**Color-Game**

**Zielsetzung und Beschreibung**

Die Zielsetzung des dritten Minispiel, im folgenden Color-Game genannt, war es neben einem Spiel mit einer Verbindung zu Wahrscheinlichkeiten ein eigenständiges Rätsel auszuarbeiten, das autonom als Spiel funktionieren kann. Dabei kann der Spieler selber Entscheidungen treffen die den gesamten Verlauf des Minispiels beeinflussen sollten. So wurde die Designentscheidung getroffen, dass das Spiel aus mehre Runden bestehen soll.

Als Spielmechanismus wurde ein Programmier-Mechanismus aus Brettspielen wie „Lords of Xidit“, „Robo Rally“ oder „AquaSphere“ als Vorbild genommen. Bei diesen Spielmechanismus soll der Spieler seine Spielzüge nicht einen nach den anderen ausführen und sofort Rückmeldung erhält, sondern erst alle seine Aktionen festlegen bevor diese durchgeführt werden. Um eine Konfliktkomponente einzuführen, soll abwechselnd mit dem Spieler das Programm selbst Spielzüge festlegen, die sich auf die Entscheidungen des Spielers auswirken sollen.

Die Aufgabe ist es dabei auf einem Spielbrett, bestehend aus 16 als Raster angeordneten Feldern, eine Spielfigur von einem Startfeld auf ein Zielfeld zu bewegen. Dabei sollte der Spieler, sowie das Programm je 4 Bewegungen festlegen. Die Schwierigkeit für den Spieler besteht darin mögliche Bewegungen der Computerzüge auszugleichen. Da alle Schritte erst nach der Wahl der letzten Bewegung ausgeführt werden, ist es Aufgabe des Spielers sich die mögliche momentane Position der Spielfigur auf dem Spielfeld durch die zuvor getroffenen Entscheidungen herzuleiten und zu merken. Um das einzelne Minispiel nicht zu lang und schwierig zu gestalten, wurde festgelegt dass die Spielerin nicht mehr als 4 Entscheidungen treffen sollte, was mit den zusätzlichen Bewegungen des Programmes zu 8 Bewegungen in einer Spielrunde führten.

Bei der Auswahl der möglichen Farbtafeln entschied man sich für einen festen Satz an Bewegungsmöglichkeiten von denen bei Spielstart zufällig 4 gewählt werden. Dies mindert die Aussicht der möglichen Situationen in denen der Spieler nicht gewinnen kann. So setzt sich der Spieler beim Color-Game nicht nur mit Wahrscheinlichkeiten auseinander sondern baut seine Fähigkeiten Vorauszuplanen aus und schult sein Gedächtnis.

Um den Schwierigkeitsgrad nicht weiter zu erhöhen, soll die mögliche Auswahl des vom Programm gewählten Feld so erkennbar wie möglich sein, ohne das Spielelement unwirksam zu machen. So war es wichtig in der späteren Umsetzung dafür zu sorgen, dass der Spieler auf einen Blick alle Informationen sehen kann. Ein wiederholtes Wechseln der Blickrichtung um das Spielfeld oder die Auswahl zu sehen sollte vermieden werden, da dieses den Denkprozess unnötig erschweren würde. Eine Variation ist möglich aber für den ersten Kontakt mit den Spiel und der Regeln unnötig und kann an einer anderen Stelle für einen höheren Schwierigkeitsgrad genutzt werden, nachdem das Spiel schon einmal absolviert wurde.

**Entwicklungsstufen des Color-Game**

Für die ersten Versionen wurden Konzeptzeichnungen und Beschreibungen des Spielablaufs angefertigt. Danach wurde ein spielbarer Papierprototyp erstellt der auch die Rolle des Programms simuliert. Dieser besteht aus einem Spielfeld, Richtungsmarker, einen Kartenstapel aus 8 Karten mit den Bewegungswahrscheinlichkeiten, sowie Würfel für die zufällige Bestimmung des Start- und Zielfeldes. Die Karten repräsentieren dabei die Wahrscheinlichkeiten welche Richtung das Programm wählt, abhängig davon auf welchem Feld sich die Figur zu dem Zeitpunkt an dem die Karte ausgeführt wird befindet. Die Wahrscheinlichkeiten werden dabei als Farben Grün, Blau, Orange und Magenta dargestellt. Dies ist auch die Prioritäten-Reihenfolge in der sich das Programm für eine bestimmte Bewegungsrichtung entscheidet. Sie wurde festgelegt um das Spiel einfacher zu halten. Falls mehrere Felder der höchsten Priorität zur Wahl stehen, wird eines der Felder zufällig gewählt.

Der Spielablauf konnte mit diesem Prototyp, wie folgt, dargestellt werden. Nach der Bestimmung des Start- und Zielfeldes mit den Würfeln zieht der Spieler abwechselnd eine Bewegungsrichtung und eine Farbkarte von dem vorher gemischten Kartenstapel. Nach der letzten Wahl wird die Spielfigur entsprechend der gewählten Richtungen auf dem Spielfeld bewegt. Sollte sich die Spielfigur nach der letzten Bewegung auf dem Zielfeld befinden, gilt das Spiel als gewonnen, ansonsten kann eine neue Runde gespielt werden.

Der erste digitale Prototyp wurde wie das Endprojekt in Unity programmiert. Dabei war die erste Version eine Umsetzung des Papierprototypens bei der der Spieler alle Spielkomponenten von oben herab betrachtet. Die Interaktion beschränkte sich bei dieser Version auf das wählen einer Bewegungsrichtung mit den Pfeiltasten und das Bestätigen der Wahl mit der Return Taste.

Eine spätere Version wurde als ein eigenes Level in eine Inselumgebung umgesetzt mit der von Unity zur Verfügung gestellten First Person Kontroller und für das Projekt erstellten Funktionen für die Interaktion mit der Umgebung. Durch die neuen Interaktionsmöglichkeiten wurde das wählen der Bewegungsaktion durch Tasten aus dem Spiel entfernt. Durch die neue Bewegungsfreiheit des Spielers musste die einzelnen Spielelemente neu angeordnet werden. Diese sollten in die Umgebung eingearbeitet werden. Zugleich sollte der Spieler bei der Auswahl seiner Bewegungen das Spielfeld, sowie die Farbtafel im Blickfeld haben.

In der nächsten Version wurden das neue 3D Modelle für die Schalter der Bewegungsauswahl eingesetzt. Diese Version des Spieles konnte dann in das Gesamtprojekt exportiert werden und wird dort weiter bearbeitet.

Durch das Gewinnen des Spiels sollte sich ein Weg für den Spieler eröffnen um zum nächsten Gebiet der Insel und dadurch zum nächsten Rätsel zu kommen. Das Hindernis das es zu überwinden galt war ein See der nicht betreten werden konnte. Das Spielbrett wurde dazu in diesen See gesetzt. In der ersten Version stiegen, sobald der Spieler das Spiel gewann, Steinplatten aus dem Wasser. Über diese konnte sich der Spieler bewegen. Da diese aber keinen durchgehenden Weg darstellten musste der Spieler an der richtigen Stelle springen um auf die andere Seite zu kommen. Dies stellte ein weiteres Spielelement dar.

Im Laufe der Entwicklung wurde die Möglichkeit des Springens entfernt um zu verhindern, dass Spieler an ungewünschte Positionen gelangen konnte. Entsprechend musste der Weg der zum nächsten Areal führte bei dem dritten Rätsel geändert werden. Dabei entschied man sich ein Boot einzubauen, das sobald der Spieler das Spiel erfolgreich beendet hat, ihn auf die andere Seite bringen würde. Dies sollte automatisch passieren sobald der Spieler das Boot aktivierte. Der Spieler hat dabei keinen Einfluss auf den Weg den das Boot einschlägt und kann sich bis zur Ankunft auf der anderen Seite nicht bewegen. Die Landung auf der anderen Seite des Sees beendet das dritte Rätsel und der Spieler ist frei sich noch etwas in der Umgebung umzusehen bevor er sich zum letzten Rätsel des Spiels aufmacht.

**Umsetzung im Code**

Da es eine Designentscheidung war ein festes Set von Farbfeldern zu nehmen die die Wahrscheinlichkeiten für die Bewegung des Programms repräsentieren, sind diese fest im Code verankert. Zu diesem Zweck ist jede Karte mit der entsprechenden Farbposition als Array repräsentiert. Vier Karten wurden umgesetzt, da es mit der zufälligen Wahl der Reihenfolge und die des Ziel- und Startfelds genug Variation gibt, so dass sich kein Spiel wie das vorherige für den Spieler anfühlt.

In der ersten digitalen Umsetzung wurde im ersten Schritt an der Bewegung gearbeitet. Zu diesen Zweck war es eine Funktion zu schreiben die eine Position übergeben bekommt und die Spielfigur entsprechend bewegt. Da es eine feste Anzahl an Positionen gibt, die abhängig von der Position des Spielbrettes sind, wurden die Positionen fest in den Code gebunden.

Eine weitere wichtige Funktion war es die Bewegungswahl abzuhandeln. Die Wahl der Bewegung des Spielers wird nach jeder Interaktion aktualisiert. Da die Wahl immer aus dem Set der Bewegungen nach links, rechts, oben und unten besteht und die Auswahlschalter immer in der gleichen Ausgangssituation starten, hat man sich dazu entschieden durch dieses Set zu iterieren. Bei jeder Interaktion wird eine Animation der Schalter abgespielt und die aktuelle Auswahl gespeichert. Diese gewählte Bewegung wird weitergegeben sobald der Spieler die Wahl mit einem weiteren Schalter bestätigt. Dabei wird das aktuelle Paar aus Schaltern für die Bewegungswahl und Bestätigungsschalter deaktiviert und das nächste Paar aktiviert.

Bei der Auswertung welche Position vom Programm gewählt wird, werden zuerst alle benachbarten Felder in einen Array zwischengespeichert. Dabei werden die Wahrscheinlichkeiten der Farben als Integer gespeichert, die kleinste Zahl entspricht dabei der höchsten Wahrscheinlichkeit. Die Felder werden sortiert und das mit der höchsten Wahrscheinlichkeit zurückgegeben. Als Sortieralgorithmus hat man sich für einen Bubblesort entschieden, da es sich um eine sehr geringe Menge an Daten handelt. Der Bubbelsort legt bei dieser Menge an Daten eine vertretbare Laufzeit an den Tag, die bei dem Minispiel dem Spieler nicht auffallen wird und deshalb nicht ins Gewicht fällt.

Alle gewählten Positionen, durch den Spieler oder des Programmes, werden in eine Queue gespeichert. Die nächste Position wird dabei durch die aktuelle Position und die getroffene Wahl der Bewegung berechnet. Diese Queue wird nach der letzten Auswahl abgearbeitet und die Spielfigur nacheinander um jeweils eine Position bewegt.

Nach der letzten Bewegung wird überprüft ob die momentane Position der Figur mit dem Zielfeld übereinstimmt. Falls dieses nicht der Fall ist, wird das Spiel zurückgesetzt und der Spieler kann erneut spielen. Dabei bleibt das Zielfeld immer gleich und das Startfeld der Spielfigur ist das Feld der letzten Bewegung der vorherigen Runde. Hat der Spieler hingegen das Spiel gewonnen, werden die Schalter deaktiviert und die mögliche Interaktion mit dem Boot aktiviert.

Der Transport des Spielers nach dem Aktivieren des Bootes wird mit einer Animation in Unity umgesetzt. Die Fähigkeit des Spielers sich zu bewegen wird für diesen Zeitraum außer Kraft gesetzt und wieder aktiviert sobald die komplette Animation abgelaufen ist.