

# Willkommen an Bord!

Onboarding-Dokument für die Strategie-Abteilung

Strategie-Team

## Herzlich willkommen im Strategie-Team!

Dieses Dokument dient als Leitfaden für deine Einarbeitung. Du wirst schrittweise an die verschiedenen Aspekte unserer Software herangeführt und bekommst die Möglichkeit, durch die Implementierung und Konzeption von zentralen Bestandteilen der Robotersteuerung direkt in die Entwicklung einzusteigen.

Die folgenden Aufgaben sind so konzipiert, dass du einen tiefen Einblick in die Architektur unserer Strategie-Software erhältst. Im Source Code findest du zahlreiche Kommentare, die dir nützliche Methoden und Funktionen für die jeweilige Implementierung erläutern.

## Setup

### Dein Einarbeitungsplan

Die Einarbeitung ist in mehrere Module aufgeteilt, die sich mit den Kernkomponenten unserer Strategie befassen:

- **RoleManager:** Die dynamische Zuweisung von Rollen an unsere Roboter.
- **Goalie:** Die erste Implementierung unseres Torhüters.
- **Striker:** Die Implementierung eines Stürmers mittels eines Zustandsautomaten.
- **Defense:** Die Entwicklung einer grundlegenden Verteidigungsstrategie.
- **Goalie v2:** Die Weiterentwicklung des Torhüters zu einem Zustandsautomaten.
- **Verbesserungen:** Die Auswahl und Implementierung einer von mehreren vorgeschlagenen Verbesserungen.

## 1 RoleManager: Die dynamische Rollenzuweisung

Der RoleManager ist eine entscheidende Komponente, die dafür verantwortlich ist, jedem Roboter auf dem Feld die für die aktuelle Spielsituation am besten geeignete Rolle zuzuweisen. Eine effiziente Rollenverteilung ist der Schlüssel zu einer erfolgreichen Strategie.

### Deine Aufgaben

1. Implementiere `create_roles_to_assign_list`: Diese Methode soll eine Liste der zu gebenden Rollen erstellen. Achtung! Diese Methode ist lediglich dafür da, eine

Liste an Rollen zu erstellen. Die konkrete Verteilung findet erst später statt. Zum Anfang sind nur die Rollen Goalie, Defender und Striker vorgesehen.

2. Implementiere `calc_costs`: Diese Funktion bewertet, wie gut ein Roboter für eine bestimmte Rolle geeignet ist.
3. Implementiere `create_cost_matrix`: Erstelle eine Kostenmatrix, die die Scores aus `calc_costs` für alle Roboter und Rollen enthält. Achtung! Der Goalie-Roboter wird am Spielanfang fest vergeben und ist durch `self.core.game_data_provider.getGoalie()` findbar.
4. Verknüpfe alles in der `update`-Methode, um die Rollen bei jedem Simulationsschritt neu zu berechnen und zuzuweisen.

## Test

Starte die Software und überprüfe in der Visualisierung oder den Logs, ob den Robotern erfolgreich Rollen zugewiesen wurden.

## 2 Goalie: Unser Fels in der Brandung

Der Goalie ist die letzte Verteidigungslinie und schützt das Tor. In dieser ersten Implementierung soll er eine grundlegende Funktionalität erhalten.

### Deine Aufgabe

Implementiere die `update`-Methode des Goalies: In dieser Methode soll die Logik implementiert werden, die der Goalie in jedem Zeitschritt ausführt. Für den Anfang genügt es, wenn der Goalie sich auf der Torlinie zwischen dem Ball und dem Tor bewegt.

## Test

- Starte die Software und überprüfe, ob dem Goalie die korrekte Rolle zugewiesen wird.
- Teste, ob der Goalie wie erwartet funktioniert und den ihm zugewiesenen Skill ausführt.

## 3 Striker: Der Torjäger

Der Striker ist dafür zuständig, Tore zu schießen. Um komplexe Verhaltensweisen zu ermöglichen, wird er als Zustandsautomat implementiert.

### Deine Aufgaben

1. Implementiere `_setup_states`: Definiere die verschiedenen Zustände (`GET_BALL`, `SHOOT`) und Zustandsübergänge.
2. Implementiere die Action-Methoden für jeden Zustand.
3. Implementiere die Zustandsübergangsfunktionen
4. Implementiere die `update` Methode

## Konzeptionelle Überlegungen

Überlege dir alternative Metriken für die Zustandsübergangsfunktionen (z.B. zeitliche oder geometrische Metriken)

## 4 Defense: Der defensive Riegel

Eine solide Verteidigung ist die Basis für ein erfolgreiches Spiel. In diesem Schritt implementierst du eine grundlegende Formationsverteidigung.

### Deine Aufgaben

1. Definiere eine Verteidigungslinie vor dem eigenen Tor.
2. Implementiere `compute_ball_intercept_pose`.
3. Implementiere `create_positions_around_center`.
4. Implementiere `distribute_positions_among_robots`.
5. Verknüpfe alles in der `update`-Methode.

### Konzeptionelle Analyse

Notiere 3 konzeptionelle Schwächen dieser Verteidigung und entwickle ein Konzept zur Verbesserung (ohne Implementierung).

## 5 Goalie v2: Der mitdenkende Torwart

In der zweiten Version soll der Goalie zu einem intelligenten Agenten werden, der aktiv auf Schüsse reagiert.

### Deine Aufgaben

1. Implementiere den Goalie als Zustandsautomaten mit den Zuständen `DEFEND_ON_CIRCLE` und `INTERCEPT_SHOT`.
2. Schreibe die Stubs aller notwendigen Methoden (`_setup_states`, Actions, Übergänge).
3. Implementiere die Observer-Funktionen:

- `is_shot_on_goal`
- `is_shot_aborted`

Diese Funktionen sollen auf Grundlage des Winkels der Balltrajektorie zur Linie von der Tormitte zur Ballposition sowie der Ballgeschwindigkeit eine Entscheidung treffen, ob die Ballbewegung als Schuss erkannt werden soll bzw. ob ein Schuss als Schuss aberkannt werden soll.

4. Implementiere den vollständigen Zustandsautomaten.

## 6 Wähle deine Verbesserung

Nachdem du die Grundlagen kennengelernt hast, wähle mindestens eine der folgenden Verbesserungen aus:

- **Striker:** Füge Zustände `STEAL_BALL` oder `INTERCEPT` hinzu.
- **Defense:** Implementiere eine verbesserte Verteidigung, z. B. eine Linie senkrecht zur Ball-Tor-Verbindung.
- **Goalie:** Füge den Zustand `KICK_OUT` hinzu, damit der Goalie den Ball aus dem Strafraum klärt.

## Abschluss

Bei Fragen und Unklarheiten steht dir das gesamte Team jederzeit zur Verfügung. **Viel Erfolg und Spaß bei der Einarbeitung!**