# CAT-MOUSE-CHEESE

Torben Miller (3164082)
Fabian Biedlingmaier (3224303)
Nicolas Hartlieb (Matrikelnummer)

Heidelberg, am 30. August 2020

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Zusammenfassung

Nico

# 1 Aufgabenstellung

Nico

## 1.1 Maps Robo

Nico

## 1.2 Technischer Stand minmax Kraftbasiert etc. A Star

Nico

## 1.3 Zielsetzung

Nico

# 2 Konzept

## 2.1 Grundverhalten

Die Katze hat zwei wesentliche Grundbestandteile. Sie kann ‚auf einen gegebenen Punkt, „homen“ und sie besitzt eine Kollisionsvermeidung.

## 2.2 Homing

### 2.2.1 Ziele

Für das Homing gibt es vier verschiedene Ziele (1). Den Käse, der Mittelpunkt zwischen Käse und Maus, die Maus und der vermutete Punkt auf den sich die Maus zubewegt.

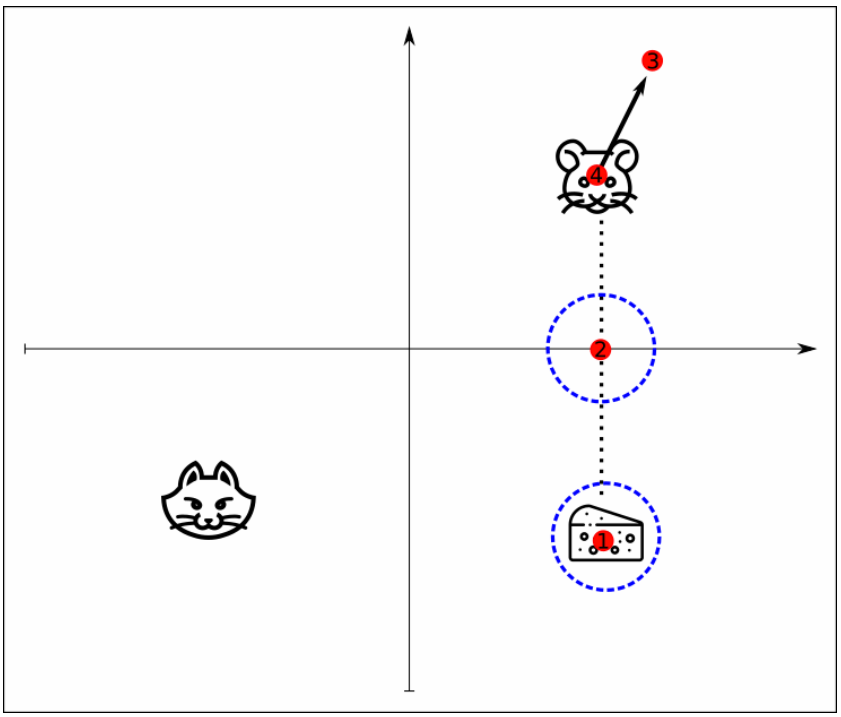


Abbildung 1: Ziele für das Homing

### 2.2.2 Zustandsautomat

Die Zielauswahl wird durch einen Zustandsautomat (2) festgelegt. Der Zustandsautomat besitzt drei Zustände, mit dem Zustand „go to cheese“ als Startzustand. Der Zustand „go to cheese“ enspricht dem Käse, der Zustand „go to mid“ den Mittelpunkt zwischen Maus und Käse als Ziel und der Zustand „hunt“ fasst die zwei Ziele Maus und den vorrausberechnenten Punkt zusammen. Wie der Zustand „hunt“ sich verhält wird im Abschnitt 2.2.3 genauer erläutert.

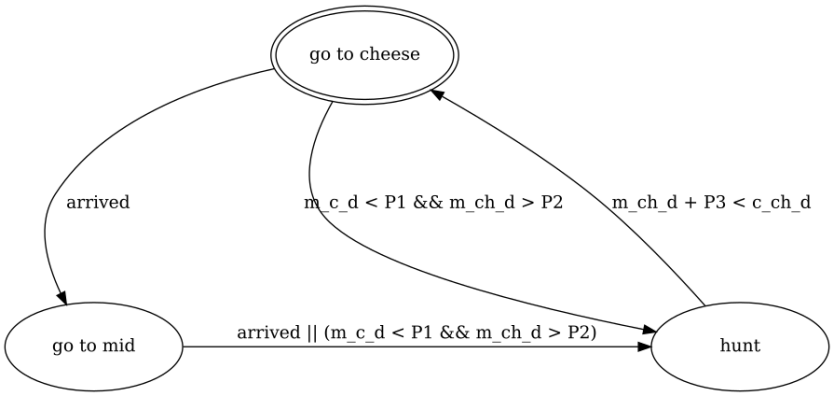


Abbildung 2: Zustandsautomat

Es gibt drei Kriterien die einen Zustandswechsel zur Folge haben. Das „arrived“ Kriterium ist hierbei das einfachste und wird erreicht sobald ein gewisser Abstand zum Zielpunkt unterschritten wird.
Das zweite Kriterium „$m\_c\_d < P_1 \wedge m\_ch\_d > P_2$“ besagt: sollten der Abstand zwischen Katze und Maus einen gewissen Wert $P_1$ unterschritten haben und zusätzlich der Abstand zwischen Käse und Maus einen weiteren Wert $P_2$ überschritten haben so wird der Zustand gewechselt. Umgangsprachlich ausgedrückt bedeutet es das sollte sich die Katze nah an der Maus befinden und die Maus aber noch weit genug vom Käse weg sein so wird die Maus gejagt. Wobei der Parameter $P_1$ „nah “ und der Parameter $P_2$ „noch weit genug“ eingerenzt. $P_1$ und $P_2$ wurden empirisch festgelegt.
Das dritte Kriterum „$m\_c\_d + P_3 < c\_ch\_d$“ bedeutet: ist der Abstand zwischen der Maus und dem Käse kleiner dem Abstand zwischen der Katze und dem Käse so erfolgt der Zustandswechsel. Hier wird ein Wert $P_3$ hinzugefügt der als Hysterese fungiert.

Im Zustand „go to cheese“ löst das „arrived“ Kriterium einen Wechsel in den Zustand „go to mid“ aus. Äquivalent dazu löst das gleiche Kriterium im Zustand „go to mid“ den Wechsel in den Zustand „hunt“ aus. Von beiden Zuständen, „go to cheese“ und „go to mid“, findet ein Wechsel in den Zustand „hunt“ statt, sollte das zweite Kriterium, „$m\_c\_d < P_1 \wedge m\_ch\_d > P_2$“, erfüllt sein. Im Zustand „hunt“ greift das dritte Kriterium, „$m\_c\_d + P_3 < c\_ch\_d$“, um den Wechsel in den Zustand „go to cheese“ einzuleiten.

### 2.2.3 Zustand: „hunt“

Grundlegend wird im Zustand „hunt“ entschieden ob die Maus direkt als Ziel angefahren wird oder ob die Katze versucht die Bewegung der Maus vorrauszusehen.
Das vorrausberechnen der Maus bewegung bezweckt eine Schlingerbewegung der Maus weg vom Käse. Dieser Ansatz zielt auf das Gegenspiel gegen Kraftbasierte Mäuse ab.

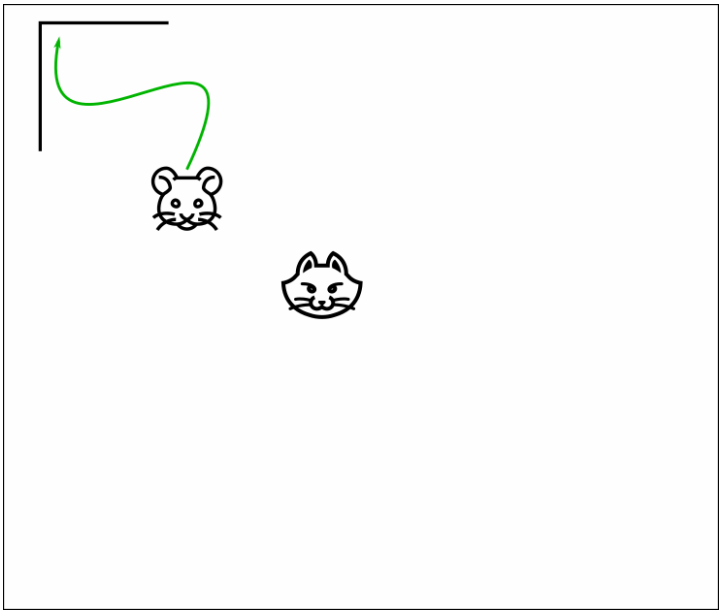

Abbildung 3: Schlingerbewegung

Diese Schlingerbewegung wird errreicht indem

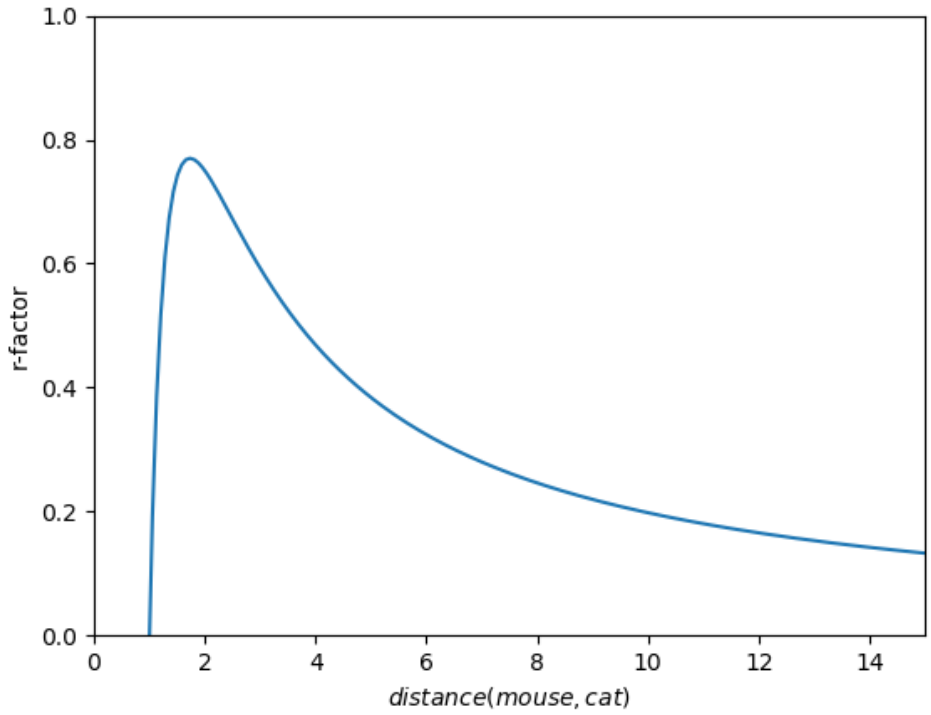


Abbildung 4: r-Faktor Funktion

## 2.3 Collison Avoidance

Fabian Die Drehung der Katze wird durch einen Radianten [1] angegeben. Ein positiver Wert führt zu einer Drehung gegen den Uhrzeigersinn, dementsprechend führt ein negativer Wert zu einer Drehung in Uhrzeigersinn. Die Lasersensoren scannen 360 Grad um die Katze in je 1 Grad Schritten, abgegeben in Radianten [2], Objekte werden ab einer entfernung von 3,5 Metern erkannt. Bei der Kollisionsvermeidung werden nicht alle 360 Werte berücksichtigt. Die relevanten Werte werden durch einen Parameter angegeben [3], welche angibt wie groß der relevante Scannbereich sein soll.

### 2.3.1 Funktionsentwiklung

Folgende Kriterein waren für die Funktion der Kollisionsvermeidung wichtig.

- Je geringer die Distanz zu einem Objekt [4] desto größe die Kraft
- Je geringer der Winkel einem Obekt zur Fahrtrichtung der Katze, desto größer deren Kraft

Nun ist ein genauer Blick auf die Wertebereiche der relevanten Variablen nötig.

| Variable | Wert | Auswirkung auf die Kraft |
|---|---|---|
| sensor.angular | $[0 : rel\_phi]$ | negativ (Drehung mit den Uhrzeigersinn) |
| sensor.angular | $[2\pi : 2\pi - rel\_phi]$ | positiv (Drehung gegen den Uhrzeigersinn) |
| sensor.range | gegen 0,8 | Kraft gegen 0 |
| sensor.range | gegen 0 | Kraft wird größer |

$$Force = -\sin(sensor.angular) * (rel\_range - sensor.range)$$

Die negative Sinusfunktion wird hier als Grundgerüst genommen. Da sich sensor.angle zwischen 0 und $2/pi$ bewegt. Werte nahe der 0 führen zu einer negativen Kraft und bewirken somit eine Rechtsdrehung. Werte nahe $2\pi$ bewirkeln hingegen eine Linksdrehung. Die so entstehende, Winkel abhänige, Kraft soll nun größer werden je kleiner die Distanz ist.

### 2.3.2 Maus reausrechen

Die aufgestellte Funktion erzeugt das gewünschte Verhalten, zeigt jedoch bei späterne Tests einen gravirenden Fehler. Die Katze weicht nun bei der Jagt nach der Maus selbigerauch aus und ist so nicht im Stande die Maus zu fangen.

[1] Wertebereich [0:2 $\pi$]
[2] 1 Gard 0,0174533
[3] rel_phi Zeile 123
[4] sensor.range

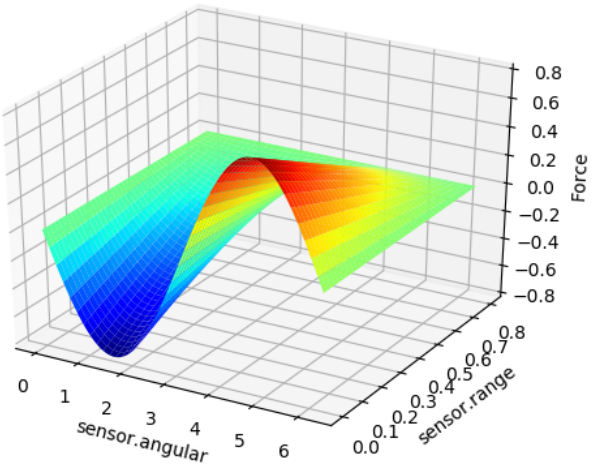


Abbildung 5: Funktion Collison Avoidance

Um dieses Problem zu lösen wurde ein Speziallfall entwickelt. Ziel ist es die Maus für die Kollisionsvermeidung unsichtbar zu machen.

**Strategie:**

1. Ist Maus ist in Scannreichweite?
2. Gescannte Range entspricht der Entfernung zur Maus
3. Relevante Winkle in bezug auf die Größe der Maus berechnen
4. Alle sensor.range-Werte der relevanten Winkel auf infinity setzen

Sobald die Maus in Sensorreichweite ist, ist es wichtig zu überprüfen ob die Sensor.range der mathematsichen Distanz zur Maus entspricht. Falls man dies nicht überprüft kann es passieren, dass sich ein Hindernis zwischen Maus und Katze befinden. Das Hindernis wir von dem Alorithmus als Maus tituliert und für die Kollisionsvermeidung nicht berücksichtigt. Dies hat zur Folge, dass die Katze in das Hindernis hineinfährt.

Sofern diese Fälle berücksichtigt worden sind, kann es daran gehen die relevanten Sensorwerten zu löschen( auf infinity zu setzten).

Zuerst wird der angkle_mouse berechnet dies ist der Winkel aus Sicht der Katze zur Maus.

$$angle\_mouse = tan(\frac{\Delta x}{\Delta y}) - cat\_phi$$

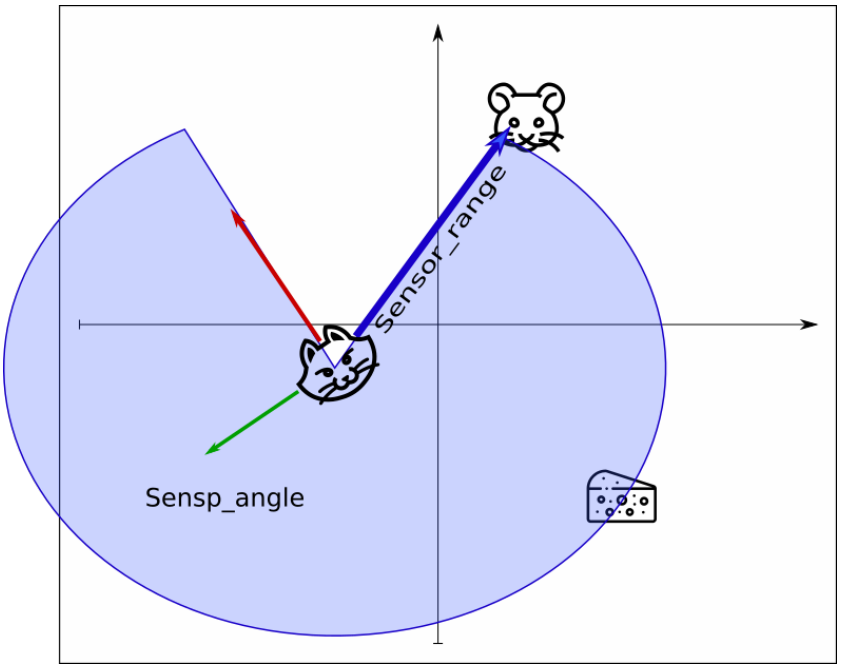


Abbildung 6: Blickwinkel der Katze

# 3 Dikussion und Ausblick

Alle

## 3.1 Videos

Alle

## 3.2 Szenarien

Alle

# 4 Hallo

## 4.1 whadiwahfliwahfi

Facilis repudiandae deleniti facere tempora tenetur nostrum eos iusto. Ipsum non nihil magni rem quam. Quas autem non esse ut voluptatem omnis. Adipisci laboriosam aut omnis nam saepe impedit. Deleniti voluptates corporis in. Dolorem hic dolore quibusdam sed non sed. Impedit ea provident atque quo. Dolor eaque quia consectetur veniam in voluptatum. A nihil voluptatem sed autem similique molestiae. Tenetur veniam architecto iure ducimus natus atque hic. Repellat fuga animi nam ut neque et ea sunt. Odit velit esse quidem facilis. Quibusdam qui est et rem a velit distinctio. Ut maiores dignissimos rem provident iure sed. Dolores quis voluptates error non eaque. Atque autem sed labore velit. Repellat itaque animi sit fuga omnis ut velit. Voluptatem nostrum eos ea eligendi adipisci nemo veritatis. Est temporibus quaerat nostrum id. Ullam voluptatem neque hic at odit et architecto laboriosam.

# 5 Hallo2