Spelfysik

Spelmotorn Unity har en inbyggd fysikmotor som låter objekt i scenen interagera med varandra på ett fysiskt korrekt sätt med avseende på kollision, krafter, hastighet och gravitation. Genom att ge objekten i scenen passande parametrar för massa, friktion, storlek och gravitationskonstant betedde de sig på ett verklighetstroget sätt.

4.5.1 Rörelse

För att kunna få så bra respons som möjligt förflyttades klubban genom att kontinuerligt uppdatera klubbans position att vara där användaren rör skärmen med fingret. Detta uppnåddes genom att först detektera om spelaren nuddar klubban med fingret, och sedan flytta klubban till fingrets position.

Genom att kontinuerligt ändra klubbans position till fingrets position kunde maximal precision uppnås. Det visade sig dock att hårdvaran inte uppdaterade fingrets position tillräckligt ofta, vilket skapade en födröjning mellan finger och klubba som låg utanför räckhåll för vad som kunde åstadkommas med mjukvaran.

Några andra metoder som testades men inte användes var att ge klubban en hastighet, acceleration eller kraft mot fingrets position efter att klubban blivit pekad på. Detta gav dock en oönskad födröjning eller skapade förödande överskjutningar om hastigheten, accelerationen eller kraften blev för låg respektive för hög.

För att optimera puckens rörelse sättes puckens materialfriktion mot omgivningen till noll, detta för att den skulle glida längre och se ut som att planen var friktionslös som på ett verkligt Air-hockeybord.

4.5.2 Kollision

Unitys fysikmotor kan hantera kollision men för den implementering som användes för klubbornas rörelse krävdes ytterligare funktionalitet. Eftersom positionen ändrades direkt i förflyttningen av klubborna hade de aldrig någon hastighet utan snarare teleporterades. Detta innebar att en egen kollisionshantering var tvungen att implementeras då Unitys inbyggda fysikmotor beror på att objekt har hastigheter vilket klubborna i teorin inte hade.

För att erhålla kollision måste därför ett test genomföras innan varje förflyttning av klubban för att se om klubban är på väg att kollidera. Detta genomfördes genom att använda *Ray casting* mellan klubban och fingrets position, om något påträffades på vägen stannade klubban där istället.

Ray casting [23] innebär att en linje beräknas ett visst avstånd i en riktning från en punkt, i detta fall från klubban till fingrets position, och ger till känna huruvida linjen korsar något på vägen. Om linjen bryts av ett objekt i scenen kan information fås om vilket objekt det var och var den är så att ett lämpligt beslut kan tas rörande förflyttningen av klubban. Om det är pucken som klubban kolliderar med ska en kraft överföras och om det är en power-up ska klubban gå rakt igenom, annars ska klubban stanna vid objekt så som väggar eller andra avgränsningar.

När klubban träffar pucken beräknas en teoretisk hastighet och acceleration som kan användas för att beräkna en kraft som ska föras över på pucken i tillslagsögonblicket. Eftersom klubbans och fingrets positioner var kända kunde deras avstånd ifrån varandra beräknas som $\Delta Position$ och genom att sedan dividera detta med tiden för hur lång tid en frame tar att beräkna, ΔTid , fås hastigheten v , se ekvation (4.1). Om hastigheten sedan divideras med ΔTid igen fås accelerationen a , se ekvation (4.2), med vilken kraften F , som ska överföras till pucken, kan beräknas enligt Newtons andra lag [24] genom att multiplicera den med klubbans massa m , se ekvation (4.3).

$$v = \frac{\Delta Position}{\Delta Tid} \quad (4.1)$$

$$a = \frac{v}{\Delta Tid} \quad (4.2)$$

$$F = m \cdot a \quad (4.3)$$

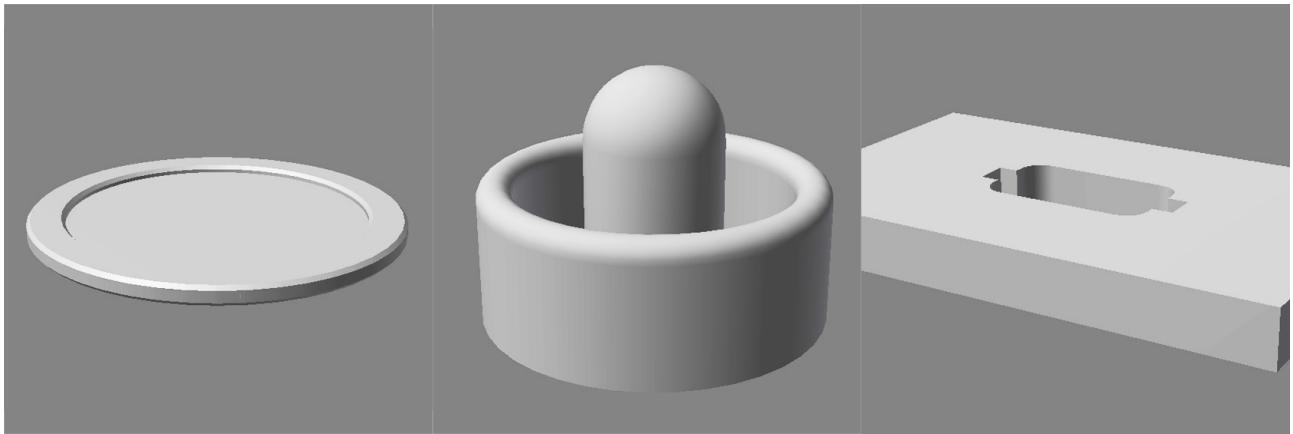
Eftersom pucken fick en hastighet av tillslaget med klubban kunde Unitys inbyggda kollisionshantering användas för att den skulle kollidera med omgivningen. Parametrarna i puckens material anpassades så att pucken skulle studsa mot omgivningen på ett tillfredsställande sätt.

Det märktes sedan att pucken ibland hade svårt att kollidera med klubban om de var på väg till samma plats samtidigt. Detta berodde på att i beräkningsteget såg det ut som att ytan var tom, för både klubbans och puckens kollisionsalgoritmer. Detta kunde leda till att pucken for rakt igenom klubban vilket var en oönskad händelse ur ett speloperspektiv. För att lösa detta kompletterades pucken med en funktion som, då pucken skulle finna sig inuti klubban, gav pucken en hastighet ut ur klubban samma väg som den kommit från, detta för att det skulle verka som att en kollision inträffat.

4.6 Grafik

Ett av kundkraven var att spelet skulle baseras på 2,5D-grafik. För att uppnå denna effekt av 3D-grafik på en tvådimensionell yta modellerades spelobjekten som 3D-modeller, se figur 4.4. Med hjälp av kamerans och modellernas position kunde spelet sedan få sitt djup och 2,5D-grafik.

Material till 3D-modellerna skapades i Unitys material API (Application Programming Interface) med hjälp av de texturer som skapades i Adobe Photoshop och importerades till spelmotorn. Dessutom importerades några av Unitys egna fysikaliska material och användes till speciella objekt.



Figur 4.4: De 3D-modeller som användes till pucken, klubban och spelplanen.

Ljus var en stor del av grafiken och det låg mycket tyngd i arbetet att skapa rätt ljus som därefter skapade skuggor. Spelets ljus beräknades i realtid och uppdaterades kontinuerligt vid varje frame. Detta innebar att vid förflyttning av ljusobjekt eller GameObjects under spelets gång uppdaterades ljuset samtidigt. Detta var fördelaktigt då spelet innehåller mycket rörelse. Rätt ljussättning och skuggor var viktigt för att behålla 3D-perspektivet i en 2D-miljö.

Ambient ljus tillsammans med belysning via *spotlights* användes till att skapa en enhetlig scen med en betydlig ljusstyrka. Ambient ljus fungerar som en global ljuskälla där varje objekt i scenen blir upplyst från alla riktningar [25]. Användningen av ambient ljus var en enkel lösning för att höja ljusstyrkan hos en scen utan att behöva ändra värdena hos ljusobjekten individuellt.

Till skillnad från ambient ljus gav ljuset från en *spotlight* en direkt ljuskälla med en riktning. Spotlights riktar sig mot en specifik punkt och lyser upp denna med en smal och intensiv ljusstråle [26]. De används ofta till att framhäva något, till exempel en spelare på en fotbollsplan eller en artist på en scen. I spelet användes detta för att ge mer fokus åt spelplanen, liknande en sportarena med mycket ljus riktad mot spelplanen.

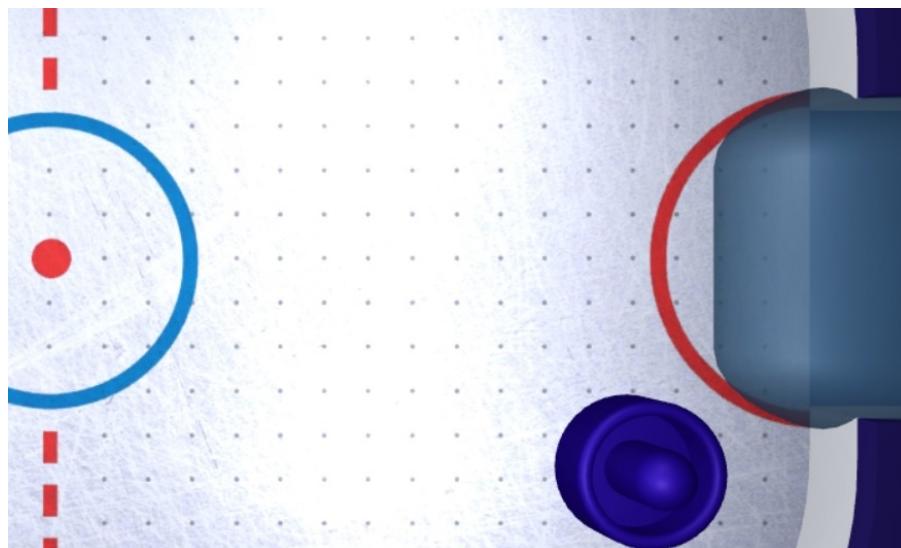
4.7 Ljud

Ljud spelar en viktig roll i spel då det ger en återkoppling vilket leder till att interaktioner känns mer realistiska. Detta är en viktig aspekt i projektet då känslan av ett verkligt Air-hockeybord ska simuleras. Ljud för allt som hörs i ett fysiskt Air-hockey spel, såsom när klubban slår till pucken och när en spelare gör mål förekommer i spelet. En implementation som ökade realismen var en dynamisk volymkontroll. Detta gjorde att volymen ändrades beroende på hur kraftigt klubban träffade pucken.

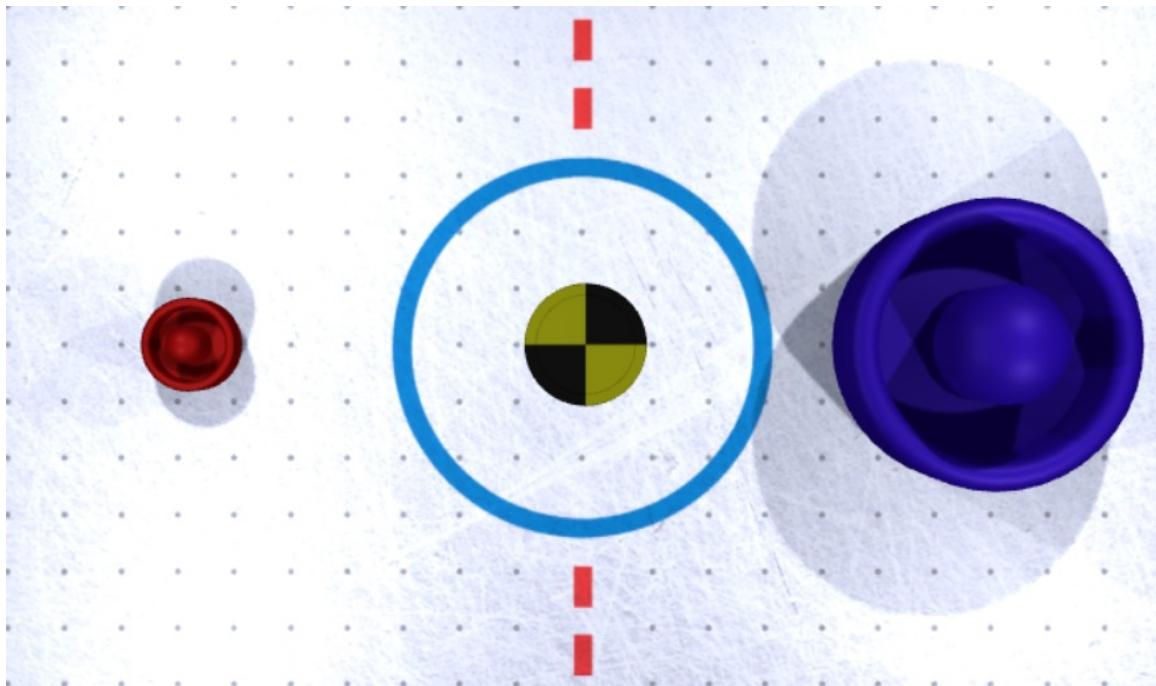
4.8 Power-ups

I spelet dyker det upp slumpmässiga power-ups som förstärker spelupplevelsen. I spelet finns det tre olika power-ups som påverkar spelaren på olika sätt. De effekter som dessa ger illustreras i figur 4.5 och 4.6, samt beskrivs nedan:

- **Sköld-power-up** aktiverar en sköld vid målet för den spelare som tog upp effekten. Denna sköld kan ses som ett extra liv och går sönder när den kolliderar med pucken en gång. Vilket kan ses i figur 4.5.
- **Minska-power-up** krymper spelarens klubba och varar i fem sekunder. Detta kan ses i figur 4.6. Att minska klubban är en nackdel för spelaren då det ger mindre area för spelaren att använda för att skydda sitt eget mål.
- **Förstora-power-up** förstorar spelarens klubba och varar i fem sekunder. Tvärtemot minska-power-up är detta en positiv effekt som gör att spelaren lättare kan blockera målet från pucken. Effekten kan ses i figur 4.6.



Figur 4.5: Skölden dyker upp vid spelarens mål när denne tar upp en sköld-power-up. I detta fall har den blå spelaren plockat upp en power-up som skapar en sköld som skyddar den blå spelarens mål från den röda spelaren.



Figur 4.6: En power-up som minskar klubban är aktiv på den röda spelaren och en power-up som förstorar klubban är aktiv på den blå spelaren.

Dessa power-ups dyker upp på tolv förutbestämda platser på spelplanen och tas upp av spelare via pucken, alltså inte via spelarens klubba. Den spelare som senast rörde pucken när pucken tar upp en power-up får effekten som denna ger. Pucken känner av vilken spelare som senast hade kontakt vid den och byter färg för att förtydliga detta för spelaren. Power-ups skapas efter en slumpmässig tid i intervallet fem till tjugo sekunder.

Om en spelare redan har en power-up aktiv och sedan tar upp en power-up av samma slag förlängs tiden på effekten med fem sekunder eller så ökar antalet liv på skölden med en enhet. Detta gör det omöjligt för flera power-ups att bygga på varandra och förhindrar att en spelare kan bli väldigt liten eller stor om denne tar upp många minska- eller förstora-power-ups på kort tid.

Designen av power-up-objekten hölls enkel och minimalistisk, och modellerades i Blender som iconer. Ikonerna fick formen av en sköld, ett minustecken och ett plustecken, utefter de effekter de har. Minustecknet får alltså klubban att krympa, plustecknet får klubban att expandera och skölden skapar en sköld över målet. Alla power-ups kan ses i figur 4.7.



Figur 4.7: Den slutgiltiga designen på de tre power-up objekten. Den vänstra är en förstora-power-up, mittersta är en sköld-power-up och den högra är en minska-power-up.

4.9 Optimering

Optimering var en viktig aspekt i detta spelprojekt. Då responsiviteten skall motsvara ett verkligt Air-hockey spel krävs det så få steg som möjligt mellan touchinput och återkoppling på skärmen. Hjälpmedel så som profileringsverktyget [17] i Unity användes för att identifiera möjliga flaskhalsar i prestandan för att optimera spelet.

Ett vanligt alternativ för användaren i datorspel är att ändra grafikinställningar. Detta gör att grafiken kan skräddarsys med hänsyn av vad den egen hårdvara klarar av. I detta projektets fall skulle spelet spelas på en utställning på samma enhet. Detta förenklade optimeringen då prestandan endast behövde testas på en enhet.

För att kunna få bättre prestanda och mer responsiva rörelser stängdes *V-sync* (Vertical sync) av. *V-sync* har som uppgift att synkronisera grafiken med uppdateringsfrekvensen på skärmen för att förhindra artefakter. Detta begränsar dock uppdateringsfrekvensen till 60 FPS vilket leder till att responsiviteten för spelet minskar då klubbans position räknas om varje frame. Vid högre FPS kan klubban istället flyttas oftare och bättre hänga med fingrets rörelser.

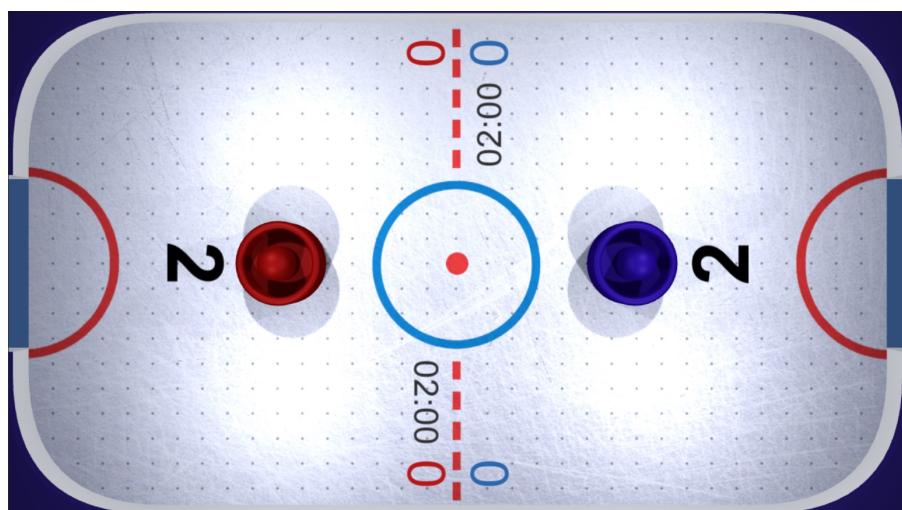
Kapitel 5

Användarupplevelse

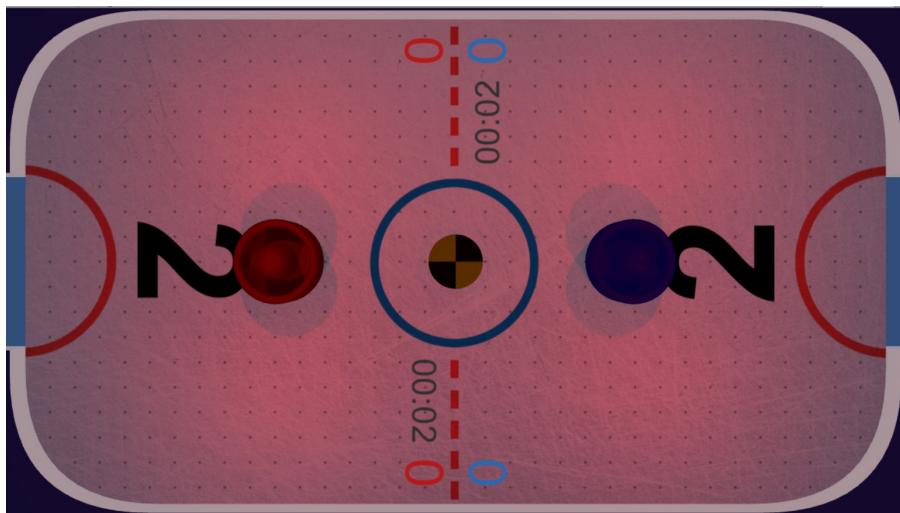
Det är viktigt att användaren förstår vad den ska göra för att komma vidare i ett spel och det ska vara tydligt vad som är möjligt och inte möjligt att göra. Detta avsnitt handlar om hur spelets design utvecklades med hänsyn till användarupplevelsen.

5.1 Användargränssnitt och menyer

Kundens krav gällande användargränssnittet var att spelet skulle vara lätt att förstå och vem som helst skulle kunna gå fram till bordet och spela utan tidigare erfarenhet. För att uppnå detta har projektgruppen bland annat fokuserat på återkoppling i form av poäng, tidtagare, nedräkning vid nedsläpp och nedräkning när tiden närmar sig slusignalen, se figur 5.1 och 5.2. Speldesignen liknar det klassiska Air-hockeyspelet som är inspirerat av sporten ishockey. Utseendet av planen tillsammans med återkopplingen ska leda till att användaren associerar dessa och därmed förstår vad spelet går ut på, det vill säga att spelarna ska använda sina klubbor för att slå in pucken i motståndarens mål.



Figur 5.1: Spelplanen när spelet startar. En timer räknar ner i tre sekunder innan pucken släpps. I bilden syns att det är två sekunder kvar innan spelet startar.



Figur 5.2: En timer räknar ner de sista fem sekunderna. I bilden syns att det endast är två sekunder kvar innan spelet avslutas.

För att underlätta förståelse och användning av spelet ytterligare, bestämdes det även att användaren inte skulle ha några valmöjligheter, till exempel kan inte användaren välja tema själv utan temat avgörs slumpmässigt.

En annan aspekt som behandlades var knapparna. Texten på knapparna valdes att vara kort men tydlig, en faktor som skulle göra gränssnittet mindre rörigt. Något som var svårare att åstadkomma med touch var den visuella återkopplingen som vanligtvis visas när muspekaren hålls över ett objekt för att visa att det är ett klickbart objekt. Detta löstes genom att knappen byter färg.

För att spelet skulle passa utställningen tidsmässigt valde projektgruppen att både ha en målgräns och en tidsgräns när spelet ska avslutas. Tidsgränsen ger ett tydligt slut och förhindrar att samma par användare fortsätter spela under en längre period. Användarna kan även lämna spelet när som helst och det blir lätt för nästa användare att starta spelet. Om ingen användare har integrerat med spelet på tjugo sekunder återgår spelet till en startmeny, se figur 5.3, och är då inbjudande för nästa par att börja spela från början.

Användargränssnittet anpassades efter de två spelarna på vardera kortsida. Detta innebär att mycket av återkopplingen är riktad åt båda spelarnas håll, exempel på detta ges i avsnitt 5.1.2. Spelet har även en utvecklarpanel som kan aktiveras med ett tangenttryck för att visa uppdateringshastigheten och antal touchinput. Syftet med detta är att ge prestandaåterkoppling till utvecklaren eller utställaren för felsökning.

5.1.1 Startmeny

I figur 5.3 visas spelets startmey. Startmenyn visas vid första uppstarten av spelet och efter en viss tids inaktivitet.

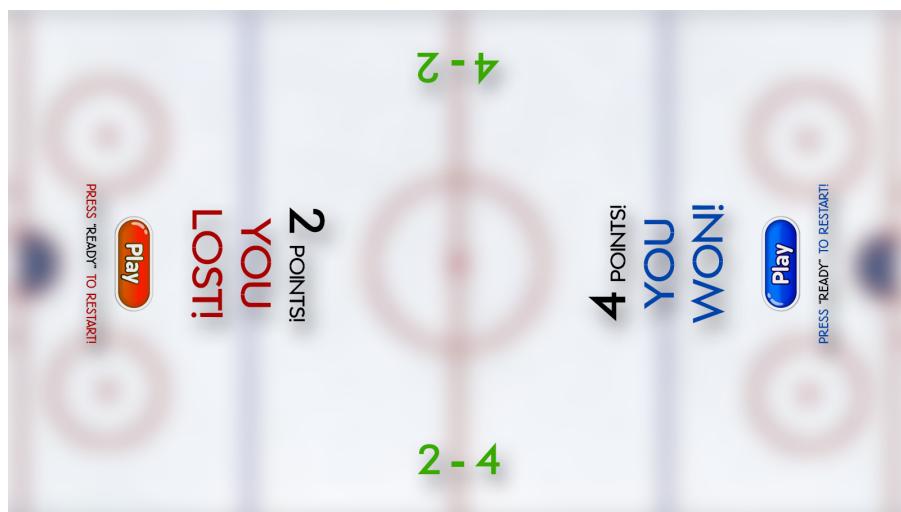


Figur 5.3: Startmeny.

Designen av startmenyn är minimalistisk. Anledningen till detta är att öka synligheten och minska risken för förvirring. Riktningen mot längssidan ska även öka synligheten ännu mer tillsammans med den stora texten och kontrasterna i färg.

5.1.2 Omstartsmeny

Omstartsmenyn är likt spelet anpassad att vara riktad till båda spelarna. Poäng, text om vem som vann och omstartsknappar finns på båda kortsidor för att ge återkoppling om matchen till spelarna, se figur 5.4.



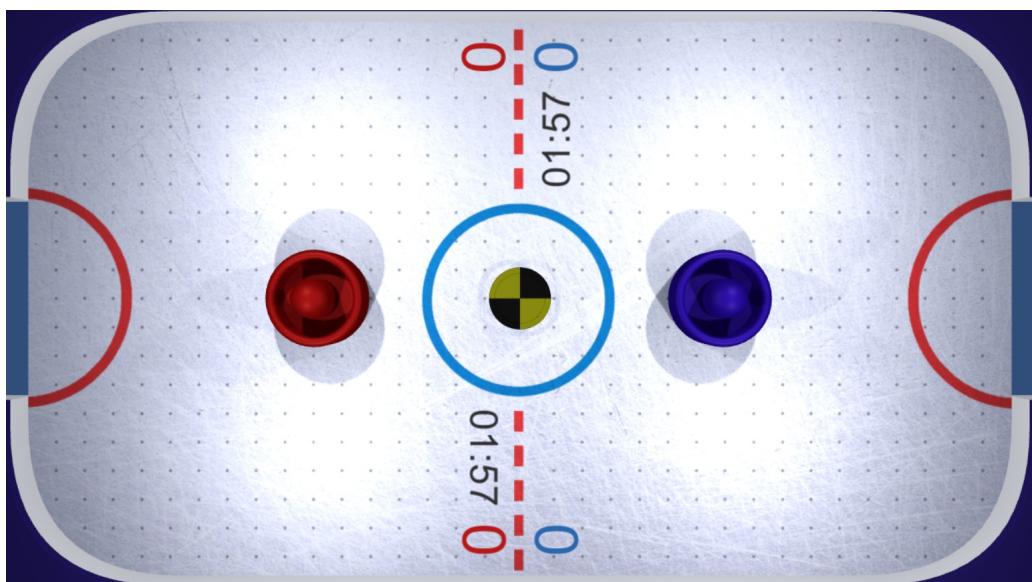
Figur 5.4: Tidig skiss på omstartsmenyn.

För omstart krävs att båda spelarna är redo. Omstartsknapparna är placerade så att båda spelarna enkelt ska kunna nå dem. Återkoppling av spelarstatus ges vid omstartsknappen i form av text: ”Väntar på motståndare” eller ”Ny omgång? Markera dig som redo”. Vilken text som visas beror på om spelaren markerat sig som redo eller inte. Här reflekteras även tillståndet i färgen på knappen, den röda knappen blir grön när spelaren är redo. Den röda och gröna färgkodning som valts för knapparna är då människan ofta förknippar grönt med att något är tillåtet eller säkert och rött med det motsatta. Detta på grund av flera olika faktorer som trafikljus och andra sociala konstruktioner [27].

5.2 Teman

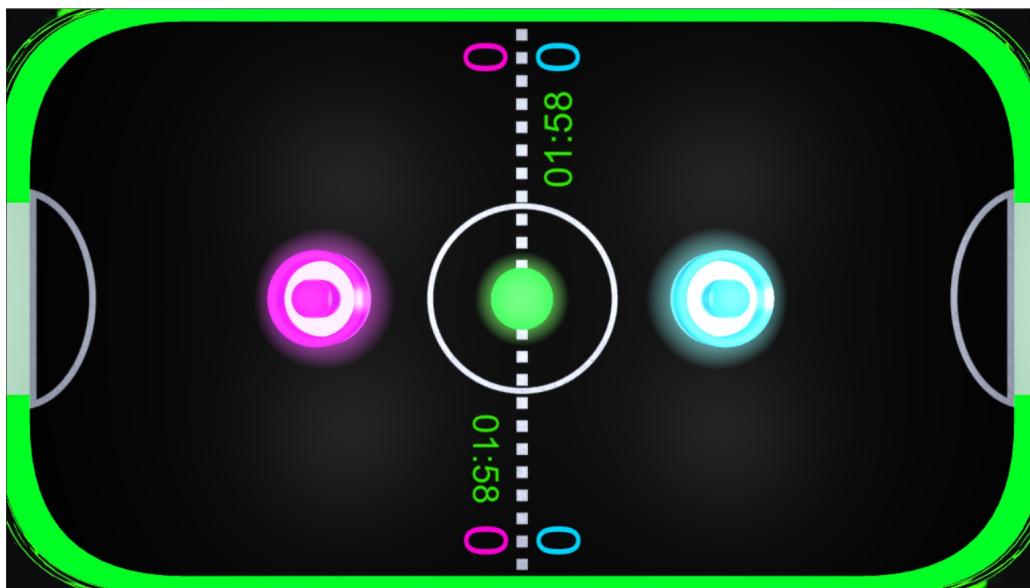
Ett av de krav som kunden hade på spelet redan från början var att det skulle finnas minst två olika teman. Projektgruppen skapade totalt fyra stycken olika teman, dessa var klassisk, neon, fotboll och godis.

Klassisk - En känsla av verklig hockey och Air-hockey med förenklad ishallsdesign och klassiskt färgtema. Inspiration från det klassiska Air-hockeyspelet från 1969 [1]. Bild på denna design kan ses i figur 5.5.



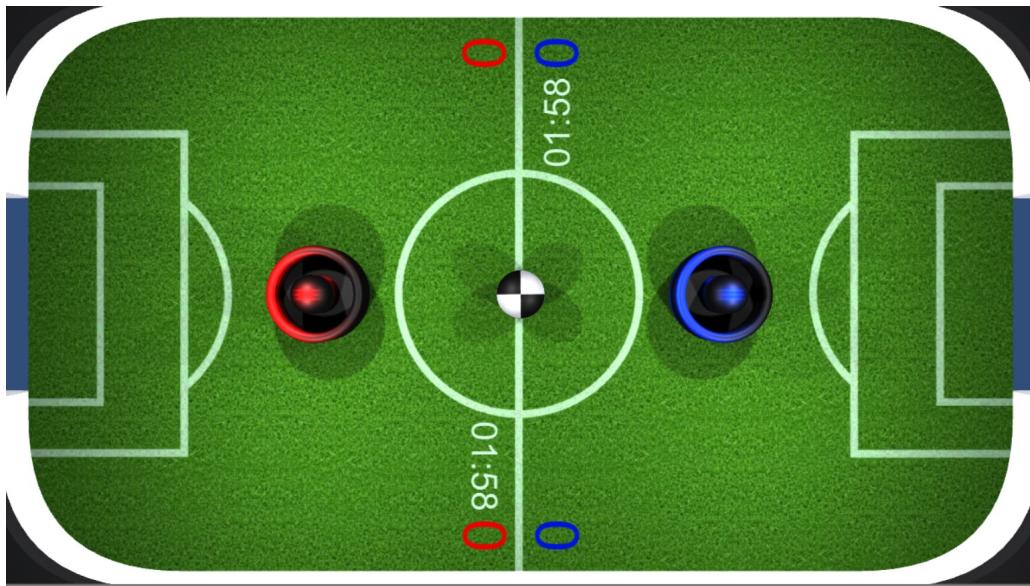
Figur 5.5: Det klassiska temat. Inspirerad av det klassiska spelet.

Neon - Ett vanligt förekommande tema i de digitala Air-hockey versionerna. Det som skiljer sig mot det klassiska temat är främst de självlysnande objekten och möjligheten att experimentera med tema-specifika power-ups, ljud, musik och effekter. Bild på denna design kan ses i figur 5.6.



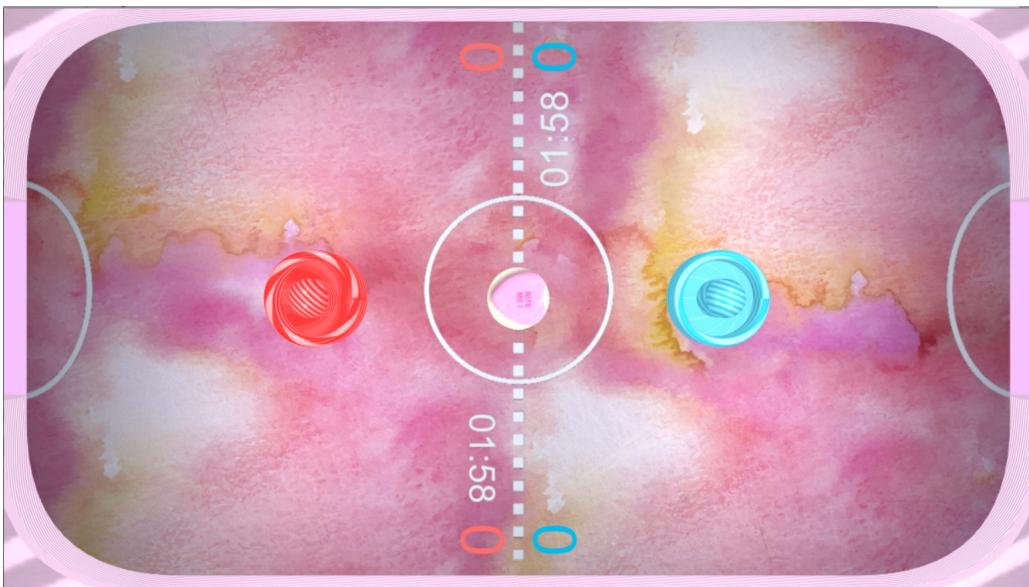
Figur 5.6: Neontemat. Klubbarna och pucken är självlysnande.

Fotboll - I detta tema byttes den platta pucken ut mot en boll som rullade eller gled genom spelplanen. Tanken med detta tema var att skapa en fotbollskänsla med hjälp av annorlunda ljussättning och fysik. En annan idé var att friktionen skulle skilja sig från den i det klassiska temat och simulera gräset på fotbollsplanen istället för isen i hockeyrinken. Bild på denna design kan ses i figur 5.7.



Figur 5.7: Fotbollstemat.

Godis - Detta temat är inspirerat av pastellfärgar och skulle skapa en känsla av ett fantasiland. I detta temat finns det en annorlunda tolkning av klubborna. Dessa ser ut som en godisklubba istället för en Air-hockeyklubba som i det klassiska temat. Bild på denna design kan ses i figur 5.8.



Figur 5.8: Godistemat med mer pastellfärgar.

Det fanns ideér till fler teman, dock fanns det inte tid att implementera dessa. Kravet från kunden var att det skulle finnas två fungerande teman.

5.3 Resultat av användartester

Under användartesterna ställdes frågor i områdena responsivitet, grafik, gränssnitt, power-ups och allmänna förbättringar. De specifika frågorna återfinns i bilaga C. Utifrån frågorna sammanfattades svaren i olika aspekter av spelet.

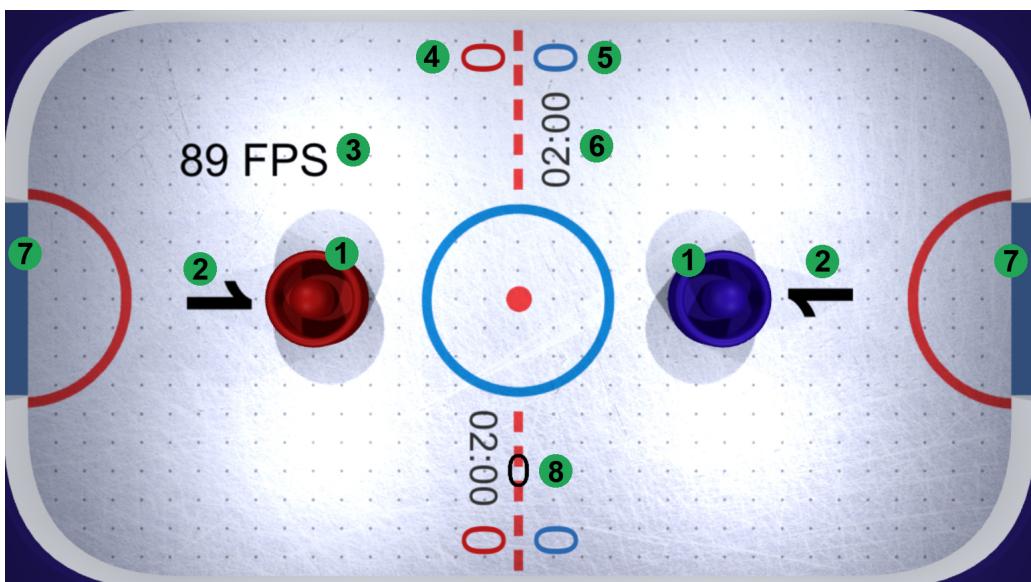
Användarnas första intryck av spelet var att det var lätt att förstå, men de upplevde vissa problem inom responsivitet och användargränssnitt. Majoriteten av användarna ansåg att positionsversionen var bättre än hastighetsversionen när det kom till responsivitet, men ingen av dem upplevdes som riktigt bra. Användargränssnittet fick generellt bra återkoppling men utformningen av spelets power-ups ansågs vara otydlig och behövde förbättras. Resten av användarnas åsikter återfinns i bilaga D.

Ett frågeformulär för att besvara de mer specifika frågorna skapades. Enligt svarsformuläret framgick det att en match skulle vara i maximalt två minuter och avslutas antingen när tiden var slut eller när en spelare gjort fem mål, och ett mål skulle inte räknas förrän pucken fallit ner i målets hål. Användarna ansåg även att de inte ska kunna passera mittlinjen med sina klubbor. Frågeformuläret och de sammanställda resultaten återfinns i bilaga E.

Kapitel 6

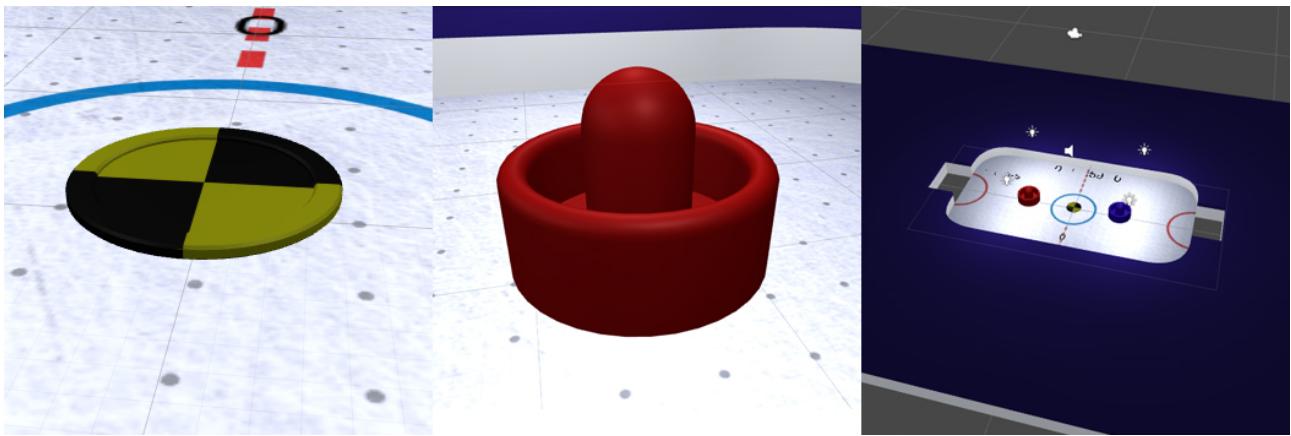
Resultat

I detta avsnitt presenteras slutprodukten och i kapitel 7 jämförs resultatet med kundens krav. En översikt över spelplanen och de komponenter som den består av, inklusive förklaring, visas i figur 6.1.



Figur 6.1: Översikt över spelplanen och dess komponenter. 1. Klubbor som spelarna kan kontrollera. 2. Nedräkning till när pucken släpps. 3. FPS-räknare. 4. Poäng för röd spelare. 5. Poäng för blå spelare. 6. Återstående tid tills spelet avslutas. 7. Mål. 8. Antal touchinputs.

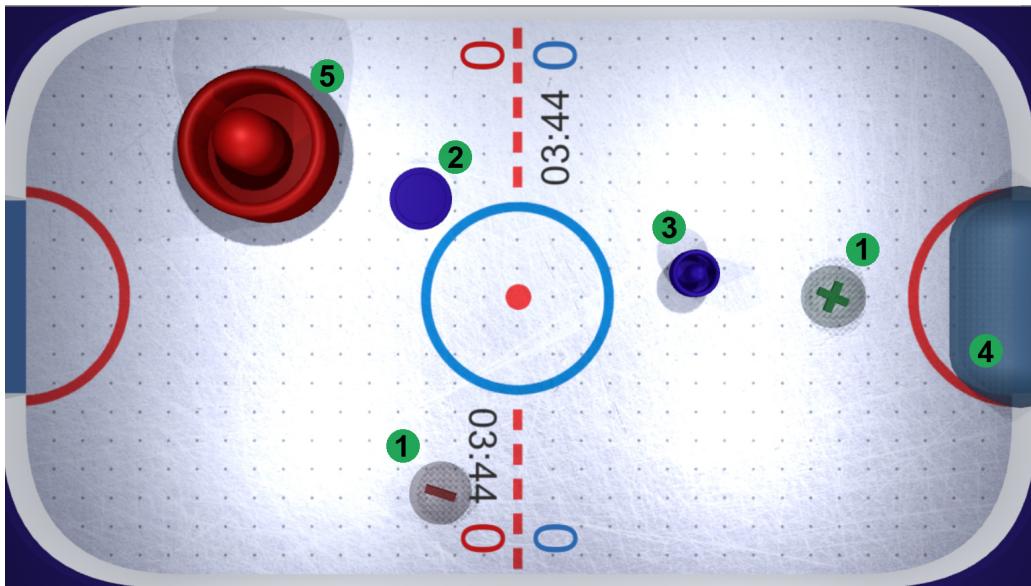
Spelet startar med en timer på tre sekunder som räknar ner för att spelarna ska vara beredda. Båda spelarna kan sedan placera ett finger på varsin klubba för att kontrollera den. Klubban, pucken och spelplanen visas närmare i figur 6.2. Spelet avslutas när en av spelarna har gjort ett visst antal mål eller när tiden löpt ut. Den spelare som vinner är spelaren med flest antal poäng. Om båda spelarna har samma antal poäng när tiden går ut blir spelet oavgjort. När spelet är slut dyker en omstartsmeny upp med matchstatistik och spelarna får en chans att starta om spelet.



Figur 6.2: Pucken, klubban och spelplanen med texturer applicerade i det klassiska temat. Pucken byter sedan färg till rött eller blått beroende på vilken spelare som sist slog till den.

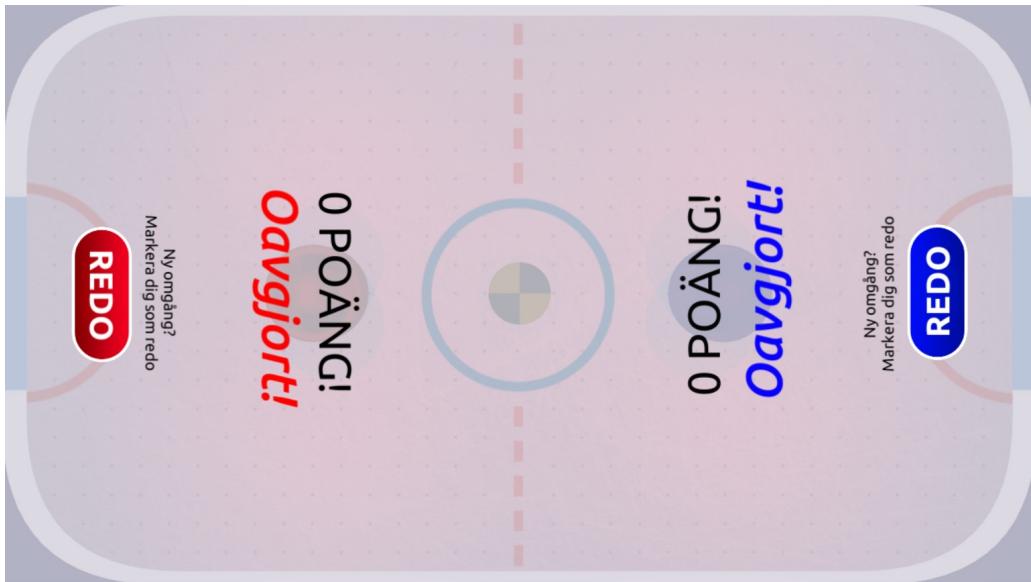
I figur 6.1 visas en FPS-räknare och en siffra för antalet touchinputs. Dessa var till för projektgruppen vid utvecklingen av produkten och lades in i ett utvecklarfönster i Unity. Detta fönster kunde sedan stängas av och på med ett knapptryck på ”d”-knappen på tangentbordet. Anledningen till detta var att spelarna inte skulle kunna se funktionen men att utställningspersonal ändå skulle få tillgång till den ifall något skulle krångla med spelet på utställningen.

Under spelets gång dyker det slumpmässigt upp power-ups. Dessa tas upp via pucken och inte via spelarens klubba. När en spelare slår på pucken och pucken träffar en power-up aktiveras effekten för den spelare som senast nuddade pucken. Vilka power-ups som finns och vilka effekter de har förklaras mer ingående i avsnitt 4.8. Ett exempel på hur spelet kan se ut efter några power-up aktiveringar kan ses i figur 6.3.



Figur 6.3: Exempel på några power-ups under en spelomgång. 1. Power-up objekt. 2. Puck träffad av blå spelare och har bytt färg. 3. Aktiv minska-power-up effekt på den blå spelaren. 4. Sköld som skyddar blå spelarens mål. 5. Aktiv förstora-power-up effekt på den röda spelaren.

När spelet avslutas dyker omstartsmenyn upp där spelarna kan se poängresultatet, vilken spelare som vann eller om det blev oavgjort. Spelet kan startas om genom att båda spelarna klickar på "REDO"-knappen som finns på båda sidorna av spelplanen. Omstartsmenyn förklaras mer ingående i avsnitt 5.1.2 och kan ses i figur 6.4. När spelet startas om ändras spelplanen till ett annat tema. Slutprodukten har totalt fyra olika grafiska teman som byts efter varje spelrunda. I avsnitt 5.2 visas dessa teman mer ingående.



Figur 6.4: Omstartsmeny efter ett oavgjort spel.

I den senaste versionen av spelet förekommer en del buggar, till exempel kan pucken i vissa fall åka igenom klubborna och en spelare kan fastna på den andra spelarens planhalva. Detta beror på den egna kollisionsdetekteringen och förklaras mer ingående i avsnitt 4.5.2 och diskuteras i avsnitt 7.2.

Kapitel 7

Analys och diskussion

Detta kapitel analyserar och diskuterar arbetet som utförts. Delarna som är i fokus här är arbetets metod, resultat och arbetet i ett vidare sammanhang.

7.1 Metod

Det faktum att GitHub användes i kombination med Unity fungerade inte helt felfritt, trots att guiden för att sammanfoga Unity och GitHub [15] användes. GitHub registrerade fortfarande metafilerna som ändringar i projektet och när dessa pushades upp skapades problem. Det förekom konflikter flera gånger mellan olika parter som redigerade i samma scen vilket ledde till att versionen på GitHub slutade fungera. Detta lösades genom att en äldre version som var sparad lokalt på en gruppmedlems dator ersatte den version som fanns i GitHub. Dock innebar detta att många ändringar implementerades flera gånger på grund av konflikterna i GitHub. Trots att källor beskrev att det var en av de bättre lösningarna som fanns när flera personer jobbade i Unity samtidigt [15] fungerade det inte optimalt. Problemet lösades aldrig under projektets gång och projektgruppen accepterade att flera inte kunde redigera samma scen samtidigt. Gruppmedlemmarna kunde dock redigera och skapa separata script eller filer som kunde pushas upp separat till GitHub samtidigt som någon annan jobbade i scenen.

Scrum som utvecklingsmetod har varit fördelaktigt då projektgruppen lättare kunnat anpassa produkten till kundens kravändringar och givit gruppen bättre förståelse för att hantera prioriteringar på uppgifter. Även om gruppen tycks förstå teorin av utvecklingsmetoden och försökt jobba enligt den, har det emellanåt känts ineffektivt i praktiken. Eftersom det var första gången gruppen arbetat med Scrum var det svårt att uppskatta hur mycket tid som krävdes för de specifika uppgifterna. En annan nackdel som uppstod med att utveckla agilt var att det var svårt att förutspå hur kundkraven kunde komma och förändras med tiden. Detta gjorde det svårt att bedöma vissa sprints arbetsdetaljer innan varje sprint startade. Därför var det smidigt att ha en översiktig tidsplan, i detta fall Gantt-schemat som visas i figur 3.1. En annan svårighet var att arbeta vidare i vissa perioder på grund av storleken av projektgruppen, sju personer kunde känna överflödigt och det kunde vara svårt att delegera uppgifter till alla gruppmedlemmar. Det har varit fördelaktigt att arbetet skett individuellt men att gruppen suttit tillsammans. Det individuella arbetet gjorde det lättare att fokusera på den enskilda uppgiften, vilket gjorde att gruppen blev mer produktiv, och fördelen med att sitta tillsammans var att gruppmedlemmarna alltid var tillgängliga, för exempelvis kodgranskning och övrig feedback. Men en nackdel som förekommit med arbetssättet var att det kunde bli tidskrävande då gruppmedlemmarna ibland hade det svårt att sätta sig in i varandras arbeten.

7.2 Resultat

Kundens krav uppfylls rent prestandamässigt men det finns fortfarande utrymme för förbättring. Det största problemet som uppkom var touchinput. Projektgruppen hade tillgång till två olika touchbord och dessa hade olika responstider på touchinput. Hos den ena fanns en relativt lång väntetid hos skärmen vid tolkning av fingrets position till en position på skärmen. Något som bidrog till en längsammare touchinput även fast spelet fungerade optimalt. Detta kunde inte projektgruppen påverka då det berodde mer på hårdvaran. Dock kunde projektgruppen påverka det avstånd som uppstod mellan den touchinput som skärmen fick och positionen på klubban i spelet.

7.2.1 Positionsbaserad styrning

Att minimera avståndet mellan de touchinput som skärmen fick och positionen på klubban, var en utmaning under hela projekttiden. Den sluttgiltiga lösningen blev att klubban var positionsbaserad, vilket innebar att klubban teleporterades till den position fingret hade vid varje frame. Något som gav en bättre responsivitet mellan klubban och fingret. Dock slutade den inbyggda kollisionshanteringen i Unity att fungera. På grund av detta blev det nödvändigt att implementera ett eget kollisionshanteringssystem. Det nya kollisionshanteringssystemet var fortfarande inte helt felfri, det uppstod buggar som till exempel gjorde att pucken kunde åka igenom klubban i vissa fall och att klubban kunde gå över till andra spelarens planhalva. Det blev även svårt att beräkna hur klubban påverkade andra objekt som den kolliderade med. När klubban kolliderar med pucken ska pucken skjutas iväg, men då klubban teleporterades vid varje frame uppstod aldrig någon stöt i ren fysikalisk mening. En kraft beräknades istället som kunde approximera hur pucken skulle påverkas vid kollision.

Den tidigare kollisionsversionen var hastighetsbaserad istället för positionsbaserad. En hastighet som var riktad mot fingrets position lades på klubban och gjorde att den alltid strävade efter att vara där fingret var placerat. Detta fungerade bra i små rörelser men vid större rörelser blev klubban mindre responsiv i jämförelse med den positionsbaserade versionen. I denna version fungerade även Unitys egna kollisionshantering och många av de problem som dök upp i positionsversionen försvann.

Vilken version som är bäst på att hantera klubbens rörelse är svårt att avgöra. Positionsversionen gav bättre responsivitet men skapade buggar som kan irritera användaren och i värsta fall förstöra spelupplevelsen. Hastighetsversionen hade inte dessa problem men är responsiv i lägre grad när det gäller att flytta klubban till fingrets position. Under användartester fick användarna testa de båda versionerna och resultatet visade att de flesta användare tyckte bättre om den positionsbaserade versionen. En sammanställning av användartesterna finns i avsnitt 5.3 där alla frågor som ställdes finns i bilaga C. Projektgruppen beslutade då att välja den positionsbaserade versionen för att hantera klubbornas rörelse.

7.2.2 Power-ups

Ett annat förbättringsområde är spelets power-ups. När en power-up dyker upp i spelet finns det en möjlighet att den dyker upp på en plats där en power-up redan finns. Det hade varit bättre att ha maximalt en power-up per position. Detta var svårt att implementera och tanken var att de inte skulle förekomma allt för regelbundet, utan tas upp av spelarna relativt fort. Alltså skulle positionerna bli tomma fortare och risken att de dyker upp på varandra minskar. Vid inaktivitet under spelet kan detta hända relativt fort. På grund av problem vid implementeringen och tidsbrist åtgärdades inte detta problem. Däremot åtgärdades att om en användare erhåller en power-up medan en annan redan är aktiv, ökas bara varaktigheten för den power-upen.

7.2.3 Teman

De teman som skapades prioriterades i ordningen klassisk, neon, fotboll och sist godis. De teman som jobbades mest på var alltså det klassiska temat och neontemat. Detta ledde till att minst tid lades ner på fotboll- och godistemat och dessa blev inte lika bra som de andra två teman som finns. Det fanns även mycket idéer hos projektgruppen som inte implementerades på grund av tidsbrist. Ett exempel på detta var temaspecifika power-ups och ljud, samt en unik friktion på underlaget i fotbollstemat. Dock var kundkravet endast två teman, vilket ledde till att projektgruppen valde att prioritera två som skulle representera spelet.

7.3 Arbetet i ett vidare sammanhang

Eftersom produkten är ett Air-hockey spel till ett elektroniskt touchbord bidrar den till all den digitala underhållning som tagit över världen sedan millennieskiftet. Unga människors ökade intresse för att sitta inne vid sina datorer istället för att vara ute i naturen och umgås med människor är en negativ trend i dagens samhälle som inte har några tydliga tecken på att avta. Med ett spel i utställningsmiljö får spelarna en verlig social interaktion vilket borde uppmuntras då det är väsentligt för människors välmående.

Spelet ska placeras på utställningar där besökare i alla åldrar med olika bakgrunder kan interagera med det. Alltså bör spelets utformning i form av ljud, teman och funktioner vara anpassade därför och försäkra att det inte förekomma några kontroversiella eller diskriminerande ämnen, exempelvis religiösa symboler eller politik. Spelet ska inte heller på något sätt vara diskriminerande mot funktionsnedsatta, olika etniciteter eller kön eftersom det kan orsaka obehag eller väcka anstöt som skulle kunna tänkas skrämma iväg besökare eller skapa dålig publicitet. Till skillnad från ett fysiskt Air-hockey är det inte diskriminerande mot individer som inte har möjligheten att greppa objekt men däremot begränsad mot individer som inte har möjlighet att använda funktionen touch. På grund av hälsoriskerna är universella teman något som eftersträvas, alltså inga snabba blinkande färger eller höga ljud, detta för att exempelvis inte utlösa epilepsi eller huvudvärk.

Spelet är på svenska men skulle kunna vidareutvecklas att innehålla möjlighet att ändra språk mellan svenska och engelska för att även icke svenska talande besökare ska känna sig delaktiga. I utställningen kan det förekomma besökare som har ett aggressivt beteende eller besökare som har avsikten att sabotera utställningsmaterial. Det är därför viktigt att programvaran har funktioner som förhindrar detta. I övrigt är det utanför mjukvarans kontroll att påverka hårdvarans livslängd och miljöpåverkan.

Arbetet skulle kunna släppas som öppen källkod, så även andra användare på nätet kan ladda ned och testa det, eventuellt utveckla vidare som fritidsintresse. Det är då viktigt att se till att inte någon använder det kommersiellt, eftersom det ska vara öppen källkod.

Övriga vidareutvecklingar skulle kunna vara att släppa spelet till mobila enheter, om detta skulle ske är det viktigt att guida användaren hur spelet fungerar och vid behov även ge en åldersgräns för spelet. Populärt inom dagens spelindustri är mikrotransaktioner, det skulle kunna innehålla att delar av systemet inte är tillgängligt för användaren och måste köpas med pengar eller virtuell valuta som existerar inuti spelet. Ett möjligt sammanhang där detta kan implementeras är till exempel olika teman som kostar extra att innehålla. En funktion som skulle kunna läggas till i systemet är delning av poängen i spelet via sociala medier. Detta skulle skapa ett ökat flöde i sociala medier för kunden och styrka kundens marknadsföring.

Om produkten däremot skulle säljas till den privata marknaden skulle den kunna bli en statussymbol eftersom utrustningen som krävs för att spela spelet inte är bland det som vanligtvis kan finnas i hemmet utan måste köpas för större kostnad. Detta skulle leda till större klyftor mellan samhällsklasserna

då de som har och de som inte har får än mer att ha som en olikhet sinsemellan.

Kapitel 8

Slutsatser

Syftet med projektet var att undersöka utvecklingsmöjligheterna för ett Air-hockey spel till touchbord. Utmaningen var att efterlikna det verkliga spelet samtidigt som konceptet skulle vidareutvecklas för att skapa en varierad spelupplevelse. Ett par frågeställningar ställdes upp som grund för att uppnå detta och svaren till frågeställningarna redovisas i detta kapitel. Förutom svaren visar även resultatet ett spel som till stor del liknar det verkliga spelet gällande spelfysik, interaktion och design. Spelet tappar lite verklighetskänsla på grund av responsiviteten men ger en varierad upplevelse tack vare olika teman samt tillägget av power-ups.

8.1 Svar på frågeställningarna

Nedan följer det gruppen kommit fram som svar på de inledande frågorna.

8.1.1 Är Unitys inbyggda kollisionshantering ett fullgott alternativ för att implementera beteendet hos en simulering av Air-hockey, för spelbarhet och realism, eller krävs externa alternativ och i så fall varför?

Implementationer som använde sig utav Unitys inbyggda kollision för klubban innebar en sämre responsivitet och användarupplevelse enligt användartesterna vilket presenteras i avsnitt 5.3. Unity krävde att objekt förflyttades med en hastighet för att kunna kollidera. Då ett riktigt Air-hockey spel ger användaren direkt styrning över klubban och makt över dess position, krävs det även av det digitala Air-hockeyspelet att kontrollen ska vara så bra som möjligt. Unitys inbyggda kollisionshantering satte krav på implementationen av hur en användare ska styra klubban i spelet, som stod i konflikt med kraven på en användares spelupplevelse och var därför inte att föredra inom ramarna för projektet. Detta innebar att Unitys kollisionshantering inte var ett fullgott alternativ för implementeringen av det digitala Air-hockeyspelet.

Andra lösningar var tvungna att utforskas och det som implementerades var en kombination utav Unitys inbyggda kollision och egna script skrivna i Visual Studio. För att hantera när klubban skulle kollidera gjordes beräkningar i scripten, men för puckens kollisioner kunde Unitys egna kollisionshantering användas.

8.1.2 Hur uppnås, genom användargränssnitt och med hjälp av design och återkoppling, att en användare intuitivt kan uppfatta hur den ska interagera med det digitala Air-hockeyspelet?

Spelets användargränssnitt är speciellt anpassat för utställningsmiljön och touchbordet. Detta innebär bland annat att spelet utvecklats på svenska för att vara lättförståeligt för alla besökare oavsett ålder, och att gränssnittet anpassats efter den stora skärmen.

Funktionaliteten förmedlas genom den design och återkoppling som finns i spelet, bland annat i form av poäng, tidtagare och nedräkning. Spelet, likt det verkliga, är designat att påminna om ishockey och ger därmed användaren en första aning om hur spelet fungerar. Genom utseendet av planen, placeringen av modellerna och med förkunskaper om ishockey kan användaren förstå att den ska ta tag i klubban och slå pucken i motståndares mål. Informationsåterkopplingen ger sedan användaren förståelse om hur matchen går: vem som leder och hur mycket tid som är kvar.

På omstartsmenyn förekommer knappar på vardera kortsida som är riktade mot respektive spelare med texten ”REDO”. Detta indikerar att båda användarna kan trycka för omstart. Färg- samt textskiftning talar sedan om att båda måste vara redo innan spelet startas på nytta, se avsnitt 5.1.2.

Färgkodningen som valts på knappar i spelet har standardfärgerna för det syfte de uppfyller. Exempelvis användes färgen grön för att symbolisera ”start” och färgen röd för att symbolisera ”stopp”. Detta är ett mycket förekommande bruk av dessa färger som är en del av ett universellt språk.

8.1.3 Vilka begränsningar för interaktion och speldesign medföljer ett touchbord som målplattform för ett digitalt Air-hockeyspel?

Eftersom touchbordet var mycket större än de vanligt förekommande touchskärmarna på mobila enheter fanns det begränsningar i interaktionen och speldesignen. Till exempel kunde ingenting klickbart placeras i mitten av spelplanen då risken fanns att spelarna inte når från kortsidorna. Dessutom behövde all text anpassas i storlek och position för att synas från bågge sidorna. Storleksanpassningen gällde även objekten i scenen och utvecklarna måste ha storleksskillnaden i åtanke under utvecklingen av spelet. Ett objekt kunde ha bra storlek på utvecklingsskärmen för att sedan vara för stor på målplattformen. Detta är något som krävde justeringar i efterhand, då olika skärmar krävde olika proportioner.

Den största begränsningen gällde målplattformens responsivitet. På grund av fördröjningen mellan spelet och användarens rörelse uppfattades spelet som icke responsivt. Detta diskuteras mer i avsnittet 7.2.

8.2 Slutsats

Spelet efterliknar ett verkligt Air-hockeyspel, detta med hjälp av realistisk fysik och interaktion, samt dynamiskt ljud. Genom att behålla det klassiska utseendet har Air-hockeyspelets ursprungliga design bevarats, samt har nya teman bidragit till en mer varierad speldesign.

Andra utvecklingsmöjligheter har varit att implementera 3D-grafik, samt introducera ett användargränssnitt. Power-ups har även implementerats för att ge en mer varierad spelupplevelse.

Litteraturförteckning

- [1] Bubble & Air Hockey. *History of Air Hockey* [Internet]. Okänt år. [citerad 2018-02-15] Hämtad från: <http://www.bubbleairhockey.com/air-hockey-history.html>
- [2] Spelbord.se. *Air hockey-historia* [Internet]. Spelbord.se; Okänt år. [citerad 2018-02-15] Hämtad från: <http://www.airhockey.se/air-hockey-historia.asp>
- [3] N734LQ. *Air Hockey table with two paddles and one puck* [Fotografi]. 2007. [citerad 27 april 2018] Hämtad från:
https://commons.wikimedia.org/wiki/File:Air_hockey_table_with_puck_and_paddles.jpg. (Public Domain).
- [4] osde8info. *ipad air hockey app* [Fotografi]. 2011. [citerad 27 april 2018] Hämtad från:
<https://www.flickr.com/photos/osde-info/5522273594/in/photolist-93weKR-arZrtU-qqDd2w-dWuiJn-yNBoqu-cYwnzA-aZna5i-ekyygp-z6eQwV-y9mD5R-y9beFC-riigDn-daKBWg-9pZ6xG>.
(CC BY-SA 2.0) <https://creativecommons.org/licenses/by-sa/2.0/>
- [5] Epic Games Inc. Unreal Engine 4, ver. 4.19 [Programvara]. Epic Games Inc; c2018. [citerad 2018-05-25] Hämtad från: <https://www.unrealengine.com/en-US/what-is-unreal-engine-4>
- [6] Unity Technologies. Unity, ver. 2017.31f1 [Programvara]. Unity Technologies; c2018. [citerad 2018-05-02] Hämtad från: <https://unity3d.com/unity>
- [7] Rešetár M. #1 Make an Air Hockey Game in Unity (Code & Art) [Video]. 2017. [citerad 18 april 2018] Hämtad från:
<http://resocoder.com/2017/04/14/1-make-an-air-hockey-game-in-unity-code-art/>
- [8] Gustafsson M. *TNM040 Kommunikation och användargränssnitt* [Internet]. 2016. [uppdaterad 2016-08-25; citerad 25 maj 2018] Hämtad från:
<https://www.itn.liu.se/grundutbildning/kurs/tnm040?l=sv>
- [9] Norman DA. *The Design of Everyday Things*. 2 rev. uppl. Basic Books; 2013.
- [10] Shari Lawrence Pfleeger och Joanne M. Atlee, *Software Engineering, Fourth Edition, International Edition*, Pearson 2010.
- [11] Schwaber K, Sutherland J. *The Scrum Guide* [Internet]. ScrumGuides.org; c2018. [citerad 2018-02-07] Hämtad från: <http://www.scrumguides.org/scrum-guide.html>
(CC BY-SA 4.0) <https://creativecommons.org/licenses/by-sa/4.0/>
- [12] Google. *Google Kalender* [Internet]. Google; c2018. [citerad 2018-05-10]. Hämtad ifrån:
<https://calendar.google.com/calendar>
- [13] Google. *Google Drive* [Internet]. Google; c2018. [citerad 2018-05-10]. Hämtad från:
<https://www.google.se/drive/about.html>

- [14] GitHub. *GitHub* [Internet] GitHub; c2018. [citerad 2018-02-15] Hämtad från: <https://github.com/features>
- [15] Philipp M. *How to Setup GitHub with Unity: Step-by-Step Instructions* [Internet] Studica; 2015. [uppdaterad 2015-10-13; citerad 2018-04-18] Hämtad från: <http://www.studica.com/blog/how-to-setup-GitHub-with-unity-step-by-step-instructions>
- [16] HacknPlan. *hacknPlan* [Internet]. hacknplan.com; Okänt år. [citerad 2018-05-02] Hämtad från: <http://hacknplan.com/>
- [17] Unity Technologies. *Profiler Window* [Internet]. Unity Technologies; 2018. [uppdaterad 2018-04-04; citerad 2018-04-25]. Hämtad från: <https://docs.unity3d.com/Manual/ProfilerWindow.html>
- [18] Microsoft. Visual Studio, ver. 2017 [Programvara] Microsoft; Okänt år. [citerad 2018-02-15] Hämtad från: <https://www.visualstudio.com/>
- [19] Autodesk Inc. 3DsMax, ver. 2018 [Programvara] Autodesk Inc; 2018. [citerad 2018-03-06]. Hämtad från: <https://www.autodesk.se/products/3ds-max/overview>
- [20] Blender.org. Blender. ver. 2.79b [Programvara] Blender.org; Okänt år. [citerad 2018-05-25]. Hämtad från: <https://www.blender.org/about/>
- [21] Adobe. Photoshop CC, ver. 2017.01 [Programvara] Adobe; 2016. [citerad 2018-03-14]. Hämtad från: <https://www.adobe.com/se/products/photoshop.html>
- [22] Audacity. Audacity, ver 2.2.2 [Programvara] Audacity; 2018. [citerad 2018-05-09]. Hämtad från: <https://www.audacityteam.org/>
- [23] Techopedia Inc. *What is Ray Casting?* [Internet]. Techopedia Inc; c2018 [citerad 2018-05-10]. Hämtad från: <https://www.techopedia.com/definition/21614/ray-casting>
- [24] Halliday D, Resnick R, Walker J. *Principles of Physics*. 10 internationell uppl. John Wiley Sons Inc; 2014.
- [25] Unity Technologies. *Ambient Lighting* [Internet]. Unity Technologies; c2018. [citerad 2018-05-08]. Hämtad från: <https://unity3d.com/learn/tutorials/topics/graphics/ambient-lighting?playlist=17102>
- [26] Unity Technologies. *Lighting* [Internet]. Unity Technologies; c2018. [uppdaterad 2018-04-30; citerad 2018-05-08]. Hämtad från: <https://docs.unity3d.com/Manual/Lighting.html>
- [27] Olsson A. *Röda eller gröna knappar?* [Internet]. uxd.nu; 2014. [uppdaterad 2014-07-01; citerad 2018-05-24]. Hämtad från: <http://uxd.nu/2014/07/01/roda-eller-grona-knappar/>

Bilaga A

Ansvarsfördelning och individuella bidrag

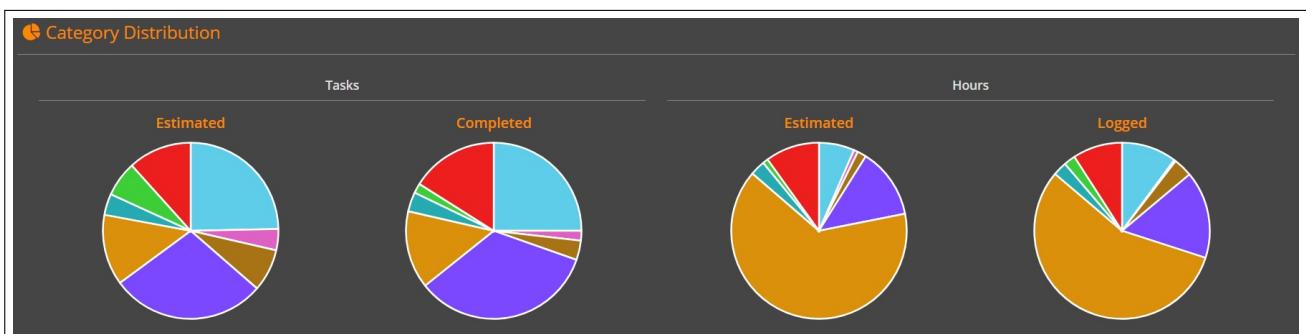
Nedan följer namnen på samtliga gruppmedlemmar, deras ansvarsområde och individuella bidrag:

- **Jennifer Bedhammar - Projektledare:** Hade ansvar över gruppen och möten samt allmän översikt över arbetet för att se till att allt flöt på bra. Om problem uppstod skulle man vända sig till projektledaren för hjälp. Jobbade individuellt med design av teman och menyer, fotbollstemat, implementation av omstartsmenyn och texturer.
- **Pontus Söderqvist - Vice projektledare:** Var projektledarens andra hand och hoppade in vid eventuell frånvaro av projektledaren. Jobbade individuellt med kollisionshantering, multitouch, klubbstyrning, tidtagning och 3D-modellering.
- **Emilie Ho - Kontaktperson och produktägare:** Hade kontakt med kunden/examinatoren och förde kundens talan. Ansvarade även för projekt-backloggen och sorterade uppgifter efter prioritering. Jobbade individuellt med design av teman och menyer, 3D-modellering, power-ups och FPS-räknare.
- **Aniisa Bihi - Sekreterare och dokumentansvarig:** Antecknade under officiella möten. Såg även till att hålla reda på alla dokument och såg till att Google Drive mapparna var organiserade. Jobbade individuellt med design av teman, samt utformandet av neontemat och godistemplet med tillhörande texturer.
- **Elias Elmquist - Rapportansvarig och vice sekreterare:** Såg till att rapporten skrevs och att rätt dokumentation fördes. Jobbade individuellt med en 2D-variant av ett Air-hockeyspel och överförde relevanta funktioner till den slutgiltiga 3D-varianten av spelet och implementerade stödet av ljud i spelet.
- **Peilin Yu - Testningsansvarig:** Ansvarade över att testning av systemet skedde kontinuerligt under projektet. Var även ansvarig över att lägga upp strukturen för användartester. Jobbade individuellt med design av omstartsmeny, programmering i kollision, fysiksimulering och multitouch.
- **Malin Ejdbo - Versionsansvarig och filhanterare:** Var ansvarig över Hacknplan och GitHub. Var även ansvarig för att all kod kommenterades på lämpligt sätt. Jobbade individuellt med power-ups, utvecklarfönster, prefabs och organisation av scenen i Unity.

Bilaga B

HacknPlan statistik

I figur B.1 följer statistik från HacknPlan över hur många uppgifter som i början av projektet antogs krävas för att implementera projektet och hur många som har implementerats. Det visas även hur lång tid de olika kategorierna av projektet förväntades att ta och hur mycket tid som spenderades på dem i slutändan. De två första diagrammen jämför antalet uppgifter som antogs i början av projektet och hur många som faktiskt implementerades. De två sista diagrammen jämför den tid som antogs spenderas på de olika kategorierna och den tid som faktiskt lades ner på de olika kategorierna.



Figur B.1: Statistik i HacknPlan. De färgade områdena representerar de olika kategorierna som fanns under projektet. Det orangea området symbolisering dokumentation och rapportskrivning, lila är programmering, röd är buggar, ljusblå är design, grön är modellering, grönbrown är användartester, rosa är ljuddesign och brun är idéer.

Bilaga C

Användartester - Frågor

Nedan följer de frågor som ställdes till sex stycken användare under användartesterna.

Responsivitet

- Vad tycker du om responsiviteten i spelet?
- Vilken version tycker du bäst om? Hastighetsversionen eller positionsversionen?
- Märker du någon skillnad mellan de två versionerna?
- Anser du att någon version är mer irriterande eller långsammare?

Grafik och gränssnitt

- Vad tycker du om grafiken?
- Vad tycker du om gränssnittet?
- Förstår du vad du ska göra i spelet? (som exempelvis i omstartsmenyn)

Power-ups

- Förstår du vad de gör?
- Hur tycker du designen borde vara?

Förbättringar

- Har du något förslag på förbättringar?
- Något du vill tillägga? Övriga kommentarer.

Bilaga D

Användartester - Sammanställning

Under användartesterna genomfördes intervjuer med användarna om deras uppfattning och åsikter om spelet i dess dåvarande skick och utformning. I denna bilaga finns en sammanfattning av de svar som erhölls i dessa intervjuer.

D.1 Första intrycket

Alla användare ansåg att spelet var roligt och hade lätt att förstå sig på hur de skulle spela. Däremot så uppstod större förvirringar hos vissa av spelets funktioner, så som power-ups. Spelet upplevdes även som hackigt och många användare förlorade kontrollen över klubban när de spelade. En ökad förståelse hos användarna uppstod efter en genomgång av responsivitet och de två olika versionerna som fanns tillgängliga. En av användarna hade aldrig spelat verkligt Air-hockey innan, men förstod ändå hur spelet skulle spelas.

D.2 Responsivitet

Majoriteten av användarna upplevde att positionsversionen var bättre än hastighetsversionen. För vissa märktes inte skillnaden före de fick det påpekat och var inställda på att det skulle finnas en skillnad. Det som uppmärksammades var att positionsversionen känns kvickare, mer responsiv och att den hängde med fingrets position bättre. Dess negativa aspekter var att det var lättare att tappa kontrollen hos klubban och att hastigheten ökade i spelet.

Användarna som föredrog hastighetsversionen ansåg att den var lättare att styra klubban med och att den följde efter fingret bättre trots fördräjningen. De negativa aspekterna hos hastighetsversionen höll även de som föredrog hastighetsversionen med om. Dessa var att den var långsammare och att den påverkade användarupplevelsen negativt mer än positionsversionen.

D.3 Generellt om grafiken

Flera användare ansåg att spelet såg ut som en kommersiell produkt. Användarna gillade att det var stora siffror vid nedräkningen i slutet av spelet. Grafiken gjorde så att de kände igen sig, men upplevdes av en användare som tråkig även om denne förstod sig på funktionerna.

D.4 Godistemat

Detta tema tyckte många var både fint och en rolig idé. Vissa tyckte att det var en intressant förändring men onödig. Även fast alla deltagare förstod att temat var godis så uppfattades temat ändå som osammankopplade då bakgrundens inte var godisinspirerad. Puckens textur var även något som kunde förbättras då den inte uppfattas som en godisbit.

Generella synpunkter var att sargent borde vara tydligare, timern borde matcha temat, bättre ljussättning och temaenliga power-ups.

D.5 Fotbollstemat

Detta tema var den som kändes mest malplacerat, detta då det inte ansågs vara logiskt att spela Air-hockey på en fotbollsplan. Temat ansågs dock fortfarande vara roligt och snyggt gjort, speciellt ljussättningen, men färgen hos klubborna kunde varit bättre. Något som nästan alla deltagare påpekade var att pucken borde se mer ut som en boll och agera som en boll, alltså rulla. En deltagare fick förklarat för sig tanken bakom fotbollstemat och hur fysiska element som friktion skulle förändras. Denna deltagare tyckte att den idén låt väldigt intressant och tyckte det var tråkigt att det inte hanns med.

D.6 Neontemat

Detta tema var alla deltagares favorittema. De tyckte att temat var innovativt, minimalistiskt, snyggt och passade stilens hos arkadspel samt ett digitalt spel. Temat upplevdes som ”snabbt” och Tron inspirerat, något som neonfärgerna och pucken bidrog till. Det påpekades dock att sargent inte var synlig och att det saknades ett djup hos målet.

D.7 Generellt om användargränssnittet

Användarna tyckte det var bra att de inte kunde dra över klubban till den andra spelarens planhalva. De tyckte även om att man kunde se vem som slog pucken senast, däremot förstod inte vissa varför det var så. Många tyckte det var svårt att inte göra självmål och att pucken borde släppas ner i mitten av spelplanen efter varje mål. Detta för att det ska vara lika rättvist varje gång en spelare har gjort mål.

D.8 Ljud

Ljudet gav en mer verklighetstroget upplevelse för många av deltagarna. Det som påpekades var att ljudet passade det klassiska temat och att de tyckte om att det låt som hockey. Deltagarna ansåg att det hade kunnat finnas ljudrespons när spelet höll på att ta slut och när power-ups togs upp.

D.9 Omstartsmenyn

Menyn var inget som förvirrade deltagarna och ansågs som väldigt tydlig. De förstod att de skulle trycka på redo-knappen och sedan vänta på den andra spelaren. Det var en deltagare som ansåg att knappen kunde ha varit större. Ett missförstånd som uppstod hos många var att de läste ”redo” på engelska, som betyder ”göra om”. Många ansåg därför att detta ord skulle bytas ut då det var konstigt att allt var på svenska förutom just det ordet.

D.10 Power-ups

Det var en ständig fråga under majoriteten av användartesterna på vad ”kuberna” innebar. Det tog en stund för vissa att förstå vad det var, medan vissa inte alls förstod förrän de fick det förklarat för sig. Naturligt för alla deltagare var att försöka fånga upp en power-up med klubban, många insåg dock snabbt att de var tvungen att träffa den med pucken, något som ansågs vara både positivt och negativt. Majoriteten ansåg att antingen tydligare former eller iconer hade varit en bättre samt tydligare design. Även tydligare signalementen som ger uppfattningen av att spelaren kan plocka upp dem, alltså borde de inte vara så genomsiktig som de är. Även fast mycket förvirring uppstod så ansåg majoriteten att det var en rolig funktion som gjorde spelet unikt. Om designen skulle vara tydligare så hade det ökat förståelse och även förhöjt användarupplevelsen.

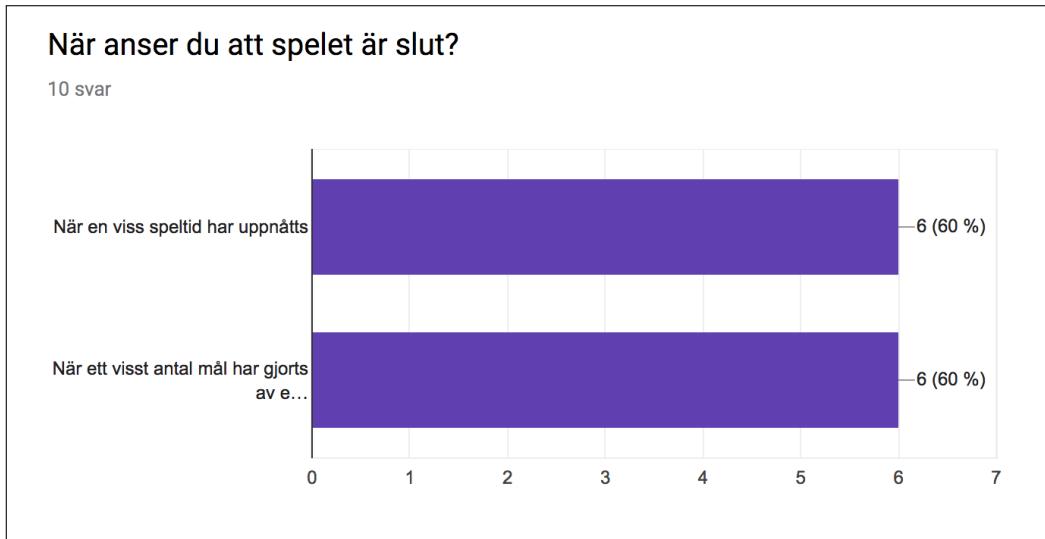
Bilaga E

Användartester - Svarsformulär

Nedan följer en sammanställning av diagram baserade på de svar som lämnades in av användarna efter användartesterna.



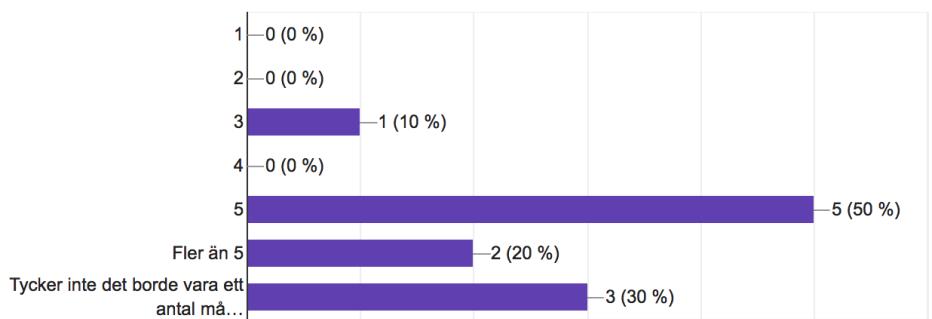
Figur E.1: Svar på fråga 1.



Figur E.2: Svar på fråga 2.

Hur många mål bör en spelare få för att vinna?

10 svar



Figur E.3: Svar på fråga 3.

Hur lång tid bör en match vara?

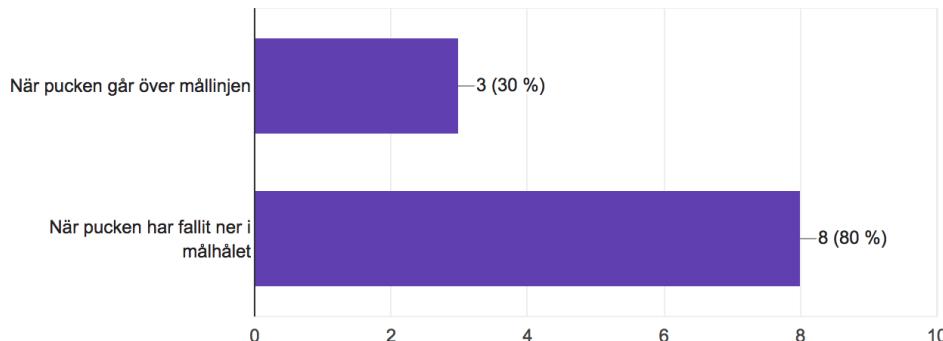
10 svar



Figur E.4: Svar på fråga 4.

När tycker du att det är mål?

10 svar



Figur E.5: Svar på fråga 5.