TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG
Die Ressourcenuniversität. Seit 1765.

Fakultät für Mathematik und Informatik
Institut für Informatik

Studienarbeit

OpenGL Game: Achtung, die Kurve!

Simon Al Nomer
Hernán Felipe Valdés González

Bachelor Angewandte Informatik
Matrikel: 64 082
63 952

19. Oktober 2020

Betreuer/1. Korrektor:
M.Sc. Jonas Treumer

2. Korrektor:
M.Sc. Ben Lorenz

Eidesstattliche Erklärung

Ich versichere, dass ich diese Arbeit selbstständig verfasst und keine anderen Hilfsmittel als die angegebenen benutzt habe. Die Stellen der Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen sind, habe ich in jedem einzelnen Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht. Diese Versicherung bezieht sich auch auf die bildlichen Darstellungen.

19. Oktober 2020

Simon Al Nomer
Hernán Felipe Valdés González

Zusammenfassung

Implementierung des Spieles "Achtung, die Kurve" in C++ mit OpenGL für die Lehreveranstaltung "Multimedia".

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

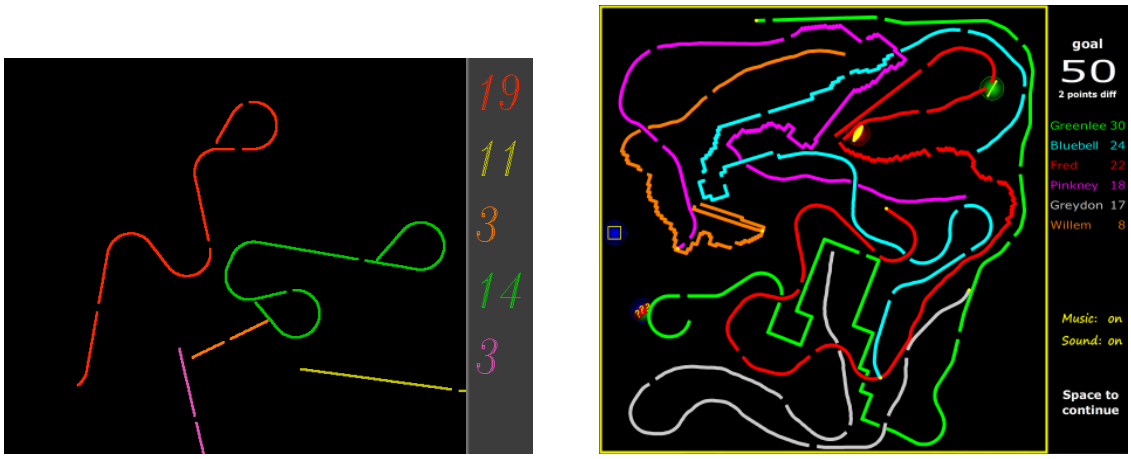
1 Einführung

Achtung, Die Kurve ist ein Multiplayer-Spiel, welches im Jahr 1995 von Filip Oščádal und Kamil Doležal in DOS entwickelt wurde. Im Jahr 2010 wurde eine neue Version des Spiels unter dem Namen "Achtung, die Kurve! Flash Remake" veröffentlicht. Diese Version ist mit Adobe Flash von Geert van den Burg entwickelt worden und kann online gespielt werden.

Nachdem das Spiel großen Zuspruch fand, beschloss van den Burg, sich mit Robin Brouns zusammenzuschließen und eine Fortsetzung zu entwickeln, welche 2011 unter den Namen Curve Fever (Originaltitel "Achtung, die Kurve! 2") veröffentlicht wurde und online gespielt werden kann.

In den folgenden Jahren wurden dem Spiel immer weiter neue Features hinzugefügt wie zum Beispiel verschiedene Boni, die Möglichkeit, in einem Team zu spielen, oder das Bestehen einer Rangliste.

2015 sammelte van den Burg ein Team um sich, um eine neue Version des Spiels in HTML5 zu implementieren, welche im September 2016 auf den Markt kam.



(a) Original version für DOS (1995). **(b)** Flash remake (2010).

Abb. 1: 2 Versionen, die als Basis für das Spiel genommen wurden.

Die Implementierung unseres Spiels basiert auf die Version von 1995 in Kombination mit dem Stil der Flash Version.

1.1 Spielmechanik

Im Spiel können bis zu 6 Spieler zusammen mit einem Bildschirm und einer Tastatur spielen. Jeder Spieler besitzt zwei vorbestimmten Tasten, die rechts und links symbolisieren. Um das Spiel zu starten, müssen mindestens 2 Spieler spielen. Der Ziel vom Spiel ist, dass der Spieler am Leben solange wie möglich bleibt.

Jeder Spieler wird mit einem Kreis dargestellt, der mit jeder Bewegung einen Pfad mit der Farbe des Spielers hinterlässt. Der Spieler darf nur nach rechts und links die Richtung seiner Bewegung ändern. Wenn der Spieler nicht reagiert, wird er weiter in der alten Richtung sich bewegen.

Ein Spieler verliert, wenn er gegen seine eigene Linie, die Linie anderen Spieler, oder den Rand gestoßen hat. Das Spiel ist zu Ende wenn nur ein Spieler noch am Leben ist.

2 Softwaredokumentation

Unseres Projekt hat als basis den Code, der zusammen während der Vorlesung erstellt wurde, genommen und in C++ migriert. Die Entscheidung, um das Spiel in C++ statt C zu programmieren, lag insbesondere wegen der Standardbibliothek von C++ und deren implementierten Daten Strukturen sowie die Eigenschaft mit Klassen zu programmieren.

Für die Softwarearchitektur wurde das State Pattern implementiert, um einen endlichen Automaten zu simulieren und die Änderungen der Zuständen sind in der Klasse *Game* durchgeführt.

Das Spiel wurde im Sinne von Szenen programmiert. Die Klasse *Game* verhaltet sich als eine Szene sowie als eine Szene-Manager. Die anderen 2 Szenen sind *Menu* und *GameOver*.

Die Daten von dem laufenden Spiel, sowie die Anzahl an Spieler und ihren Punkten werden in einem Struct gespeichert, der dann in GLFW zur Verfügung gestellt wird. Dieser Struct heißt *user_ data_ t*.

2.1 Game

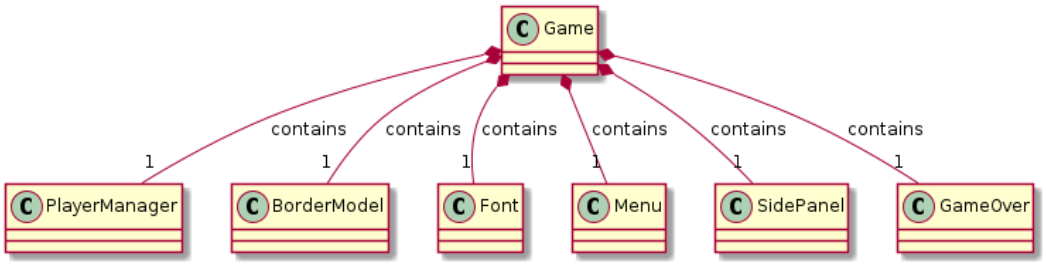


Abb. 2: UML Diagramm von der Klasse Game

Die zentrale Klasse unseres Projektes ist Game. In dieser Klasse werden die Zustände von unseren Automat geändert.

In Game wird die Klasse Font initialisiert und dann mit anderen Anderen Klassen mitgeteilt.

2.2 GameState

Das Spiel benutzt das State Design-Pattern, in dem wir einen Zustandsautomat simulieren. Der Startzustand unseres Automaten ist GAME_MENU.

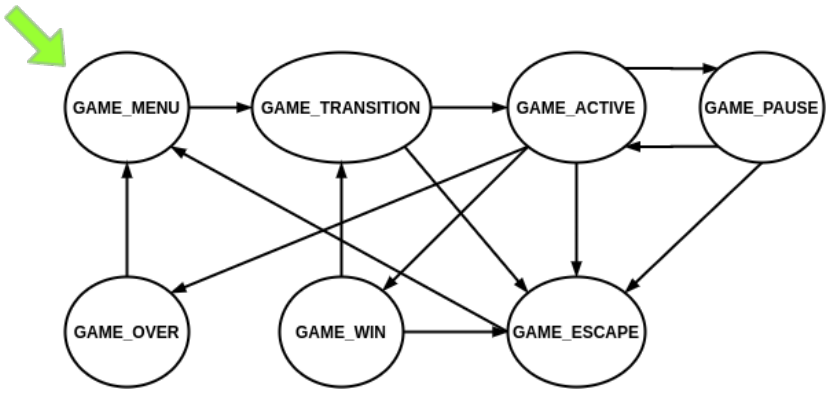


Abb. 3: Darstellung unseren Zustandsautomat

GAME_MENU Es wird das Menü für den Auswahl von Spieler gezeigt. Man kann nur zu den nächsten Zustand gehen, GAME_TRANSITION, in dem man mindestens 2 Spieler ihre Anwesenheit bestätigen und dann die SPACE-Taste gedrückt wird.

GAME_TRANSITION Das Spiel ist bereit zu starten. Alle Spieler sind in ihren Plätze und ihren Position wird mit einem gefarbten Kreis gezeichnet. Um das Spiel zu starten muss die Leertaste gedrückt werden und der neuer Zustand wird GAME_ACTIVE sein. Um zurück ins Menü zu gehen, muss die ESCAPE-Taste gedrück werden und der neuer Zustand wird GAME_ESCAPE.

GAME_ACTIVE Das Spiel läuft und die Spieler dürfen anderen schließen und in die Irre führen. Man kann in jedem Moment die SPACE-Taste drücken, um das Spiel zu pausieren, und der neuer Zustand wird GAME_PAUSE. Man kann auch mit der ESCAPE-Taste das Spiel abbrechen und zurück ins Menü gehen, dann der neuer Zustand wird GAME_ESCAPE. Wenn nur ein Spieler übrig auf dem Feld ist, der Zustand wird automatisch zu GAME_WIN oder GAME_OVER gehen, abhängig ob die maximalle Punktzahl erreicht wurde oder nicht.

GAME_PAUSE Das Spiel ist pausiert. Man kann zurück zu GAME_ACTIVE mit der SPACE-Taste gehen oder zu GAME_ESCAPE mit der ESCAPE-Taste.

GAME_ESCAPE Zwischenzustand um die Daten wieder ins Default bringen. Es wird direkt zu GAME_MENU gehen.

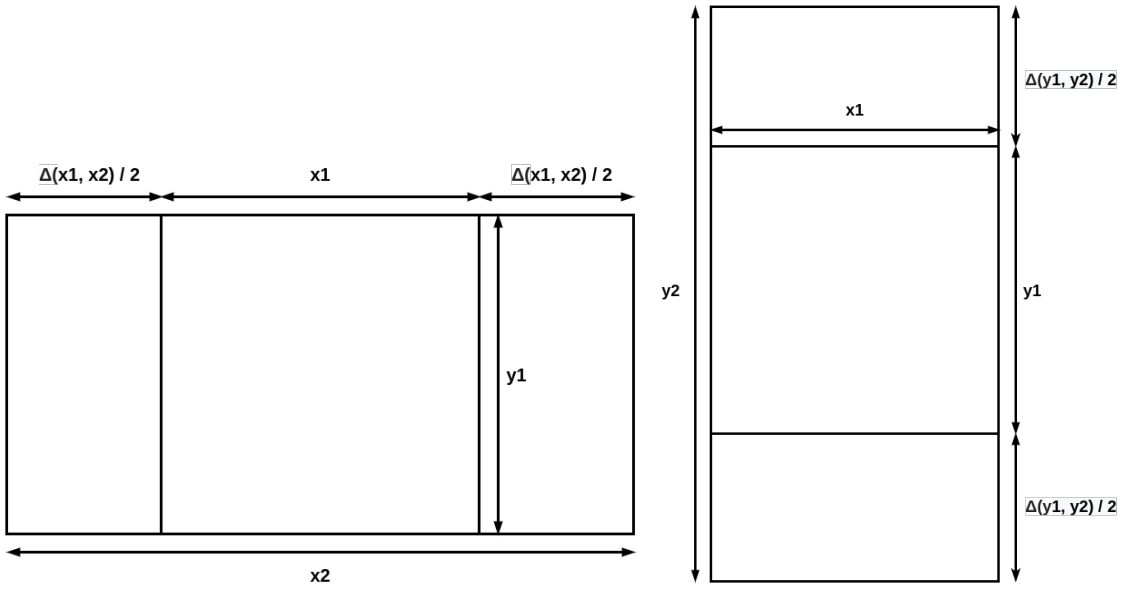
GAME_WIN Zwischenzustand um den Gewinner der Runde zu zeigen (mit einem blinken Namen). Nach ein paar Frames wird direkt zu GAME_TRANSITION gehen umd eine neue Runde zu starten.

GAME_OVER Es wird eine Liste mit der gesamten Punktzahl gezeigt. Wenn man den SPACE-Taste druck, geht man zu GAME_MENU und startet alles von vorne.

2.3 Display

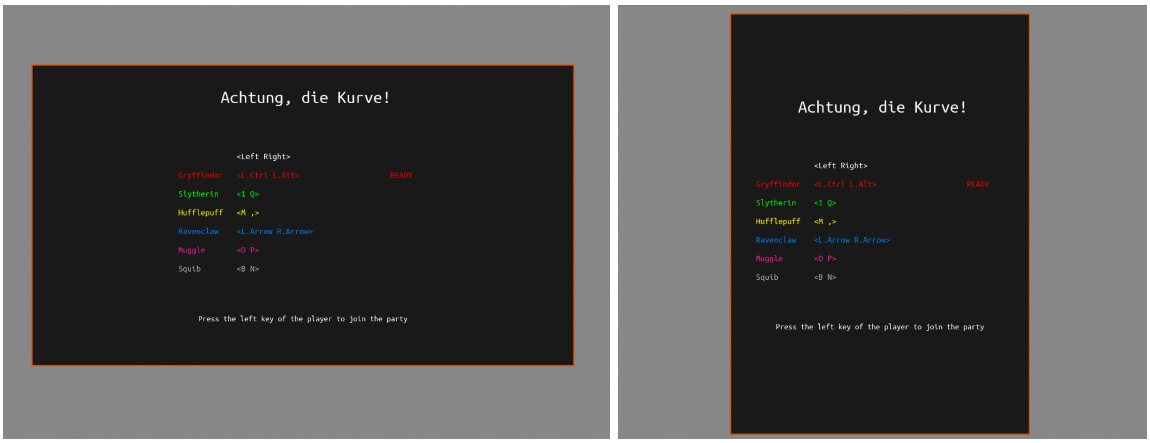
Die Klasse Display initialisiert und besitzt ein Instanz von GLFWwindow. Ein Pointer von dem Fenster wird dann mit der Klasse Game mitgeteilt. Display wird dann am Ende des Programms auch das Fenster schließen (beenden).

Das Fenster hat ein ursprünglichen Ratio von 1:1 mit der Größe 700x700 pixels. Das Fenster wird sein Aspect Ratio behalten, in dem immer die kleinste Wert zwischen Höhe und Breite nimmt und dann positioniert sich in der Mitte von der gegende Position. z.B: Wenn die Höhe größer als die Breite ist, zentriert das Fenster sich in der Mitte von der Höhe mit der Größe von der Breite.



(a) Anpassung in horizontellen Resize **(b)** Anpassung in vertikallen Resize

Abb. 4: Anpassungen bei der Änderung in der Größe des Fensters



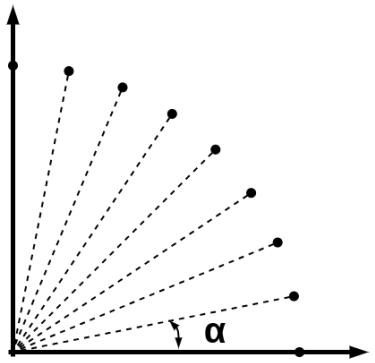
(a) Erstellung von den Punkten mit der Kreis-Gleichung **(b)** Darstellung von GL_TRIANGLE_STRIP

Abb. 5: Screenshots von dem Beispiel beim Resizing

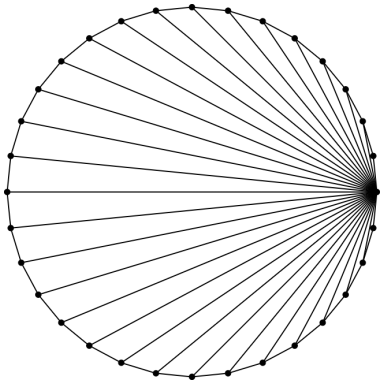
3 Zeichnungen

3.1 Player

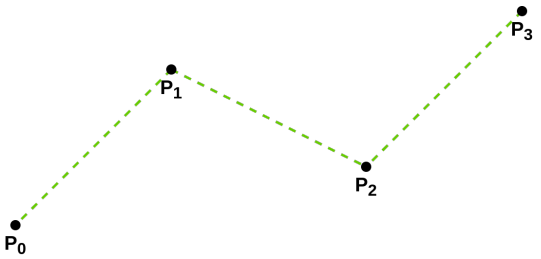
3.2 Linien



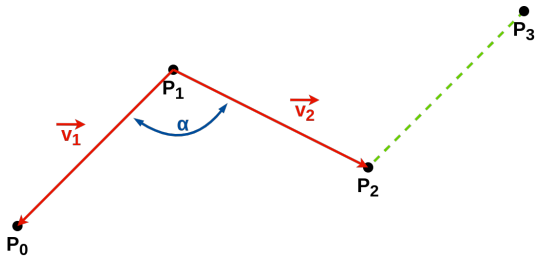
(a) Erstellung von den Punkten mit der Kreis-Gleichung



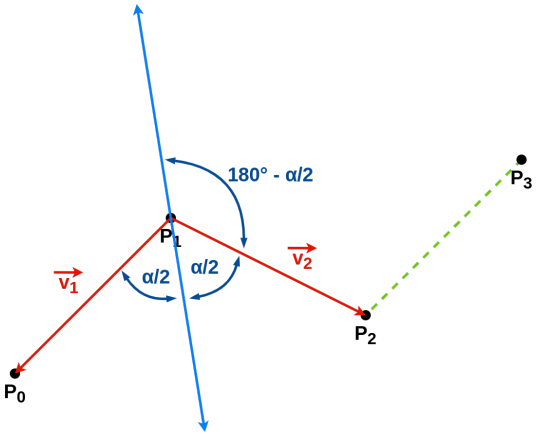
(b) Darstellung von GL_TRIANGLE_STRIP



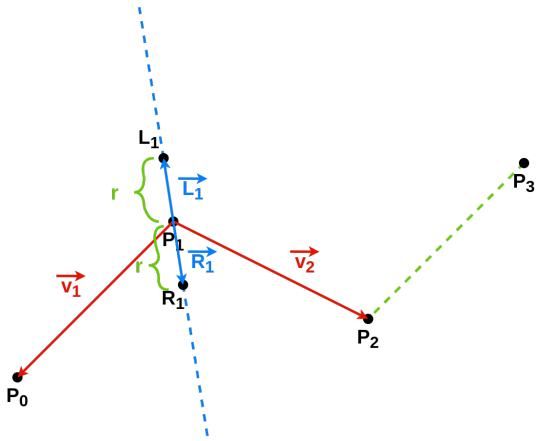
(a) Erstellung von den Punkten mit der Kreis-Gleichung



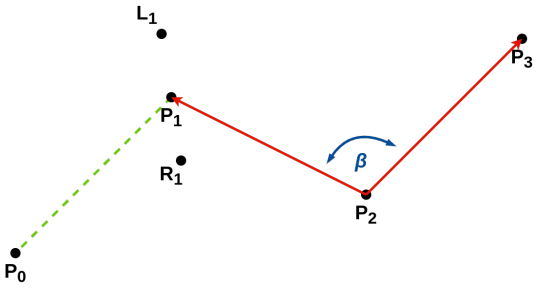
(b) Darstellung von GL_TRIANGLE_STRIP



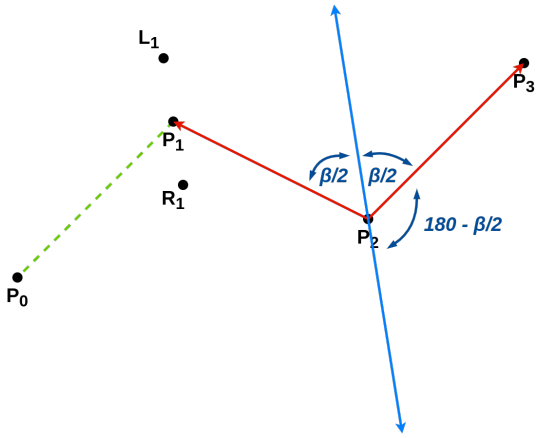
(c) Darstellung von GL_TRIANGLE_STRIP



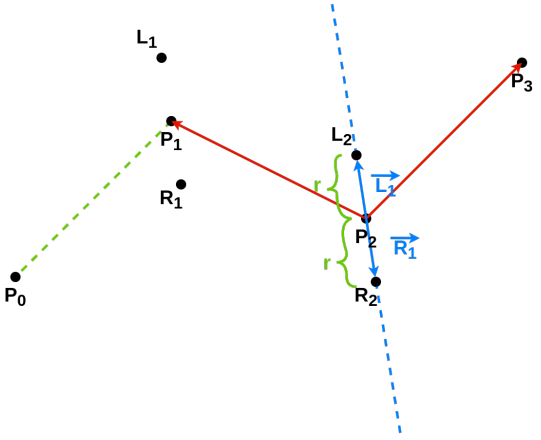
(d) Darstellung von GL_TRIANGLE_STRIP



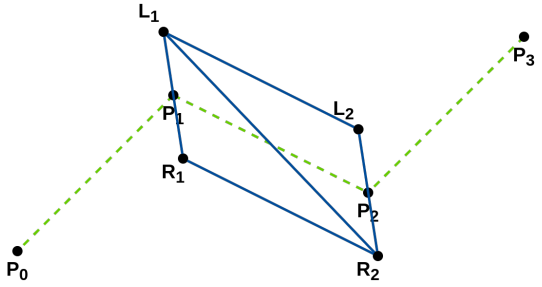
(e) Darstellung von GL_TRIANGLE_STRIP



(f) Darstellung von GL_TRIANGLE_STRIP



(g) Darstellung von GL_TRIANGLE_STRIP



(h) Darstellung von GL_TRIANGLE_STRIP

Abb. 7: Darstellung von GL_TRIANGLE_STRIP

4 Kollisionen

5 Projektdokumentation

6 Benutzerhandbuch

Abb. 8: Menü Szene.

Abb. 9: Bestätigung der Anwesenheit von Spieler.

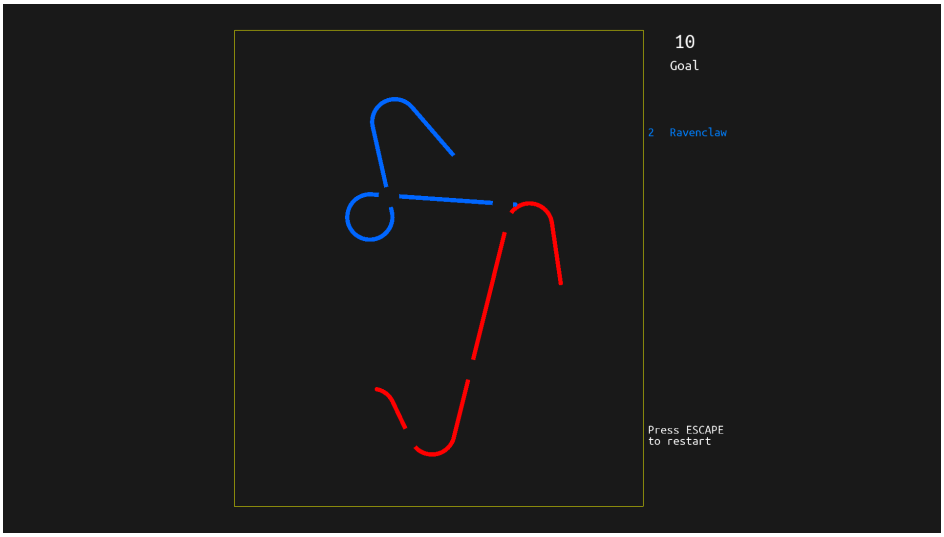


Abb. 10: Ein Spiel pausiert.