Die Lösungen für diese Übung sind abzugeben bis Sonntag den 02.06.2023 um 18:00h.

Rasterisierung

Praktischer Aufgabenteil (5 Punkte)

In dieser Aufgabe werden wir einen Software-Rasterisierer entwickeln, indem wir den **inkrementellen** Algorithmus von Pineda aus der Vorlesung implementieren. Dazu brauchen wir den Großteil des Frameworks nicht, sondern wir schreiben mithilfe der stb-Bibliothek direkt in eine Bilddatei `"output.png"`. Das Speichern der Bilddatei ist bereits implementiert, ihr könnt einfach die Funktion `drawDistances` benutzen, um die drei Abstandsfunktionen zu den Kanten des Dreiecks in den Rot-, Grün-, und Blaukanal zu schreiben. Das Dreieck ist durch die drei Eckpunkte in Pixelkoordinaten definiert: `A`, `B`, `C`. Die Normalen für die drei Kanten sind bereits als `N0`, `N1`, `N2` gegeben.

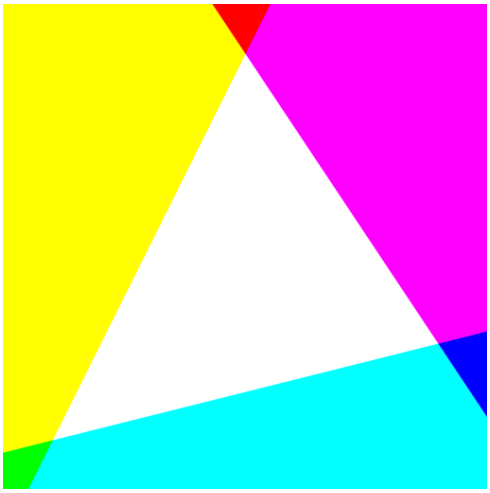

Abbildung 1: So sollte Euer Ergebnis aussehen

Ihr könnt einfach über das gesamte Bild iterieren und müsst keine Bounding-Box berechnen. Zum Füllen könnt ihr abwechselnd von links nach rechts und in der nächsten Zeile von rechts nach links traversieren (siehe linkes Traversionsverfahren).

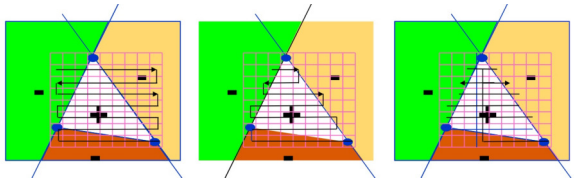

Abbildung 2: Traversionsverfahren

Bonusaufgabe: Deferred Rendering (5 Bonuspunkte)

Anweisungen in `cgintro-bonus-deferred-rendering.pdf`

Theoretischer Aufgabenteil (5 Punkte)

Wendet die **ganzzahlige** Variante des **inkrementellen** Bresenham-Algorithmus an, um eine Linie zwischen den Punkten $(2, 2)$ und $(5, 9)$ zu rasterisieren. Gebt dazu bitte für jeden Iterationsschritt $i \in \{0, \ldots, 7\}$ die Koordinaten x_i, y_i und den Wert der Entscheidungsvariable E_i' inklusive Rechenweg an. Die Punkte liegen dabei immer in der Mitte eines Pixels.