Dungeon Crawler

Ich habe mich in den letzten Wochen mit den Kamera-Objekten in Unity beschäftigt. Kamera sind vor allem für alles was man sehen und hören kann zuständig. Dies beinhaltet das Rendern der Grafiken, die Art der Audioaufnahme, aber die Kamera kann mit den passenden Skripts auch praktische Befehle wie das Zerstören von Objekten durchführen. Zuerst habe ich mich mit dem HDR (High Dynamic Range-Rendering) beschäftigt. Dieses lässt Farbkanäle außerhalb von 0 bis 1 zu. Damit können unterschiedliche Töne derselben Farbe besser und vor allem realistischer dargestellt werden. Im Spiel Dungeon-Crawler hat dies besonders Einfluss auf die Fackeln, welche einen weitaus wärmeren Gelb-Ton ausstrahlen als zuvor, was durchaus Sinn ergibt, da es sich bei Fackeln um Feuer und nicht um LED-Lampen handelt.

Zu dem HDR gehört auch das Tonemapping. Durch das verändern der Grafiken mit Hilfe von HDR sind diese nicht mehr mit dem Bildschirm kompatibel. Hier kommt das Tonemapping ins Spiel. Es wandelt die Grafiken mit auswählbaren Techniken wieder in den LDR-Bereich um. Diese Techniken sind sehr unterschiedlich, zum Beispiel die Technik „Adaptive Reinhard Auto White“, welche es aussehen lässt als sei der Player gestorben und in den Himmel gekommen.

Ein weiteres, meiner Meinung nach sehr wichtiges, Component ist das Antialiasing, oder auch die Kantenglättung. Diese lässt zum Beispiel die Holzbalken der Wände und die Steine zum Werfen wesentlich geschmeidiger und realistischer aussehen. Hierbei gibt es auch wieder verschieden Arten der Berechnung, wobei ich bis jetzt noch keinen gravierenden Unterschied zwischen den Arten erkennen konnte.

Danach habe ich noch einen Nebel eingefügt und die Lichtverhältnisse verändert, sodass der Dungeon dunkler und gruseliger aussieht. Dazu habe ich auch noch eine Color Correction eingebaut, welche einzelne Farbtöne unterschiedlich hervorhebt. In meinem Spiel habe ich diese dazu benutzt die Fackeln wärmer und den Nebel kälter aussehen zu lassen, was zu einer meiner Meinung nach passenderen Atmosphäre beiträgt. Theoretisch könnte man diese Atmosphäre auch durch Gamma-Korrektur hervorrufen. Diese lässt einen die Helligkeit einstellen.

Des Weiteren habe ich mich mit verschieden Arten von Kameraführung theoretisch auseinandergesetzt. Praktisch konnte man in diesem Spiel keine umsetzen, da eine First-Person Kamera hier die sinnvollste ist und diese bereits vorliegt.

Als nächstes habe ich drei weitere Kameras eingefügt. Zwei davon bieten die Möglichkeit um die Ecke zu gucken. Eine für eine Ecke links vom Player und eine für eine Ecke rechts vom Player. Die beiden Kameras sind etwas niedriger als die Main Kamera und haben eine kleine Rotation, damit es so aussieht als würde der Player sich neigen, um um die Ecke zu gucken. Die dritte Kamera ist über dem Dungeon und wird dazu genutzt eine Karte darzustellen und diese auf Knopfduck aufzurufen. Zusätzlich ist über dem Player ein Pfeil, der die aktuelle Position anzeigt. Anstatt der Karte hätte ich auch eine Minimap erstellen können, welche im Gegensatz zu der Karte dauerhaft angezeigt werden würde. Ich habe mich jedoch dagegen entschieden, da sonst der Sinn des Labyrinthes verloren gegangen wäre.

Das grafische Aufbereiten des Spiels hat die Performance deutlich gesenkt. Das Spiel hat angefangen stark zu ruckeln und die Eingaben kamen verzögert an. Um dieses Problem zu heben, habe ich das sogenannte Occlusion Culling verwendet. Das Occlusion Culling führt dazu, dass nur die Objekte, welche momentan von der Kamera gesehen werden, gerendert werden. Dies steigert die Performance des Spiels deutlich.

Zudem habe ich mich noch mit den verschiedenen Formen von Skyboxen beschäftigt, welche in unserem Spiel nicht benutzt werden, jedoch bei einem Spiel mit einem Himmel sehr wichtig sind. Hierbei gibt es wieder verschiedene Arten von Skyboxen. Zum Beispiel welche die mit Farben arbeiten und welche die mit mehreren Bildern arbeiten. Die Procedural Skybox, welche mit Farben arbeitet, wird während des Spielens generiert und besitzt die Möglichkeit durch verschiedene Farben eine besondere Atmosphäre, wie zum Beispiel einen Sonnenuntergang, darzustellen. Die 6 Sided Skybox arbeitet mit sechs verschiedenen Bildern, welche passend aneinandergereiht ein großes Bild ergeben. Somit kann man sehr gut einen bewölkten Himmel darstellen. Hier ist jedoch darauf zu achten, dass die Bilder perfekt aneinanderpassen, da sonst Grafikfehler erkennbar sind.

Zum Abschluss jetzt noch ein paar wichtige Dinge, welche beim Thema Kamera zu beachten sind. Man sollte immer daran denken, dass die einzelnen Komponenten von oben nach unten abgearbeitet werden. Sollte man beispielsweise das Tonemapping als erste Komponente haben, so bringt einem das HDR nichts, da es direkt wieder in HDR umgerechnet wird und dann erst die anderen Komponenten, wie Color Correction oder Antialiasing, angewendet werden. Auch ist es gut zu wissen, dass die Skybox immer als erstes gerendert wird.

Nun komme ich zu einem Fazit über die Funktionen und Vielfältigkeit der Kameras in Unity. Kameras sind meiner Meinung nach das wichtigste Objekt im Spiel. Ohne die verschiedenen Grafikverbesserungen würde das Spiel unrealistisch und nicht lebhaft aussehen. Jedoch muss man aufpassen was man an den Grafiken alles verändert, da es sehr schnell passieren kann, dass das Rendering nicht mehr hinterherkommt und es anfängt zu ruckeln.