Saarbrücker Forschungstage @ DFKI Spielwiese

Infos und How-to's rund um den Workshop Humanoide Roboter und heuristische Spiele-KI an den Saarbrücker Forschungstagen

Kai Waelti

27. June 2019

Intro

- Einführung in einfache Robotik mit humanoiden Roboter Pepper
- Grundlegende Algorithmen der künstlichen Intelligenz KI und
- Computer Vision CV werden diskutiert
- Den Studenten wird alles zum tüfteln bereitgestellt

Das DFKI

- Heute macht jeder KI aber das DFKI gibts nun seit über 30 Jahren

  $\Rightarrow$ kein Newcomer

-

Das Pentagon der Innovation

Figure 1: Standorte DFKI

Geplantes Verhalten

Es sollte eine Anwendung entwickelt werden, die es ermöglicht gegen Pepper Tic Tac Toe zu spielen. Um ein Spiel zu starten gibt man Pepper sprachlich einen Startbefehl. Danach fordert Pepper den Spieler auf seine Spielzüge zu spielen und wenn Pepper am Zug ist, zeigt er dem Spieler, wo er Peppers Spielstein platzieren soll. Wird ein unerlaubter oder falscher Spielzug erkannt, meldet Pepper dies und fordert den Spieler auf die auf dem Display angezeigte Spielsituation wiederherzustellen. Gewinnt einer der Spieler, verkündet Pepper den Gewinner und wartet auf einen neuen Spielstart, oder das beenden seines Verhaltens.

Setup

Damit Pepper TicTacToe spielen kann, müssen zuerst ein paar Probleme gelöst werden. Zuerst muss Pepper seine Umgebung - in diesem Fall das Spielfeld, die Magnete und Züge - erkennen können.

Die Umgebung wird in der Anwendung über den Kamera Sensor wahrgenommen. Das hat zur Folge, dass die Kamerabilder verarbeitet werden müssen, damit Pepper den Spielstatus weiss und überlegen kann welchen Schritt er als nächstes ausführen soll. Die Schritte die Pepper ausführt, verändern das Spiel ebenfalls und sollten in der Berechnung berücksichtigt werden.

Mit dem internen Weltmodell und einfachen Bedingungsregeln findet Pepper heraus, welchen Zug er als nächstes machen soll. Damit Pepper den Spielzug ausführen kann braucht er noch Hilfe eines Menschen, da seine Hand weder genug kräftig, noch genau ist. Deshalb, muss er seine Wünsche über Sprache klar und deutlich ausdrücken können. Neben Sprache wird in der Anwendung auch auf Körpersprache gesetzt. Wobei Pepper auf die Position wo sein Magnet gesetzt werden soll zeigt.

Lösungsansatz - Agententypen

Als Grundstruktur wird ein Model-based Agent verwendet. Dieser ist so ausgelegt, dass er einem festen Ablauf folgt mit dem Ziel, das Spiel erfolgreich zu beenden.

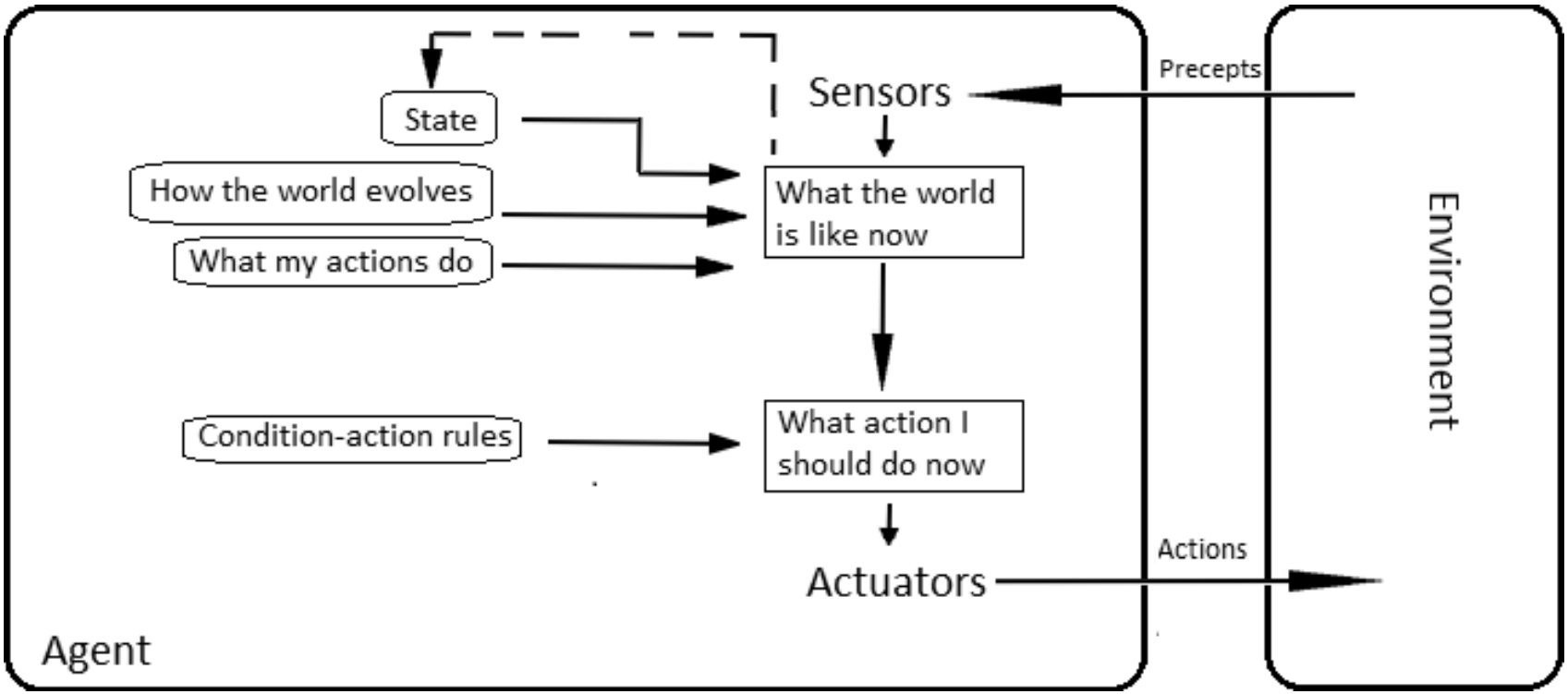

Figure 2: Model-based Reflex Agent

Ansatz für Spracheingabe für Spielstart / -abbruch

Ideen woran getüftelt werden kann

- Mach Pepper unbesiegbar
  - Schau dir die Bedingungsregeln in `TicTacToeAiHeuristic.py` an und
    - gibt es momentan eine Reihe von Zügen womit du Pepper jedes mal besiegst?
    - überlege dir was daran verbessert werden könnte
    - sieh dir auch `TicTacToeAiRand.py` an als alternative Spielstrategie die einfach einen zufälligen Zug zurückgibt
    - suche nach alternativen Spiel-Strategien [//]: # (TODO: Einfaches RL? ML? DL?)
- Die Bildverarbeitung verbessern
  - Studiere die CV Pipeline in `DetectBoard.py` und
    - versuche die Bilderkennung in verschiedenem Licht aus
- Andere/Mehr Behaviors und Animationen
  - Momentan reagiert Pepper nur auf falsche Züge, wenn man versucht zu Cheaten oder wenn ein Spiel zu ende ist
    - Diese Reaktionen können verändert werden oder
    - es können neue hinzugefügt werden, bspw.
    - Denkbar wäre auch ein Erkennen was nicht in Ordnung ist und kommunikation zur Behebung an Spieler

Eingesetzte externe Software

+ Python 2.7
+ OpenCV 3.4
+ Entwicklungsumgebung PyCharm

Requirements

Folgende Software muss vorher installiert werden:

- VirtualBox >= v6 (aktuellste Version v8.0.8 emfohlen)
  1) Eine Installations-Anleitung für jegliche Betriebssysteme kann auf dem VirtualBox Wiki gefunden werden
  2) Nach der Installation kann die von uns gelieferte `.ova` Datei in der VirtualBox Anwendung über `Datei` ‘ → ‘ `Appliance importieren` ‘ → ‘ Angabe genauer Pfad zur `.ova` Datei + `next` klicken <» TODO add description for complete import

Verbindung mit dem WLAN:

SSID: `PRAF`
KEY: `poposoft`

Wie weiter

1) Einteilung in Gruppen
2) Jede Gruppe arbeitet in eigenem Git Repository unter tictactoe-G[GRUPPE-NR]
3) Fragen ist willkommen :)

Beispiel anhand Gruppe 10

1) Terminal öffnen (`Ctrl+Alt+t`)
2) Ausführen `mkdir source && cd source`
3) GitLab öffnen und navigieren zu https://gitlab.enterpriselab.ch/forschungstageAtdfki/group-repositories
4) Klick auf Projekt mit zugehörigen Gruppen Nummer
5) Kopieren `clone` mit `https` Adresse
6) Ausführen im Terminal ‘git clone https://gitlab.enterpriselab.ch/forschungstageAtdfki/group-repositories/tictactoe-g10.git"
7) Ausführen `cd tictactoe-G10` (eigene Gruppen Nummer einsetzen)

Was macht was und was ist wofür zuständig

- `Main.py`
  - Ist das Startup File für das Hauptprogramm
  - Damit werden alle Komponenten gestartet
- `DetectBoard.py`
  - Ist die Datei die den Hauptsächlichen Bildverarbeitungscode beinhaltet
    - Im Ordner image_processing befinden sich zudem noch 3 weitere Hilfsdateien mit Hilfsfunktionen die von `DetectBoard.py` aufgerufen werden
  - TODO: Pipeline beschreiben
- `TicTacToe.py`
  - Darin wird die gesamte Spiellogik abgewickelt
  - `TicTacToeAiHeuristic.py`
    - Darin befindet sich die Spiellogik bestehend aus klar vordefinierten Bedingungsregeln
    - Als Alternative zeigt `TicTacToeAiRand.py` eine rein zufällige Spielstrategie