Simulation zur Weltraumpiraterie

Projektstatus:

- UI: komplett
- Umgebung: komplett
- Erster autonomer Agent: debugging.

Entwickler:
Tim Beier

Forschungsfrage

- Eine Potenzialeinschätzung unterschiedlicher Regelwerke für Computerspiele anhand einer exemplarischen Simulation für Weltraumpiraterie
- Alternativ: Wie sinnvoll verhält sich abgesprochene Zusammenarbeitm gegenüber Unabgesprochener anhand einer exemplarischen Simulation für Weltraumpiraterie -Wie verhalten sich adaptive Agenten gegenüber einfachen/komplexen Regelwerken?

Untersuchung
Es findet eine Aufwandseinschätzung adaptiver Regelwerke und ihres Nutzen gegenüber statischer Regelwerke für Agenten statt. Bei der Aufwandseinschätzung wird die Qualität des Regelwerks in Kontrast zum Arbeitsaufwand und der benötigten Rechenzeit gesetzt.

Software:
Engine: Unity 3D
Sprache: C#
IDE: JetBrains - Rider
Dokumentation: atom
Auswertung: Microsoft Excel
Abgabe: PDF

Welt

Der Weltraum wird als Umfeld gewählt, da es einen sehr freien Raum repräsentiert und vielseitig
interpretiert werden kann.
Piraten werden als Entitäten/Agenten gewählt, da sie ein Konkurrenzverhalten inne halten aber dennoch Kooperationen möglich sind.
Die Piraten starten auf einem Rasterfeld aus quadratischen Feldern und haben eine

begrenzte Menge an Aktionen (Bisher 1). Piraten sind Teil einer Fraktion und greifen sich untereinander nicht an. Fraktionen sind automatisch miteinander verfeindet.

Aktionen

- Nichts tun: Notwendig, um falsche Aktion aus Zugzwang zu vermeiden
- Schießen: Abschüsse von Feinden inkludieren das Plündern und gewähren generische Ressourcen.
- Bewegen: Notwendig, sobald keine Feinde mehr in Reichweite sind

Regelwerke

- Zentralisiert -> Eine 'Fraktionsführung' besitzt alle verfügbaren Informationn und bestimmt Aktionen.
- Individiduell -> Jede Agent hat nur seine lokalen Informationen und bestimmt seine Aktion selbstständig.
  - Kooperation
    - Vorhersehung -> Eine Entität sieht die Aktionen einer anderen vorher und adaptiert seine Aktion.
    - Absprache -> Eine Entität äußert ihr Vorhaben und andere Entitäten willigt ein/ lehnt ab
      - Verwendung globaler (statischer?) Aktionsfolgen (Angriffsmanöver, Flugformation, Erkdungsformation)

Verfügbare Informationen

Entitäten und Alliierte(inklusive sich selbst) in Sichtweite sind anhand ihrer Fraktion bekannt.
Die Entitäten können Ressourcen bezahlen, um neue Informationen zu berechnen

- Entfernungen zu Entitäten in Reichweite
  - Entitäten in Angriffsreichweite
- Verhältnis Allierter zu Feinden

Non-Adaptiv-individuell

- 1. Bewege dich zum nächsten Verbündeten außerhalb deiner Sichtweite, wenn du in Unterzahl bist. (Falls existiert)
- 2. Schieße auf Gegner in Reichweite (falls existiert)
- 3. Bewege dich zum nächst nahesten Gegner (falls existiert)
- 4. Bewege dich in eine zufällige Richtung (erkunden)

Detailliert:

- Es gibt Feinde
    - Es gibt mindestens genausoviele Allierte wie Feinde
        - Es gibt einen Feind in Angriffsreichweite
            - Greife Feind an
        - Es gibt keine Feinde in Angriffsreichweite
            - Bewege dich zum nächstnahesten Feind.
    - Es gibt mehr Feinde als Alliierte
        - Es gibt einen Allierten der mehr als 1 Feld entfernt ist.
            - Es gibt ein freies Feld in die Richtung des Alliierten
                - Bewege dich auf das Feld
        - Es gibt kein freies Feld
            - Es gibt einen Feind in Reichweite
                - Greife Feind an
            - Es gibt keinen Feind in Reichweite
                - Bleibe stehen
- Bewege dich in eine zufällige Richtung oder gar nicht.

Adaptiv

Übergänge zwischen allen Aktionen mittels Markov und Q-Werten möglich.
Eigene Matritzen für: Verhältnis von Freundanzahl - Feindanzahl -> (<0, 0, >0)

Aktionsmatrix: (-1, 0, +1, Unwissen?)

- Schießen -> Schießenmatrix(angrenzende Felder: V-Werte)
    - Gibt eine Runde lang Boni auf Schießenmatrix & Bewegenmatrix abhängig für Felder mit Feind/Freund:
        - Schießenmatrix: Bonus auf Feld von Feind durch Paramerer v_schießenFeind (Startwert 0)
        - Bewegenmatriy: Bonus auf Feld von Feind durch Paramerer v_bewegenFeind (Startwert 0)
        - Schießenmatrix: Bonus auf Feld von Freund durch Paramerer v_schießenFreund (Startwert 0)
        - Bewegenmatrix: Bonus auf Feld von Feind durch Paramerer v_schießenFreund (Startwert 0)
- Bewegen -> Bewegenmatrix(angrenzende Felder: V-Werte)
- Nichts tun

Vorausschauend

Praktisch nicht möglich ohne sehr umfassendes Wissen über das Enviroment.

- Den Agenten fehlen Daten, um vorausplanen zu können.
    - Sichtweite

    - Gegnerische Ressourcen

Grundsätzliche Vorgehensweise:

- Loop through all units able to move
    - Loop through every space this unit may move to
        - Loop through every valid target per space
            - Simulate the attack and score the results
        - Return the highest scoring attack action for this unit
    - Return the highest scoring attack action from all units
    - If the attack score is greater than a minimum threshold
        - Perform the action
    - Else
        - Signal that we have no more moves

Messungen

- Es existiert ein Graph, welcher die die Ressourcenbalance jedes Agenten anzeigt.
- Die Veranschaulichung wird nach jeder Aktion vom Enviroment aktualisiert. (Bisheriger Wert += neue Balance)
- Die Anzahl der veranschaulichten Werte ist vorab freiwählbar. (z.B. die letzten 5-1000 Werte)
- Die Daten sind jederzeit abgreifbar und können somit einzelne Runden, oder ganze Simulationsdurchläufe darstellen.
- Weitere abgreifbare Daten:
    - Aktionsanzahl
    - Rundenanzahl
    - Agentenanzahl
    - Fraktionsanzahl

ToDo:

- Es ist möglich die Daten als .csv zu exportieren
- Eine genauere Auswertung mehrerer Durchläufe via Excel ist möglich
- Listener Methoden, welche speziell auf Berechnungskosten achten.

Offene Fragen

Welche Algorithmen eignen sind dynamisch genug für adaptive Regelwerke?

Konzept in Stichpunkten

Ausgangssitation:

- Eine begrenzte, rechteckige Spielwelt (z.B. 8x6 Felder)
- Die Welt ist mit Piraten verschiedener Fraktionen bevölkert(2-x Fraktionen)
- Rundenbasiert entscheiden sich Piraten für ihre Aktion(feste Reihenfolge)
- Regelmäßig fliegen Transportschiffe durch die Welt

Ziel/Selbstorganisation

- Piraten wollen sich eine maximale Menge an Ressourcen aneignen.
  (Ihre Ressourcenaneignung optimieren)

Belohnungsszenario

- Abschuss eines Schiffes -> +Ressourcen

Spielregeln:

- Schiffe lagern Ressourcen (100 max)
- Schiffe verlieren mit der Zeit Ressourcen (1 pro Tick)
- Besitzt ein Schiff keine Ressourcen mehr (0), so wird es zerstört
- Schiffe erhalten Ressourcen beim Zerstören von Schiffen
- Schiffe können schießen und verursachen Schaden in Ressourcen
- Schiffe besitzen eine begrenzte Sichtweite
- Ein Schiff, dass sich auf ein besetztes Feld bewegt erleidet Schaden(25) und bewegt sich nicht

Variablen

- Sichtweite(angrenzend)
- Scanner(angrenzend)
- Schießreichweite(angrenzend)
- Schaden(25)

Optionales

- Die Entitäten können Ressourcen bezahlen, um neue Informationen zu erhalten
  - Informationsaustausch mit Verbündeten.
  - Kaufen von Informationen anderer Fraktionen
- Aufkaufen von Entitäten feindlicher Fraktionen
- Schiffe mit 100 Ressourcen spalten sich in 2 Schiffe mit je 25 Ressourcen (Spawn auf freiem, angrenzendem Feld und restliche Ressourcen werden aufgeteilt)
  Neue Schiffe kopieren ihre Werte von ihrem Erschaffer (Fraktion, v-werte)
  -> Erfordert,dass Belohnungsszenarios des Klons für den Vater gelten, damit es nicht
  als Ressourcenverlust zählt sondern als Investition.
- Schildaktion, welche 1-3 Runden Schaden/einen Treffer abfängt

- Schiffe hinterlassen nicht passierbare Wracks
- Das Zerstören von Wracks gibt Ressourcen(50)
- Schiffe verursachen Flächenschaden bei Zerstörung
- Neue Schiffe kosten 50 Ressourcen und Zeitpunkt wird vom Agenten gewählt
- Schiffe können die Stärke des Feindes einschätzen
- Waffensysteme
- Upgrades
- Variable: Waffenschaden
- Variable: Geschwindigkeit
- Variable: Zielgenauigkeit
- Variable: Schätzgenauigkeit