1. Beschreibung der Spielidee

Die grundlegende Spielidee ist, dass zwei Clients bzw. deren KIs, *ohne menschliches Eingreifen*, auf der gleichen Spielkarte eine vergleichbare Aufgabe erfüllen müssen. Die KI, die diese schneller erfüllt, gewinnt das Spiel. Dabei werden die KIs von einem Server unterstützt der beide Clients koordiniert, Datenaustausch ermöglich, als Schiedsrichter auftritt und für Clients Daten speichert, aktualisiert und auswertet.
Die *Grobarchitektur* ist damit als klassische Client/Server Architektur vorgegeben.

Initial gilt es am Server ein neues Spiel anzufordern. Dieser erste Schritt wird noch von einem Menschen durchgeführt, alles danach erfolgt immer vollautomatisch durch eine Client/KI-Implementierung. Nach Start des Clients registrieren sich die KIs für das Spiel am Server und erstellen/tauschen danach mit dem Server Kartenhälften aus. Die Karte, auf welcher gespielt wird, ist hierbei nicht fest vorgegeben, sondern entsteht durch die Kombination dieser zufällig erzeugten Kartenhälften durch den Server.

Nach dem Kartenhälftenaustausch starten die Spielfiguren jeder KI jeweils auf einer selbst ausgewählten Position (ihrer Burg) auf der von ihr erstellten Kartenhälfte. Anschließend muss jede KI die Karte erkunden um schnellstmöglich ihren Schatz und danach die gegnerische Burg zu finden. Hierzu wird je ein Schatz vom Server auf jeder der beiden Kartenhälften versteckt. Der Schatz einer KI ist immer auf der Kartenhälfte zu finden auf welcher die KI startet. Ausserdem kann er nur von dieser aufgenommen und gesehen werden (**kein "Schatzdiebstahl")**.

Um den Schatz zu finden, bewegen beide KIs ihre Spielfigur über die Karte und decken dabei mit ihrer Spielfigur *schrittweise Kartenfelder auf*. Dieses "aufdecken" geschieht indem der Server die Position der KIs *auswertet* und die Spieldaten (z.B. was gesehen, aufgenommen, gewonnen/verloren) für die Clients passend aktualisiert. Ziel ist es dabei möglichst schnell den versteckten eigenen Schatz und *danach* die gegnerische Burg zu finden. Die KI entscheidet selbst, ohne menschliches Zutun, wie hierzu am Besten vorzugehen ist. Je geschickter *Algorithmen*, *Spielregeln* und eigene *Ideen* während der KI-Entwicklung ausgenützt werden desto schneller wird diese dabei sein.

Bereits direkt nach dem Kartenaustausch sind für die KIs *alle Terrains* aller Kartenfelder, alle Spielfigurpositionen und die *eigene Burg* sichtbar bzw. auf der vom Server abfragbaren Karte vermerkt. Andere Details wie der eigene Schatz oder die gegnerische Burg sind "*versteckt*" und müssen erst *gefunden* (aufgedeckt/gesehen) werden. Die Position der Gegnerspielfigur ist in den ersten Spielrunden *zufällig*, später entspricht diese der *echten* Gegnerposition.

Zufällige Position des Gegners: *Bis zum Abschluss der ersten 8 eigenen Clientaktionen hat die vom Server gemeldete Position des Gegners zufällig zu sein. Dies kann zu unerwarteten Positionen und vermeintlichen Sprüngen zwischen Feldern für den Gegner führen. Zeigen Sie diese trotzdem einfach im Client-UI an. Warum? Dies verhindert, dass die Position der gegnerischen Burg über die Spielerposition direkt verraten wird. Gleichzeitig kann der Client-Code immer davon ausgehen, dass Informationen über alle Spieler vorhanden sind. So ist keine Fallunterscheidung dafür im Client notwendig.*

Sobald eine KI "ihren" Schatz gefunden bzw. aufgedeckt hat, muss sie ihre Spielfigur zu diesem bewegen, um ihn *aufnehmen* zu können. Danach, sobald der Schatz aufgenommen wurde, muss die KI mit ihrer Spielfigur die gegnerische Burg finden. Sobald diese gefunden wurde, muss die KI sich zu dieser bewegen, die Bewacher der Burg mit dem gerade gefundenen Schatz bestechen und so die Burg "übernehmen". Hat die KI dies geschafft, gewinnt diese und damit auch der menschliche Spieler, der diese KI erstellt hat.

Das Aufnehmen eines Schatzes und die Übernahme der Burg erfolgen automatisch durch den Server, sobald eine Spielfigur das jeweilige Spielfeld der Karte betritt. Im Falle des Schatzes wird hierbei der gerade aufgenommene Schatz direkt auch von der Karte entfernt (dieser ist jetzt im vom Server verwalteten "Inventar" der KI).

Die Spielaktionen selbst werden rundenbasiert durchgeführt. Jede KI kann immer nur eine Aktion setzen (z.B. einen Bewegungsbefehl oder die Übertragung einer Kartenhälfte) und muss danach warten, bis die andere KI ihre Aktion gesetzt hat. Eine KI kann hierbei nicht auf das Setzen einer Aktion verzichten, sondern muss immer eine Aktion durchführen. Eine KI darf aber keine Aktion setzen solange die andere KI an der Reihe ist. Der Server unterstützt dies, können Clients doch abfragen ob diese gerade an der Reihe sind. Bestraft (indem betroffene Clients verlieren) aber auch Clients die diese und andere Spielregeln nicht einhalten.

Um die Spiele für die Zuschauer spannend zu gestalten, wurde festgelegt, dass ein Spiel insgesamt nicht länger als **320 Spielaktionen**

(und damit 320 Runden) dauern darf. Für **jede dieser rundenbasierten Spielaktion hat die KI maximal 5 Sekunden** Bedenkzeit. Insgesamt dauert ein Spiel maximal 10 Minuten. Sollten diese Bedingungen nicht erfüllt werden, verliert die KI, welche gerade an der Reihe ist, das Spiel automatisch und der zugehörige menschliche Spieler bekommt dies vom Client mitgeteilt. Der andere menschliche Spieler wird von seinem Client über seinen Sieg informiert. Das gleiche automatische Vorgehen wird auch angewendet, wenn auf normalem Weg gewonnen\verloren wird. Daher wenn eine KI mit ihrem Schatz als Erster die gegnerische Burg erreicht. Im Anschluss terminiert sich der Client immer selbstständig und sendet keine Aktionen (Kartenhäfte, Bewegungen) mehr an den Server.

Die von den KIs bespielte Karte wird bei Beginn des Spiels von beiden beteiligten künstlichen Intelligenzen kooperativ erstellt. Hierzu erstellt jede der beiden KIs zufällig eine Hälfte der finalen Spielkarte (mit je 5 x 10 Feldern). Die Kartenhälften werden vom Server *zufällig* entweder an den kurzen (Kartenabmessungen 5 x 20) oder den langen Seiten (Kartenabmessungen 10 x 10) zusammengefügt. Wie dies geschieht können die KIs nicht beeinflussen und können/müssen daher vom Server den Zustand der gesamten Karte abfragen. Die Karten sind in Felder aufgeteilt, zwischen denen sich die Spielfiguren, auf Anweisung der KIs, schrittweise waagrecht und senkrecht bewegen. Jedes Feld repräsentiert genau eine von drei möglichen Terrainarten: Wasser, Wiese oder Berg.

Burgen und Schätze werden ebenfalls auf eines der Felder gesetzt. Allerdings können die **Burg (vom *Client*) und der Schatz (vom *Server*) nur auf Wiesenfeldern und nicht auf demselben Feld platziert werden**. Außerdem darf **Wasser unter keinen Umständen betreten** werden (Spielfiguren können nicht schwimmen). Bewegt sich die Spielfigur einer KI in Richtung eines Wasserfeldes, verliert die KI automatisch. Dies passiert auch wenn die Spielfigur von der **Karte flüchten** würde (z.B., wenn sich die Spielfigur bereits am linken Rand der Karte befindet und sich trotzdem weiter nach links bewegt).

Wasser und Wiesen können jeweils mit einer Bewegungsaktion betreten und im Fall von Wiesen auch wieder verlassen werden. Sobald eine Wiese betreten wurde, wird aufgedeckt, ob sich auf dieser Wiese eine Burg oder ein Schatz befindet bzw. wird letzteres direkt aufgenommen. Im Gegensatz zu Wiesen benötigen Berge zwei

Bewegungsaktionen, um diese (das Bergfeld) zu betreten und zwei zusätzliche, um den Berg wieder zu verlassen. Dafür werden beim Betreten des Bergfeldes versteckte Schätze und gegnerische Burgen in bis **zu einem Feld Entfernung** (auch diagonal) rund um das Bergfeld für die KI vom Server aufgedeckt. Eine KI verbleibt in der Visualisierung sowie den vom Server abfragbaren Positionsdaten so lange am ursprünglichen Spielfeld, bis diese ausreichend Bewegungen an den Server gesendet hat um das aktuelle Feld zu verlassen und das neue Feld zu betreten.

Im Abschnitt Detaillierte Beschreibung der Bewegungen werden die Bewegungsaktion, Schritte, Positionen und Runden für verschiedene Terrainkombinationen erläutert.
Es kann jederzeit eine neue Bewegungsrichtung eingeschlagen werden (z.B. rechts statt links). Die Anzahl der notwendigen Bewegungsbefehle wird dann neu bestimmt und gezählt. Bereits an den Server gesandte Bewegungsbefehle verfallen, die dafür aufgewendeten Spielrunden zählen trotzdem als verbraucht. Wurden beispielsweise drei Bewegungsbefehle in eine Richtung geschickt (um zwischen zwei Bergen zu wechseln) und danach erfolgt ein Bewegungsbefehl in eine andere Richtung verfallen die drei bereits gesendeten Bewegungsbefehle. Selbst wenn danach wieder ein Bewegungsbefehl in die alte zuvor verwendete Richtung versendet werden würde.

Eine Spielfigur kann sich nur horizontal und vertikal zu direkt benachbarten Feldern bewegen, das Überspringen von Feldern ist nicht möglich.

Die KIs müssten während der Erstellung der Kartenhälften eine Reihe von Regeln einhalten:

- Kartenhälften müssen *zufällig* mit *Algorithmen* generiert und nicht statisch vorgegeben werden.
- Jede Kartenhälfte muss mindestens 10% Bergfelder, 48% Wiesenfelder, 14% Wasserfelder und 2% Burg beinhalten.
- Es muss möglich sein von jedem Kartenfeld jedes andere Kartenfeld, sofern es die Spielregeln erlauben, betreten zu können. Entsprechend:
    - Dürfen **keine nicht erreichbaren** aber potentiell betretbaren Felder enthalten sein.
    - Um den **Wechsel zwischen** beiden **Kartenhälften** sicherzustelle

n, egal wie diese später vom Server kombiniert werden, müssen ≥ 51% der Felder jedes Randes betretbar sein.

- Diese und andere in diesem Dokument genannte Bedingungen/ Regeln sind vom Client einzuhalten und vom Server zu überprüfen. Letzterer reagiert, bei Bedarf, mit passenden Fehlerrückmeldungen und Sanktionen sodass die betroffene KI verliert und die andere KI automatisch gewinnt.

Während des Spiels müssen die Karte und deren bekannten Eigenschaften und wichtige Spielzustände von den Clients mittels **command-line interface** (CLI) für Anwender nachvollziehbar visualisiert werden. Mindestens alle anfallenden Kartendetails (ähnlich eines älteren CLI-Spiel, siehe z.B. Dwarf Fortress) aber aber auch den Endzustand (Sieg, Niederlage) der Spieler. Weitere Visualisierungsanforderungen werden gerade im Workshop mit dem Kunden finalisiert. Diese werden gegen Ende der Prototypenphase abschlossen sein.