

# **Squad Maprotagenerator 2.0**

**Für politische Entscheidungsträger**

timbow, fletschoa, kappakay

28. September 2022

# Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>4</b>
<b>1 Einleitung</b>	<b>5</b>
1.1 Problemdarstellung . . . . .	5
1.2 Ziel . . . . .	5
<b>2 Aufbau</b>	<b>7</b>
2.1 Grundlegende Struktur . . . . .	7
2.2 Aufbau im Detail - Mode . . . . .	8
2.3 Aufbau im Detail - Map . . . . .	9
2.3.1 Distanzweight . . . . .	9
2.3.2 Mapvoteweight . . . . .	9
2.4 Aufbau im Detail - Layer . . . . .	10
<b>3 Mapweights</b>	<b>11</b>
<b>4 Optimizer</b>	<b>12</b>
4.1 Funktionsweise . . . . .	12
4.2 Anwendung . . . . .	12
<b>5 Ergebnisse</b>	<b>13</b>
5.1 Bewertung des Systems . . . . .	13
5.1.1 Mapverteilung . . . . .	13
5.1.2 Mode/Modus verteilung . . . . .	15
5.1.3 Varriation der Maps . . . . .	15
5.1.4 Map Wiederholung . . . . .	15
5.2 Grenzen des Systems . . . . .	16
<b>6 Ein Blick in die Glaskugel</b>	<b>18</b>
6.1 Diskussion . . . . .	18
6.2 Ausblick . . . . .	18

Memory Colonel, der du bist in GooseBay, geheiligt werde dein  
Name.  
Deine Rota komme.  
Deine Locktime geschehe.  
Unser täglich Squad gib und heute.  
Und vergib uns unser Minen legen, wie auch wir vergeben unseren  
Snipern  
Und führe uns nicht in Versuchung, sondern erlöse uns von Tällil  
Denn dein ist die Rota und Biome und die Abwechslung in Ewigkeit.  
Amen

- *Das Maprota Gebet*

## Abbildungsverzeichnis

1	Oberflächlicher Ablauf des Generators. . . . .	7
2	Die Sigmoidfunktion für mehrere Werte $a$ und $b$ . Für $b = 0$ ist für alle Werte $a$ $f(0) = 1/2$ . Der Parameter $b$ dient als Offset. . . . .	10
3	Cluster . . . . .	10
4	erwartete Mapverteilung im Modus RAAS nach den Layervotes vom 19.09.2022	14
5	generierte Mapverteilung im Modus RAAS nach den Layervotes vom 19.09.2022 (1.Mio. Layer Rota) . . . . .	14
6	Häufigkeiten des gleitenden Mittelwertes der Distanzen . . . . .	15
7	Häufigkeiten der Map Wiederholung . . . . .	16

# 1 Einleitung

Dem gemeine Squad Spieler mag nach der Sammlung an einiger Erfahrung gemerkt haben, dass eine abwechslungsreiche Rotation von Layern (Maprota) die Qualität des Spieles deutlich verbessert. So viel uns in der Vergangenheit Zeit auf, dass auf dem WLS Discord sich immer wieder Spieler über die Maprota beschwert haben. Anfangs beschwerten wir uns auch, bis wir uns das Ziel gesetzt haben ein Programm für die Generierung einer besseren Maprota zu schreiben, bevor wir uns wieder beschwerten. Die Ziele, für das Projekt, waren schnell aufgestellt und das Problem scheint im ersten Moment einfach lösbar zu sein. Wir können euch nach 2 Monaten Entwicklungszeit aber sagen es ist definitiv nicht einfach! Es wäre einfacher gewesen sich weiterhin zu beschweren. Die Lösung die wir für eine bessere Maprota gefunden haben kannst Du hier nachlesen.

## 1.1 Problemdarstellung

Aus den historischen Debatten über die Maprota wurden die Hauptprobleme versucht herauszuheben und im folgenden erklärt.

Der Mapgenerator soll qualitativ hochwertige Rotas generieren. Zur klassifizierung werden wir nun zunächst Eckpunkte definieren welche die Qualität einer Rota bemessen sollen.

Zunächst sollte sich eine Map nicht zu stark wiederholen. Desweiteren gab es in der Community bedenken welche sich damit beschäftigen dass nicht zu ähnliche Maps zu kurz hintereinander gespielt werden sollten. Ein Beispiel für letzteres wäre die Abfolge

Sumari → Logar Valley → Fallujah.

Hier würden direkt nacheinander folgend drei relativ ähnliche Maps gezogen werden. Der Charakter dieser Maps wird im wesentlichen über die Eigenschaften „Wüste“, „Stadt“, „Infanterie lastig/klein“ der Map definiert. Eine gute Rota sollte solche Map-Ketten vermeiden.

Ein weitere wichtiger Punkt ist die Vermeidung von Mustern in der Rota. Das bedeutet, dass die generierten Rotas nicht deterministisch verteilt, sondern aus zufälligem Ziehen entstehen sollen.

Das seit einiger Zeit bestehende Layervote-System muss direkten Einfluss auf die Rota haben um die Verteilung den Wünschen der Community anzupassen.

## 1.2 Ziel

Zusammenfassend werden wir im folgenden die Qualität der Maprota an folgenden Eigenschaften messen:

- Ähnlichkeit der gezogenen Maps in kurzem Zeitraum
- Wiederholung der selben Map in kurzem Zeitraum

- Keine Muster/Nicht-deterministische Verteilung
- Layervotes müssen direkten Einfluss haben

Der in diesem Dokument beschriebene Algorithmus soll alle vier genannten Eigenschaften so gut wie möglich Erfüllen. Kurz gesagt ist das Ziel des Maprota Generators: „Ein probabilistisches System dessen globale Verteilung durch Mapvotes gegeben ist und lokal Wiederholung ähnlicher Maps vermeidet.“ Anders ausgedrückt:

Das System sollte global einer Verteilung folgen welche die Map/Layervotes widerspiegeln und lokal eine gewisse Varianz unter den Mapcharakteristiken hervorruft.

Im weiteren Verlauf wird ein Algorithmus präsentiert, welcher alle oben genannten Aspekte so gut wie möglich abdeckt. Es sei aber darauf hingewiesen, dass nicht alle Punkte gleichzeitig perfekt umgesetzt werden können aufgrund kritischer Eigenschaften des Systems und der gesetzten Nebenbedingungen.

## 2 Aufbau

Im Folgenden wird der Aufbau des Algorithmus näher beschrieben. Nach einem kurzem Überblick werden die einzelnen Subsysteme detailliert erklärt.

### 2.1 Grundlegende Struktur

Es soll eine Liste an Layern, anhand der zuvor definierten Ziele, generiert werden. Die oberflächliche Struktur, des Generators, ist im unteren Flussdiagramm dargestellt. Für eine gegebene Anzahl an Layern in einer Rota wird für jedes Layer folgende Routine ausgeführt:

- wähle einen Modus, gewichtet nach den Mode-Wahrscheinlichkeiten  $w_g$  und dem Mode-Spacing  $\Delta_g$
- für den gewählten Modus werden die validen Maps gefiltert
  - zunächst wird geprüft, welche Map im Modus repräsentiert ist
  - aus den vorhandenen Maps wird nach dem Distanz-Vote-Weight  $w_m$  gewichtet eine Map gezogen
- für die gezogene Map wird gewichtet nach dem Layerweight  $w_l$  ein Layer gezogen
- die gezogenen Maps verbleiben für eine feste Maplänge im Memory-Kernel und sind damit für die nächsten Map Ziehungen nicht verfügbar

Im Folgenden werden die drei verschiedenen Schritte genauer erklärt und die verwendeten Weights definiert.

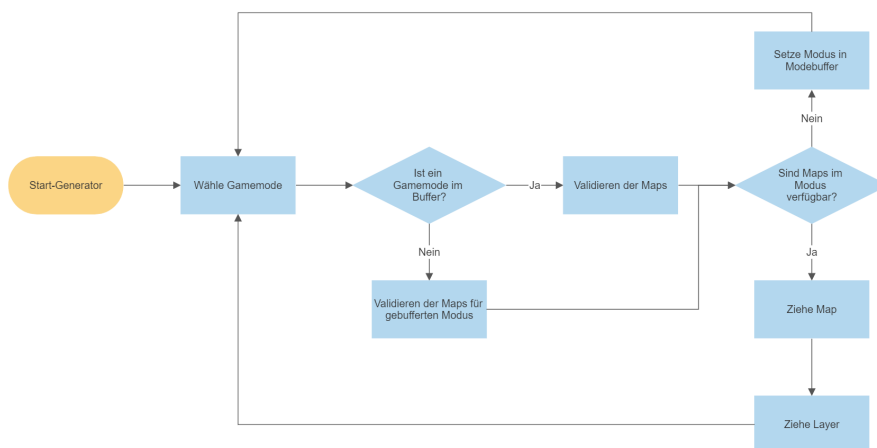


Abbildung 1: Oberflächlicher Ablauf des Generators.

## 2.2 Aufbau im Detail - Mode

Wie zuvor erwähnt gibt es zwei Faktoren, welche die Ziehung des Modus beeinflussen:

1. die Modeweights die zuvor gesetzt wurden  $w_g$
2. das Modespacing  $\Delta_g$

Des Weiteren erlaubt der Generator eine Gruppierung der Modes in sogenannte "Pools". Es gibt immer einen sogenannten *main*-Pool. Ohne weitere Einstellungen ist jeder Modus, der gespielt werden soll, automatisch im Main-Pool enthalten. Neben diesem können noch weitere Pools definiert werden. In der momentanen Fassung existieren drei Pools:

- **main:** Der zuvor erwähnte Standard-Pool, beinhaltet *RAAS* und *AAS*
- **intermediates:** Beinhaltet die Modi *Invasion* und *TC*
- **reste:** Beinhaltet *Destruction* und *Insurgency*

Das **Modespacing**  $\Delta_g \in 0, \dots, N$  ist eine Zahl größer 0 und maximal die Anzahl der Layer in einer Rota. Es sorgt nun dafür, dass für eine gegebene Anzahl an Runden nur der Main-Pool gezogen werden darf, wodurch eine „Mindestzeit“ zwischen z.B. zwei Invasion Layern garantiert wird. Sollte die Zeitspanne seit dem letzten nicht-main Modus größer sein als das Modespacing, so wird mit den Pool und Mode-weights gewichtet ein Gamemode gewählt. Hierbei wird erst der Pool ausgewählt und anschließend im Pool der Modus. Es ist zu beachten, dass dies dazu führen kann und wird, dass nicht direkt nach Ablauf des Modespacings ein andere Pool drankommen muss. Das Design wurde mit Absicht so gewählt, um repetitive Modes zu verhindern.

Das Modeweight  $w_g$  setzt sich damit zusammen aus der Wahrscheinlichkeit den enthaltenen Pool zu ziehen und den Modus

$$w_g = \mathbb{P}(G = g | P = p) = \mathbb{P}(G = g) \mathbb{P}(P = p) \quad (1)$$

wobei hier  $G$  und  $P$  die Zufallsvariablen "Gamemode" und "Pool" darstellen sollen. Durch das Modespacing ist die Wahrscheinlichkeit einen Modus zu ziehen gegeben durch

$$\mathbb{P}(G = g) = \frac{1}{N} \sum_{j=0}^{k_0} j \binom{N - j\Delta_g}{j} w_g^j (1 - w_g)^{N - j(\Delta_g + 1)} \quad (2)$$

mit  $w_g$  das oben definierte Weight,  $N$  die Anzahl gezogener Layer und  $k_0 = \lfloor \frac{N}{N_0} \rfloor$  die maximale Anzahl an Zügen des Modus. Der interessierte Leser vermag eine Herleitung im Anhang zu finden.

Anschließend wird geprüft ob die Verfügbaren Maps den gewählten Modus enthalten. Sollte dies nicht der Fall sein so wird der Modus in einem Buffer gespeichert und es wird erneute ein Modus zufällig gewählt. Der Buffer wird bei der nächsten Gelegenheit abgearbeitet.



## 2.3 Aufbau im Detail - Map

Das Weight errechnet sich als Produkt aus einem "Distanz-Weight" und einem "Mapvote-Weight".

$$w_m(m, d, v) = \frac{1}{\mathcal{N}} w_d(d, m) w_v(v, m) \quad (3)$$

wobei  $\mathcal{N}$  das Produkt-Weight normiert sodass  $\sum_m w_m = 1$ .

### 2.3.1 Distanzweight

Das Distanzweight ist eine allgemeine stückweise stetige Funktion definiert durch

$$w_d : \mathbb{R}^+ \rightarrow \mathbb{R}^+, d \mapsto w_d(d), \quad (4)$$

und der Nebenbedingung  $w_d(d) \xrightarrow{d \rightarrow 0} 0$ . In der momentanen Version ist die Funktion gegeben durch

$$w_d(d) = 1_{[0, d_{\min}]}(d) \quad (5)$$

mit  $d_{\min} \geq 0$  als Minstdistanz. Sollte eine Map näher als  $d_{\min}$  an einer zuvor gezogenen Map liegen ist das Distanzweight und damit das Mapweight 0 und wird damit nicht gezogen.

### 2.3.2 Mapvoteweight

Das Mapvoteweight wird aus dem Mapvote berechnet aus

$$w_v(v) = \sum_{i,j=0}^2 a_{ij} x^i y^j \quad (6)$$

wobei  $x$  die Anzahl an verbundenen Maps ist und  $y$  die Mapwahrscheinlichkeit errechnet aus den Layervotes. Die Koeffizienten  $a_{ij} \in \mathbb{R}$  müssen vorerst numerisch bestimmt werden. wobei  $v$  die Summe aller Layervotes eines Modus einer Maps ist und  $a$  und  $b$  zwei freie Parameter sind. Während  $a$  die Steigung moduliert kann mit  $b$  ein Offset erzeugt werden. Die Sigmoid funktion erlaubt es ein kontinuierliches weight zu definieren wodurch eine Map oder ein Layer nicht abrupt verschwindet sondern langsam ändert. Jeder Map die im Modus enthalten ist wird ein Mapvoteweight  $w_m$  und ein Distanzweight  $w_d$  zugeordnet. Aus dem Produktweight wird anschließend die Map gezogen. Hier kommt noch hinzu dass die Distanzweights vom Memory-Kernel abhängen. In der momentanen Implementierung bedeutet dies, dass eine Map, welche in den letzten  $k$  Runden gezogen wurde, nicht nochmal dran kommen kann. Effektiv heißt dies  $w_d = 0$  für diese Map. Für den seltenen Fall, dass keine Map für einen Modus verfügbar ist, weil alle in Frage kommenden Maps zurzeit gesperrt sind wird der Modus in den sogenannten **Mode-Buffer** verlegt. Dieser wird beim Ziehen des nächsten Modus abgearbeitet. Das bedeutet ein Modus wird „ge-queued“ und kommt bei der nächst passenden Gelegenheit dran.

timbow  
schreib  
mal den  
satz um

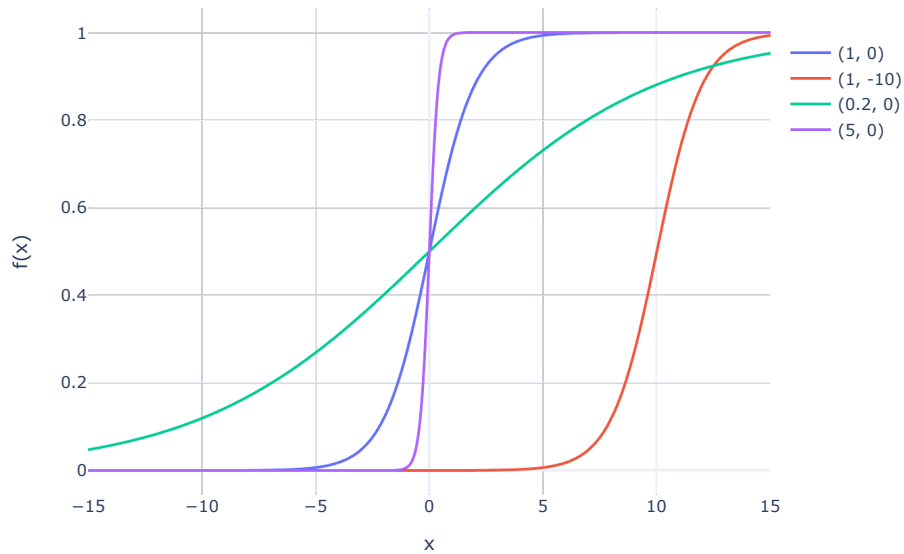


Abbildung 2: Die Sigmoidfunktion für mehrere Werte  $a$  und  $b$ . Für  $b = 0$  ist für alle Werte  $a$   $f(0) = 1/2$ . Der Parameter  $b$  dient als Offset.

## 2.4 Aufbau im Detail - Layer

Sobald eine Map ausgewählt wurde wird das Layer gewichtet nach dem Layerweights  $w_l$  gezogen. Diese hängen nur von den Mapvotes ab welche mit der vorher genannten Sigmoidfunktion moduliert werden. Damit ist dann für  $N_l$  Layer einer Map

$$w_l(j) = \frac{1}{\sum_{m=0}^{N_l} \frac{1}{\sum_{m=0}^{N_l} [1 + \exp(-a(v_m + b))]}} \frac{1}{1 + \exp(-a(v_j + b))} \quad (7)$$

mit  $v_j$  die Mapvote-Zahl des Layers  $j$  der gegebenen Map.

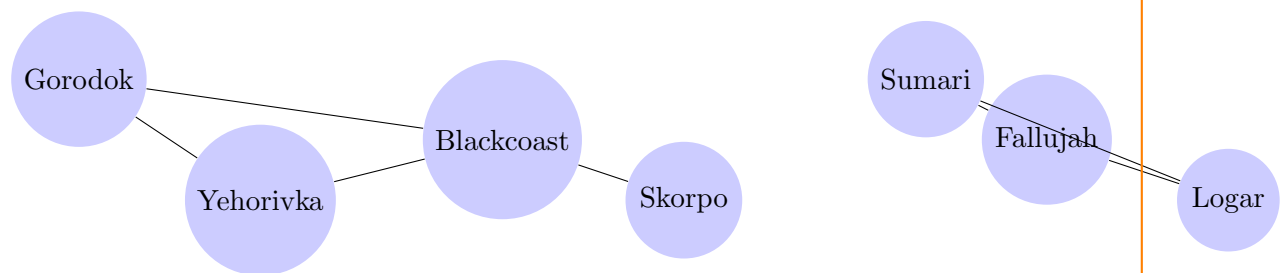


Abbildung 3: Cluster

cluster ab-  
bildung  
wurde  
nicht in  
den Text  
einge-  
bunden,  
überflüssig  
?

### 3 Mapweights

## 4 Optimizer

Der Optimizer ist ein entstandenes U-Boot Projekt, welches gute Parameter für die Mapweightgleichungen findet. Die Bestimmung optimaler Parameter, sind entscheidend für eine gute Annäherung der generierten Mapverteilung an die Erwartete. Wahrscheinlich wäre es möglich die Parameter analytisch zu ermitteln. Da dieses aber sehr zeitintensiv sein würde, haben wir uns vorübergehend für eine Lösung mit einem einfachen „Machine Learning“ Algorithmus entschieden.

wo Gleichung für ref

ref

### 4.1 Funktionsweise

Der Optimizer sucht nach der kleinsten Abweichung zwischen der vorgegebenen und generierten Mapverteilung. Es wird jeweils nur ein Modus betrachtet und optimiert.

Der erste Schritt des Optimizers ist die Variierung des ersten Parameters um ein  $\Delta p$  der Mapweightgleichung. Anhand der neuen Mapweightgleichung werden neue Mapweights berechnet und eine Maprotas generiert. Die Maprotas haben eine ausreichende Größe, dass Streuungseffekte nahezu nicht mehr vorhanden sind. Nun wird die Abweichung  $s_m$  berechnet:

$$s_m = \sqrt{(d_e[0] - d_g[0])^2 + \dots + (d_e[n] - d_g[n])^2} \quad (8)$$

das hier ist falsch wie heißt das richtig?

mit  $n \in \mathbb{Z}$  wobei  $d_e$  die diskrete Verteilung der erwarteten Maps sind und  $d_g$  die generierte diskrete Verteilung ist. Im jetzigen Zeitschritt  $k$  wird die Abweichung  $s_m[k]$  mit der Abweichung  $s_m[k-1]$  verglichen. Bei dem ersten Durchlauf entspricht  $s_m[k-1]$  der Abweichung der Grundverteilung. Sollte die Abweichung  $s_m[k]$  kleiner sein als  $s_m[k-1]$  bleibt die Variierung um das  $\Delta p$  bestehen. Es wird zum nächsten Parameter gewechselt und es wird wieder variiert. Nachdem ein Minimum mit dem  $\Delta p$  gefunden wurde, wird  $\Delta p$  reduziert und die Suche startet neu, bis ein vorgegebenes Minimum für  $\Delta p$  erreicht ist.

tim mach mal mathe in schön bitte

### 4.2 Anwendung

Das Finden neuer Parameter durch den Optimizer, wird nur bei veränderten Layervotes und veränderten Einstellungen benötigt. Daher wird der Optimizer auch nur bei solchen Veränderungen vor der Generierung automatisch aufgerufen. Für eine geringere Laufzeit wird der Optimizer parallel mit jedem Modus ausgeführt.

## 5 Ergebnisse

### 5.1 Bewertung des Systems

Um den entstandenen Maprotagenerator bewerten zu können und den Grad der Qualität feststellen zu können, werden die Metriken aus Kapitel ?? zur Hilfe genommen. Die Metriken sind für Squadmaprotas allgemeingültig und anhand dessen könnten sie miteinander verglichen werden. Es soll nicht unerwähnt bleiben, dass durch eine schlechte oder auch falsche Wahl der Einstellparameter das Maprotasystem leicht bis hin zu sehr stark beeinträchtigen werden kann. Genauer dazu ist unter 5.2 nachzulesen. Daher ist bei diese Bewertung zu berücksichtigen, dass von uns wohl überlegte Einstellparameter festgelegt wurden und als Referenz für Änderungen herangezogen werden sollten. Das Wählen passender Einstellparameter kann im User manual nachgelesen werden.

ref

Es folgt die Auswertung der Maprota anhand der vorgegebenen Werten.

#### 5.1.1 Mapverteilung

Die Mapverteilung wird von den Layervotes beeinflusst, dieses ist in Kapitel ?? nachzulesen. Diese Verteilung ist die Vorgabe für das System und dieses versucht es optimal anzunähern. Da durch die Zielvorgaben die Verteilungsvorgabe nicht immer erreicht werden kann, tritt ein Abweichung in der Verteilung auf. Diese Abweichung wird hier als mittlere quadratische Abweichung (MSD) pro Modus angegeben. Für diese Auswertung wurden den Verteilungen genommen, die aus den Layervotes vom 19.09.2022 entstanden sind. Dabei ist zu beachten, dass der Modus TC zu diesem Zeitpunkt „Verbugged“ ist und daher in der Tabelle 1 nicht auftaucht.

ref

ac?

Modus	MSD
RAAS	0.00192
AAS	0.00090
Invasion	0.00336
Insurgency	0.00836
Destruction	0.04831

Tabelle 1: mittlere quadratische Abweichung Mapverteilung

Um eine Vorstellung zu Entwickeln wird im Folgenden die angestrebte und generierte Verteilung als Diagramm dargestellt (siehe Abbildung 4 und Abbildung 5).

Bei der Betrachtung der Diagramme (Abbildung 4 und Abbildung 5) ist zu beachten, dass es sich hier nur um den Modus RAAS handelt. Beispielsweise ist die Karte Al-Bashrah hier deutlich unterrepräsentiert diese ist im Modus Invasion nicht der Fall, da die Layervotes dort für AlBashrah deutlich besser ausfallen. Zudem lässt sich erkennen, dass die Karte Yehorivka, Blackcoast und Gorodok nicht den angestrebten Verteilung erreichen. Dieses Phänomen 5.2 näher behandelt.

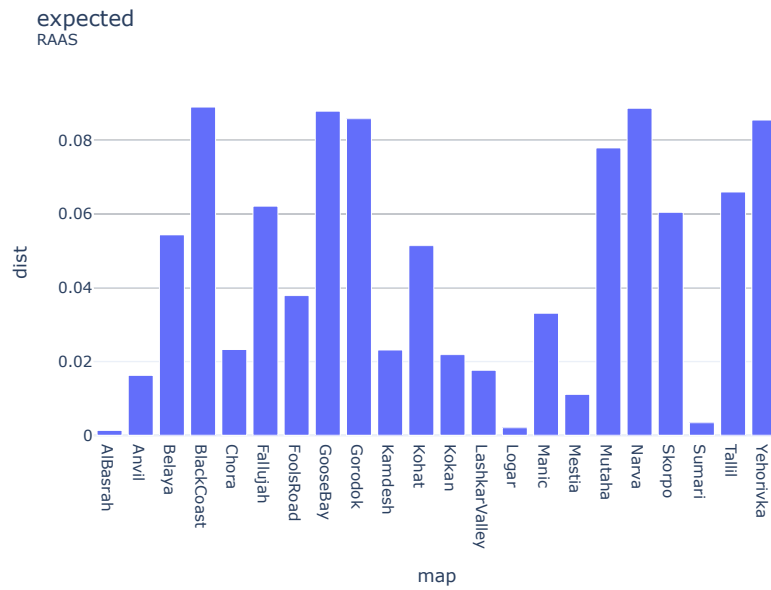


Abbildung 4: erwartete Mapverteilung im Modus RAAS nach den Layervotes vom 19.09.2022

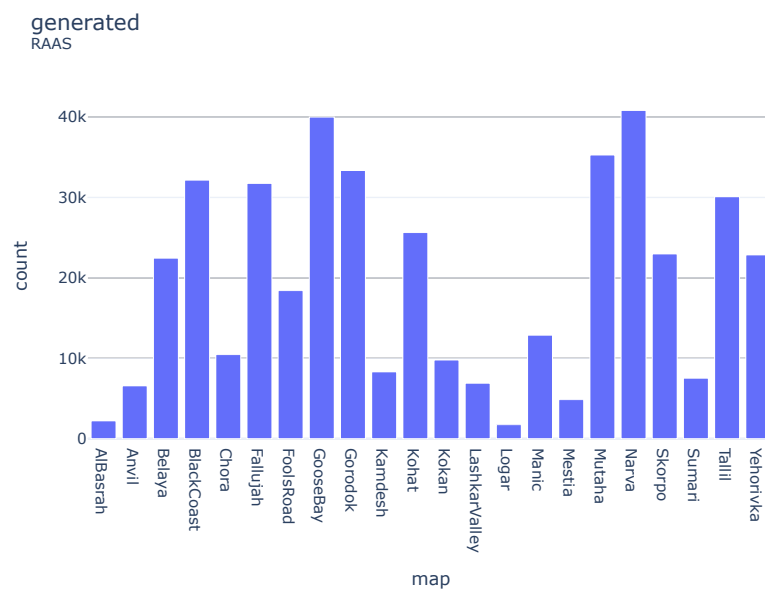


Abbildung 5: generierte Mapverteilung im Modus RAAS nach den Layervotes vom 19.09.2022 (1.Mio. Layer Rota)

### 5.1.2 Mode/Modus verteilung

Wie bei den Mapverteilungen kann bei den Modusverteilungen die mittlere quadratische Abweichung als quantifizierendes Mitteln genommen werden. Bei den Modi ergibt sich eine  $MSD = 0.04514$ . Dieser Wert ist für die vorgesehenen Einstellparameter akzeptabel, da Modi die nicht RAAS oder AAS sind einen Mindestabstand haben. In diesem Falle ist dieser Abstand 4 Runden.

### 5.1.3 Variation der Maps

Für den die Messbarkeit, der Differenz aufeinander folgende Maps, kann das arithmetische Mittel der Distanzen auf der Hyperfläche genutzt werden. Zudem ist es noch sinnvoll sich den gleitenden Mittelwert der Distanzen zu betrachten.

Das arithmetische Mittel der Distanzen beträgt  $d_m = 1.08946$

Die Betrachtung des gleitenden Mittelwertes ergibt sich für eine Mittelwertbreite von 5 und einer Rota mit 100000 Layern eine Verteilung die auf Abbildung 6 zu sehen ist.

Bewertung  
des wertes

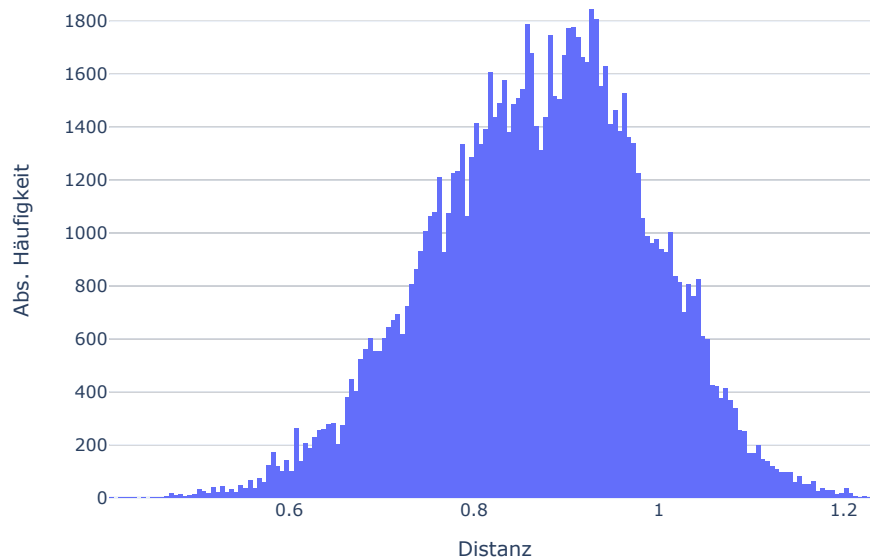


Abbildung 6: Häufigkeiten des gleitenden Mittelwertes der Distanzen

bewertung  
des histo-  
grams

### 5.1.4 Map Wiederholung

Das nächste und hier letzte benutzte Mittel, um eine Maprota zu bewerten ist, nach wie vielen Runden sich eine Map wiederholt. Hierfür wurde eine Histogramm aus einer

100000 Layer Rota erstellt. Die Abbildung 7 zeigt die Häufigkeit einer Map Wiederholung. Es ergibt sich ein Minimum von 3 Runden bevor sich eine Map wiederholen kann. Am Häufigsten ist jedoch eine Map Wiederholung nach 6 bis 9 Runden. Dabei muss bedacht werden, das Squad aktuell (09.2022) spielbare Maps beinhaltet. Daher ist dieses ein gutes Wiederholverhalten.

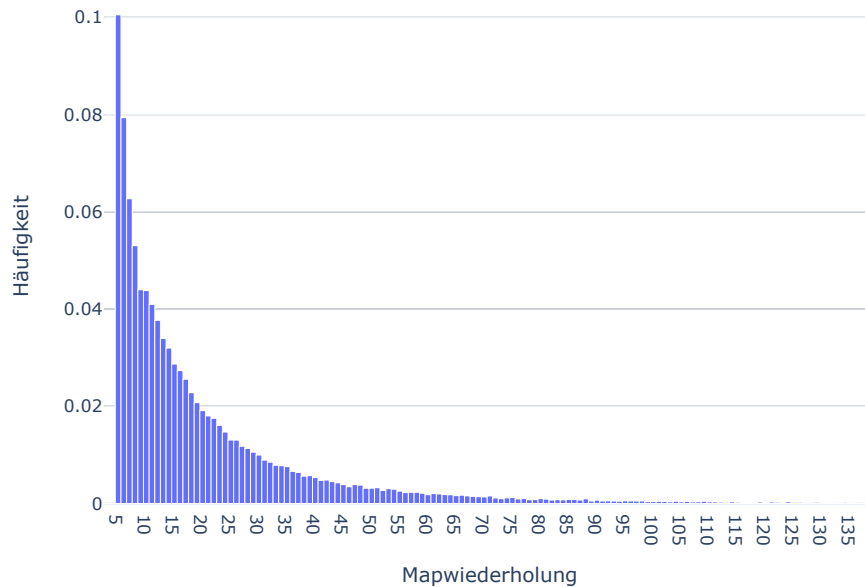


Abbildung 7: Häufigkeiten der Map Wiederholung

## 5.2 Grenzen des Systems

Um dieses Sektion am besten nachzuvollziehen zu können, sollte nochmal ein Blick auf die Ziele des Systems geworfen werden. Es wird eine qualitativ hohe Maprota gefordert, die zum einen der Voteverteilung folgen soll und zum anderen in der Map-Reihenfolge einige gewisse Diversität garantieren soll. Bei genauere Überlegung ist das schon ein Widerspruch in sich. Angenommen eine einzelne Map hat unendlich viele Stimmen und die Maprota folgt strikt den Votes. Es würde Resultieren, dass nur noch die eine Map drankommen würde. Diese, von der Maprota, angenommene Verteilung bildet aber einen Konflikt mit dem Ziel, dass die Maps eine gewisse Diversität bieten sollen. Daher sind an der Stelle die Möglichkeiten das Systems beschränkt und die Voteverteilung kann nicht immer in einer generierten Rota abgebildet werden. Sehr stark hoch gewählte Maps können nur so oft drankommen, wie es die Clusterstruktur und die Locktime zulässt. Dieser Aspekt des Systems muss aber nicht als negativer Punkt aufgefasst werden, denn keiner will immer nur eine Map spielen (solange es nicht GooseBay ist). Dieses „Feature“




wirkt damit aktiv gegen die Befürchtung, welche im Feedback-chat angesprochen wurde, dass nur noch Yehorivka und Gorodok drankommen. Um trotzdem das Optimum zwischen vorgegebener Verteilung und Diversität der Maps zu garantieren wird ein Optimizer eingesetzt.

## **6 Ein Blick in die Glaskugel**

### **6.1 Diskussion**

### **6.2 Ausblick**

## Anhang

An orange horizontal line spans the width of the page, ending in a small L-shaped bracket that points to an orange rounded rectangle containing text.

hier mem  
colonel  
einfügen