

# Doc of PathPlanning Approaches for GeckoBot

Lars Schiller

13. Juni 2019

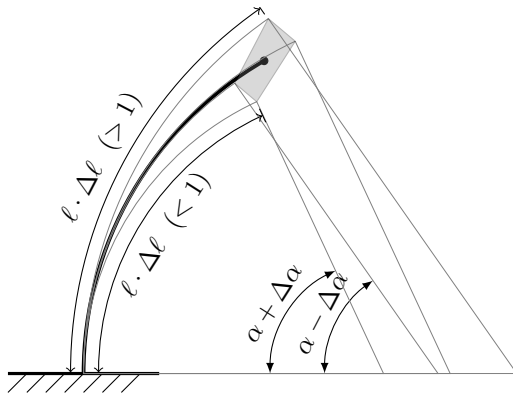
## Inhaltsverzeichnis

<b>1</b>	<b>Predicting the next pose of the robot</b>	<b>1</b>
1.1	Modeling of Soft Bending Actuator . . . . .	1
1.2	Modeling of the robot . . . . .	2
<b>2</b>	<b>Path Planning with Search Tree</b>	<b>2</b>
2.1	Different Gait Patterns for a curve . . . . .	2
2.2	Search Tree . . . . .	3
2.3	Search Tree with weights . . . . .	5
2.4	Simulation Results Curve . . . . .	6
2.5	Simulation Results Straight . . . . .	7
2.6	What happens if Process Noise occurs? . . . . .	7
2.7	Conclusion . . . . .	7
<b>3</b>	<b>Path Planning with Analytic Model</b>	<b>7</b>
3.1	Problem Statement . . . . .	7
3.2	Approach: Guess structure for a analytic model for walking curves . . . . .	8
3.3	Approach: Find a reasonable structure . . . . .	9
3.4	Approach: Optimize Extra leg bending Angle for given extra torso bending . . . . .	11

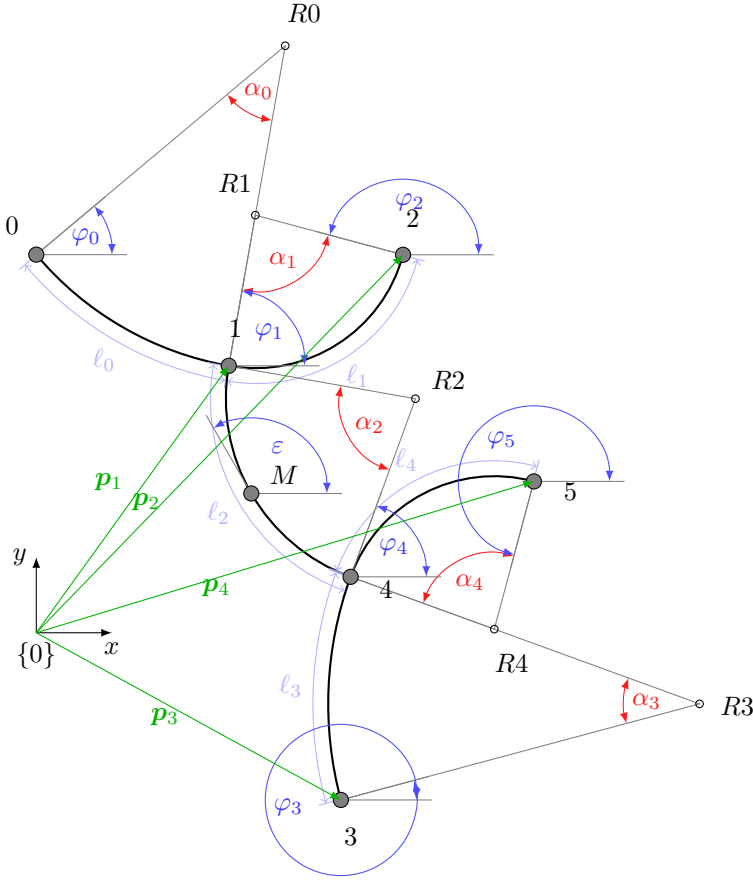
## 1 Predicting the next pose of the robot

### 1.1 Modeling of Soft Bending Actuator

- Einführung virtueller Längen, um größeren Bereich erreichen zu können, und dennoch die Annahme von *Constant curvature* nutzen zu können.
- Weil Sehr effektiv zu rechnen.



## 1.2 Modeling of the robot



- Zustands- und Eingangsgrößen:

$$\mathbf{x} = [\boldsymbol{\alpha} \ \boldsymbol{\ell} \ \varepsilon], \quad \mathbf{r} = [\boldsymbol{\alpha}_{\text{ref}} \ \mathbf{f}]$$

- Innere Spannung:

$$\begin{aligned} \sigma(\mathbf{x}_k) = & w_\ell |\boldsymbol{\ell}_k - \boldsymbol{\ell}_n|_2 \\ & + w_\alpha |\boldsymbol{\alpha}_k - \boldsymbol{\alpha}_{\text{ref},k}|_2 \\ & + w_\varphi |\text{diag}(\mathbf{f}_k)(\boldsymbol{\varphi}_k - \boldsymbol{\varphi}_{k-1})|_2 \end{aligned}$$

- Minimale Spannung:

$$\begin{aligned} \min_{\mathbf{x}_k \in \mathcal{X}} \quad & \sigma(\mathbf{x}_k) \\ \text{s. t.} \quad & \|\text{diag}(\mathbf{f}_k)(\mathbf{P}_k - \mathbf{P}_{k-1})\|_2 = 0 \end{aligned}$$

- Folgepose:

$$\boldsymbol{\rho}_k = [\mathbf{x}_k \ \mathbf{P}_k \ \mathbf{f}_k] = \text{fun}_{\mathcal{P}}(\mathbf{r}_k, \boldsymbol{\rho}_{k-1})$$

## 2 Path Planning with Search Tree

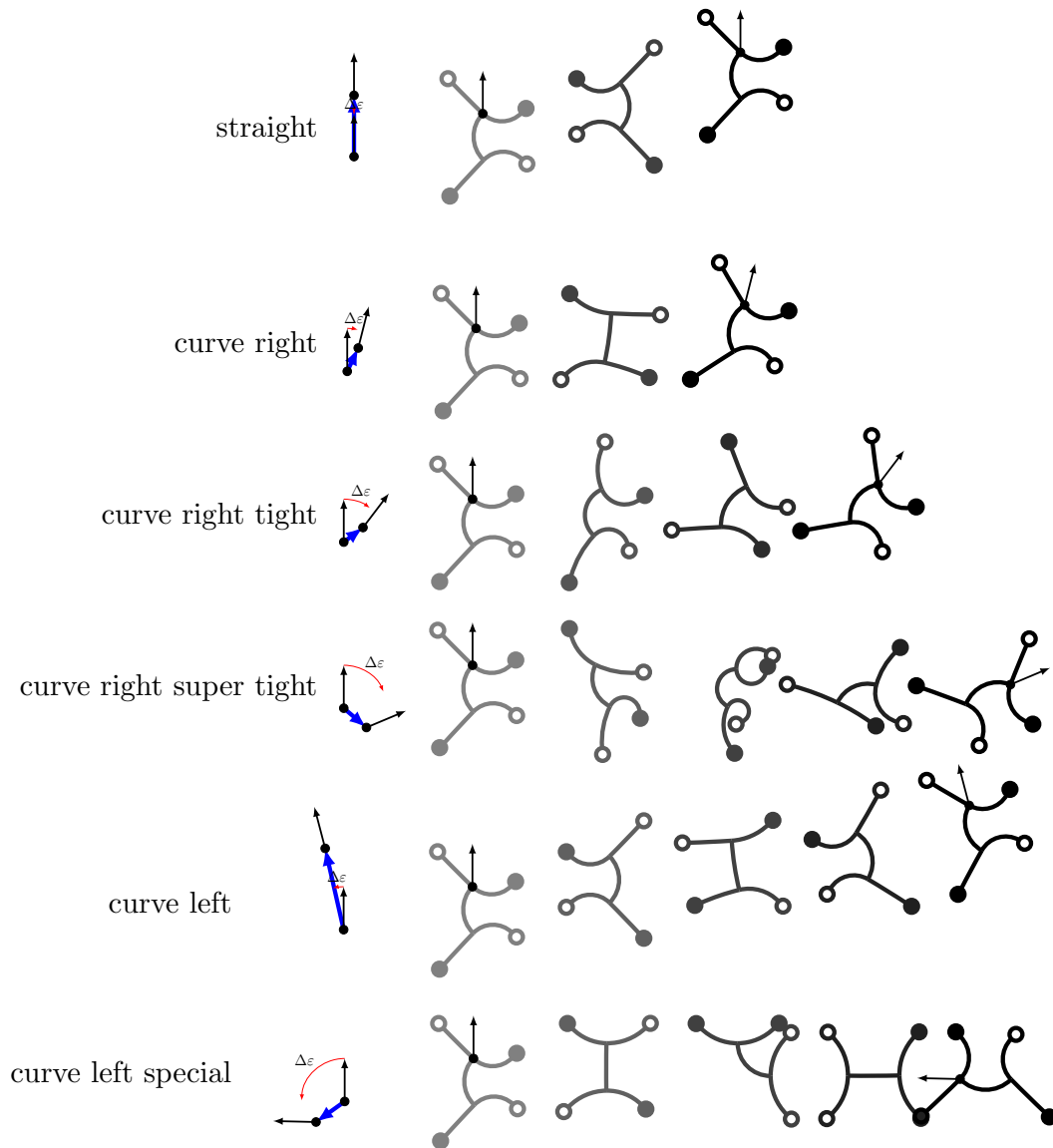
### 2.1 Different Gait Patterns for a curve

- Vom Kinematic Paper sind schon verschiedenen Laufmuster für Kurven bekannt, basierend auf dem Minimierungsproblem:

$$\min_{\boldsymbol{\alpha} \in \mathcal{A}} \varepsilon(\boldsymbol{\alpha}) \tag{1}$$

wobei  $\boldsymbol{\alpha}$  die Referenzwinkel von zwei Posen, also einem Zyklus enthält.

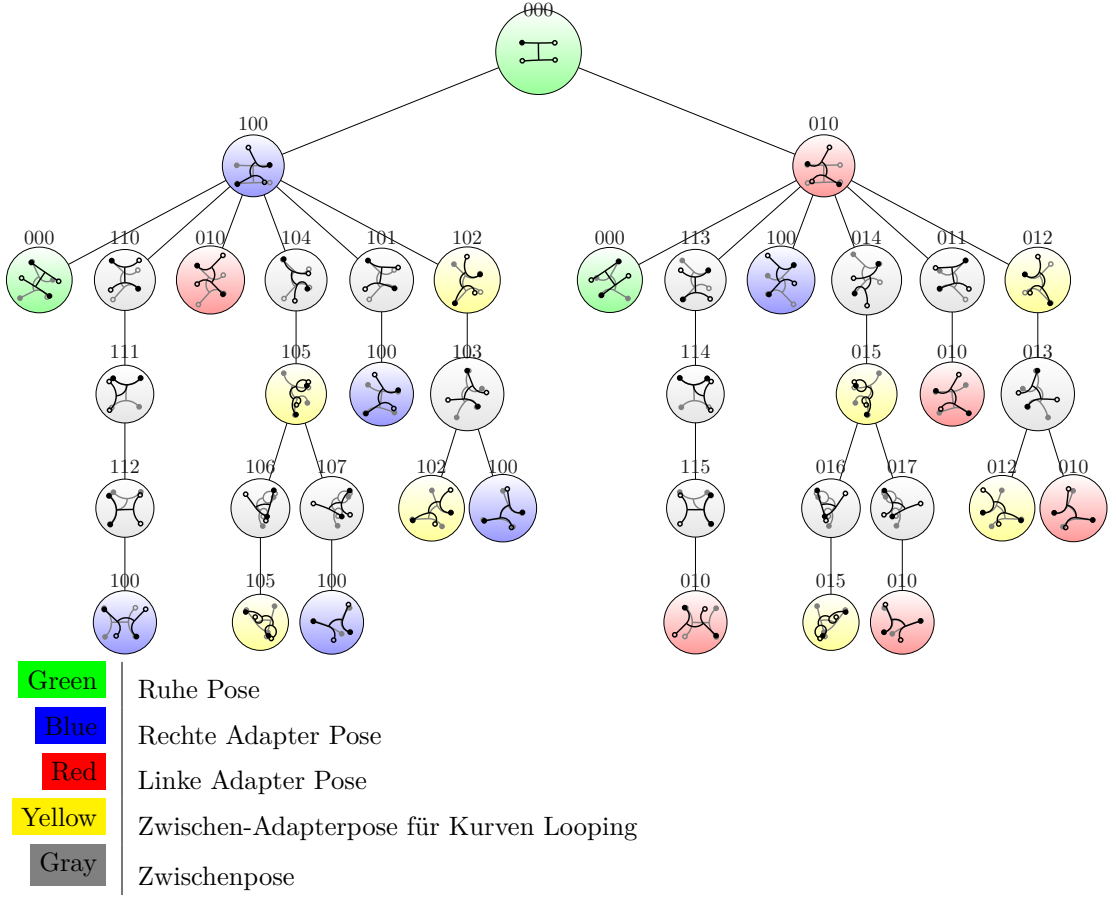
- Beispiele:



- Idee: Eine ausgewählte Anzahl an Posen, als Grundbausteine für einen beliebigen Gang.
- Diese dann wie Legosteine aufeinander setzen, um von A nach B zu gelangen.

## 2.2 Search Tree

- Folgender Suchbaum wurde implementiert.



- Dabei hat jede Kante des Baums eine Richtung und eine Gewichtung  $w$ .
- Die Gewichtung  $w = ((\delta x, \delta y), \delta \varepsilon)$  gibt an, inwieweit das entsprechende Kind (Folgepose repräsentiert durch den Knoten, der mit der gewichteten Kante mit dem momentanen Knoten  $k$  verbunden ist) den Roboter relativ zur momentanen Orientierung bewegt:  $(\delta x, \delta y)$  und wie weit diese Pose ihn drehen wird:  $\delta \varepsilon$ .
- Für eine gegebene, momentane Pose  $\rho_k$  wird für alle Kandidaten  $j \in [0, \dots, J-1]$  ausgerechnet, wie weit der Abstand  $d_j$  der potentiell neuen Pose  $\rho_j$  zum Ziel  $\bar{x}$  ist:

$$d(\rho_k, w_j, \bar{x}) = \left\| \bar{x} - \left( p_{1,k} + R(\varepsilon_k) \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} \right) \right\|_2 \quad (2)$$

- Außerdem wird die Richtungsabweichung  $\Delta \varepsilon_j$  aller potentiell neuen Pose  $j$  berechnet:

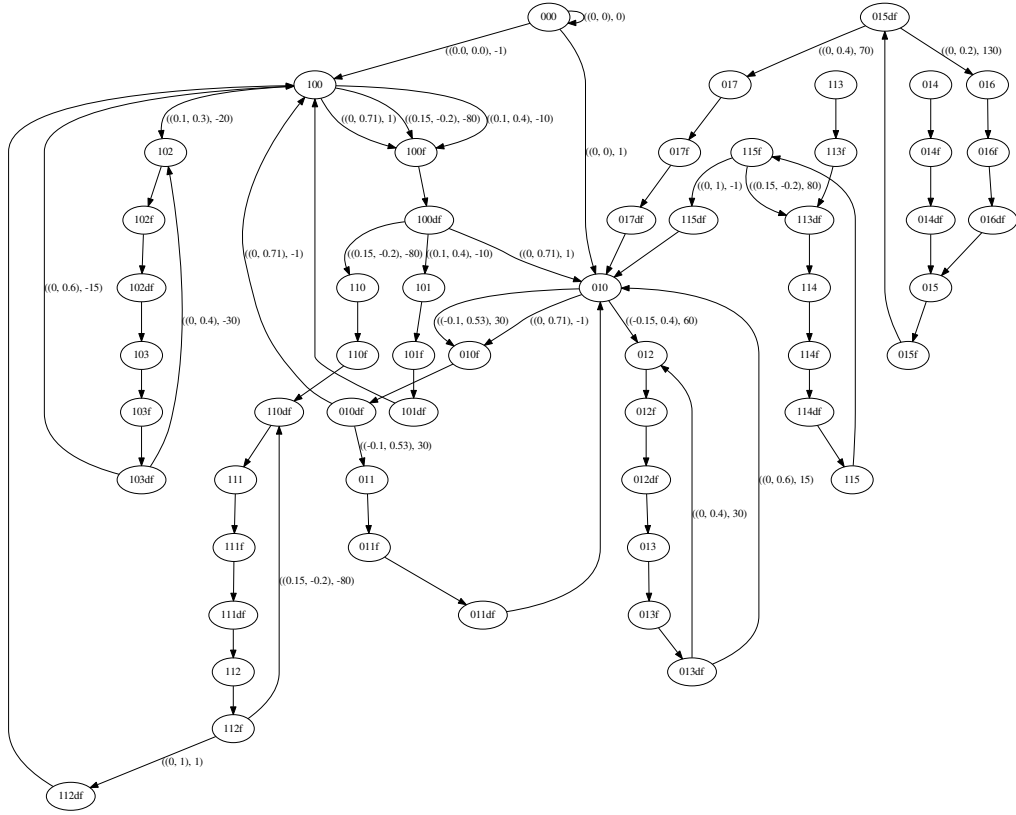
$$\Delta \varepsilon(\rho_k, w_j, \bar{x}) = \angle \left( \bar{x} - \left( p_{1,k} + R(\varepsilon_0) \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} \right), R(\varepsilon_k + \delta \varepsilon) \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \quad (3)$$

- Die Folgepose  $\rho_{k+1}$  ergibt sich dann aus dem Minimum der mit  $a = .5$  gewichteten Summe von Abstand und Orientierungsabweichung für alle Möglichkeiten  $\rho_j$ :

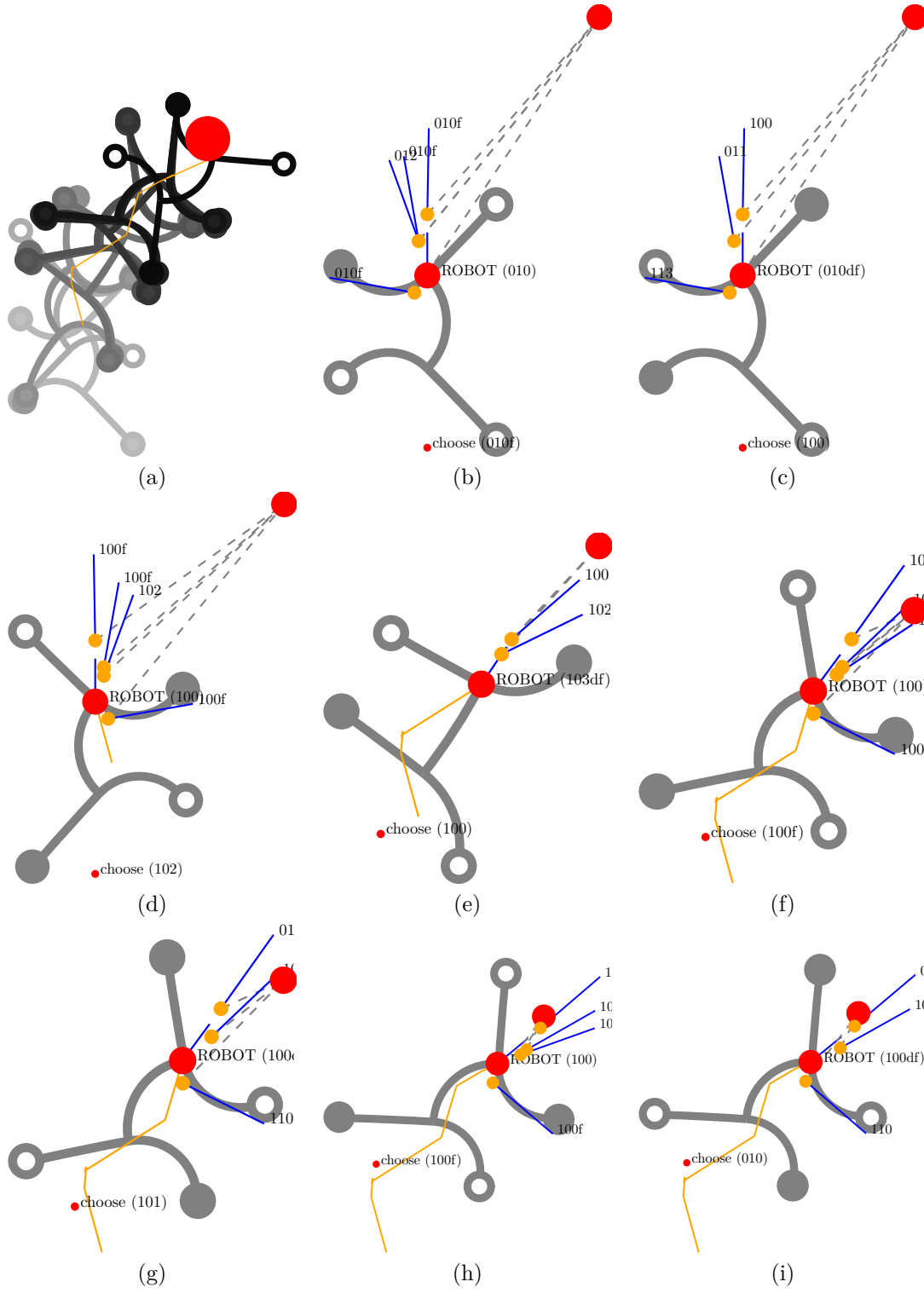
$$\rho_{k+1} = \min_j \left( a \frac{d_j}{d_{\min}} + (a-1) \frac{\Delta \varepsilon_j}{\Delta \varepsilon_{\max}} \right) \quad (4)$$

- wobei  $\Delta \varepsilon_{\max}$  die maximale Orientierungsabweichung aller Möglichkeiten ist; und  $d_{\min}$  der minimale Abstand aller Möglichkeiten ist.

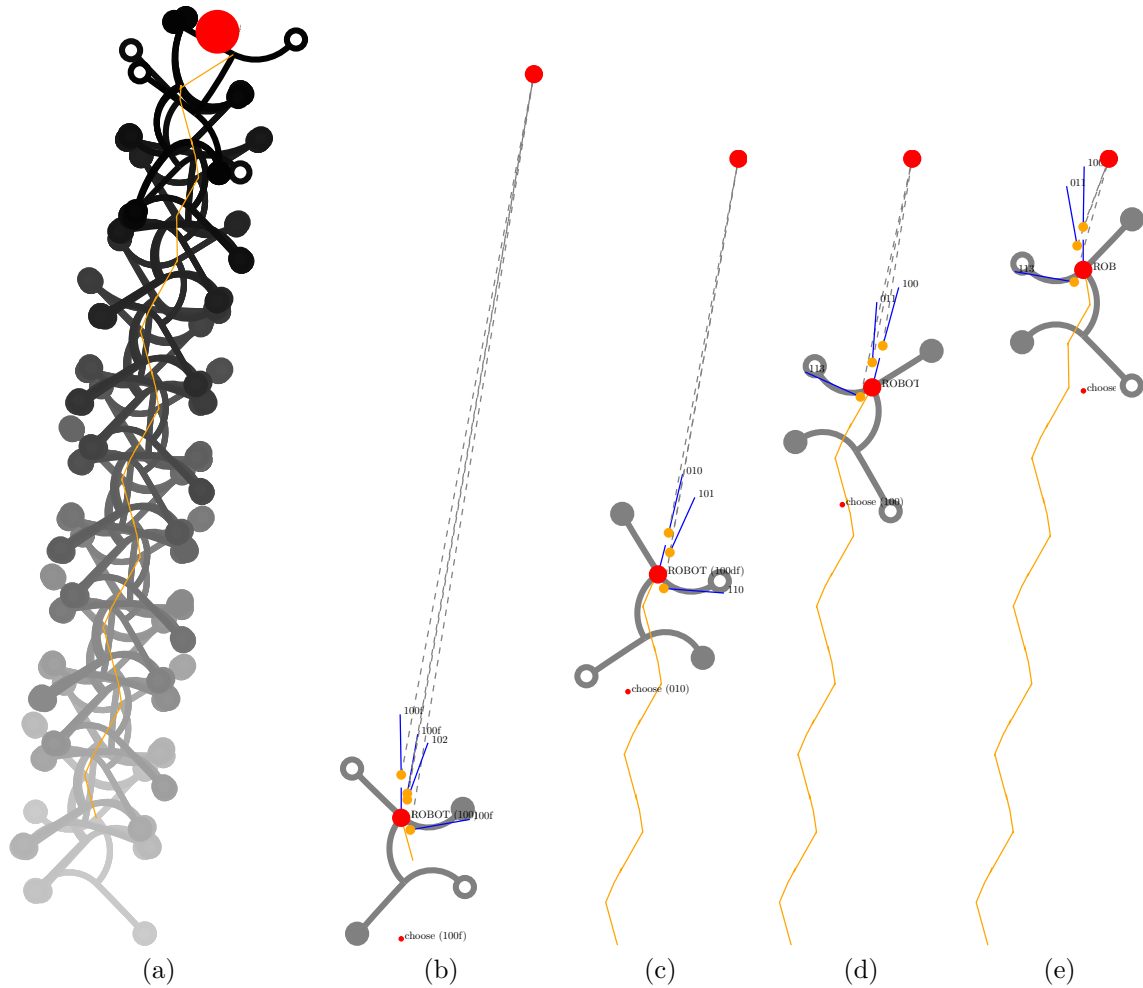
## 2.3 Search Tree with weights



## 2.4 Simulation Results Curve



## 2.5 Simulation Results Straight



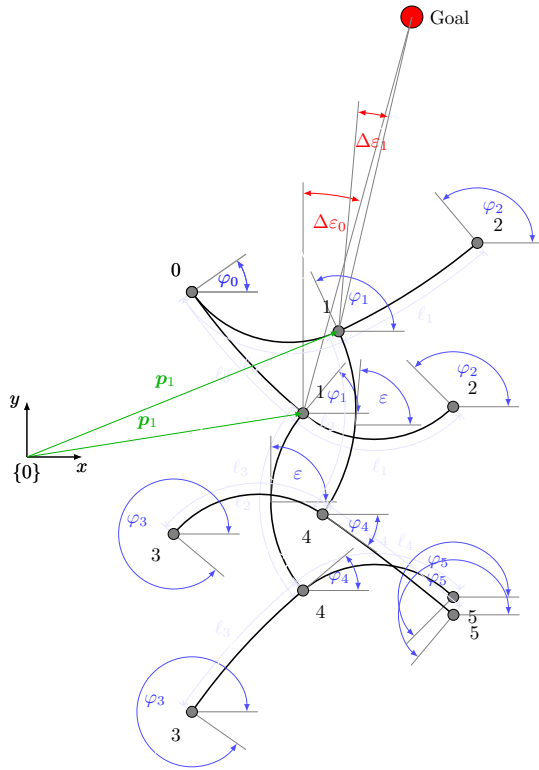
## 2.6 What happens if Process Noise occurs?

## 2.7 Conclusion

### 3 Path Planning with Analytic Model

### 3.1 Problem Statement

- Angenommen die Konfiguration / Pose des Roboters  $\boldsymbol{\rho} = [\boldsymbol{\alpha}, \mathbf{p}_1, \varepsilon]$  ist vollständig bekannt, wobei  $\boldsymbol{\alpha}$  die Gelenkkoordinaten / Biegewinkel der einzelnen Glieder sind,  $\mathbf{p}_1$  die Position des vorderen Torsoendes und  $\varepsilon$  die Orientierung des Roboters. Siehe Bild:



- Für die Pfadplanung, wäre eine Funktion hilfreich, die zu einer gegebenen Wunschkrehung  $\Delta\epsilon$ , eine entsprechende Abfolge von Roboter-Konfigurationen / Posen ausgibt, sodass sich der Roboter entsprechend dreht.
- So könnte zB die Richtung des Roboters so justiert werden, dass er sich auf ein gegebenes Ziel zu bewegt.
- Für den geraden Gang ist eine analytische Funktion bekannt, die die Geschwindigkeit des Roboters einstellt. Geschwindigkeit im Sinne von Schrittweite, bzw. Vorschub pro Zyklus:

$$\alpha = \begin{bmatrix} 45 - \frac{x_1}{2} \\ 45 + \frac{x_1}{2} \\ x_1 \\ 45 - \frac{x_1}{2} \\ 45 + \frac{x_1}{2} \end{bmatrix} \quad (5)$$

Die Schrittweite ist hier als  $x_1$  beschrieben.

### 3.2 Approach: Guess structure for a analytic model for walking curves

- Src can be found: `analytic_model.py`
- Model:  
 $x_1$  beschreibt hier die Schrittweite  
 $x_2$  das Maß der Drehung.

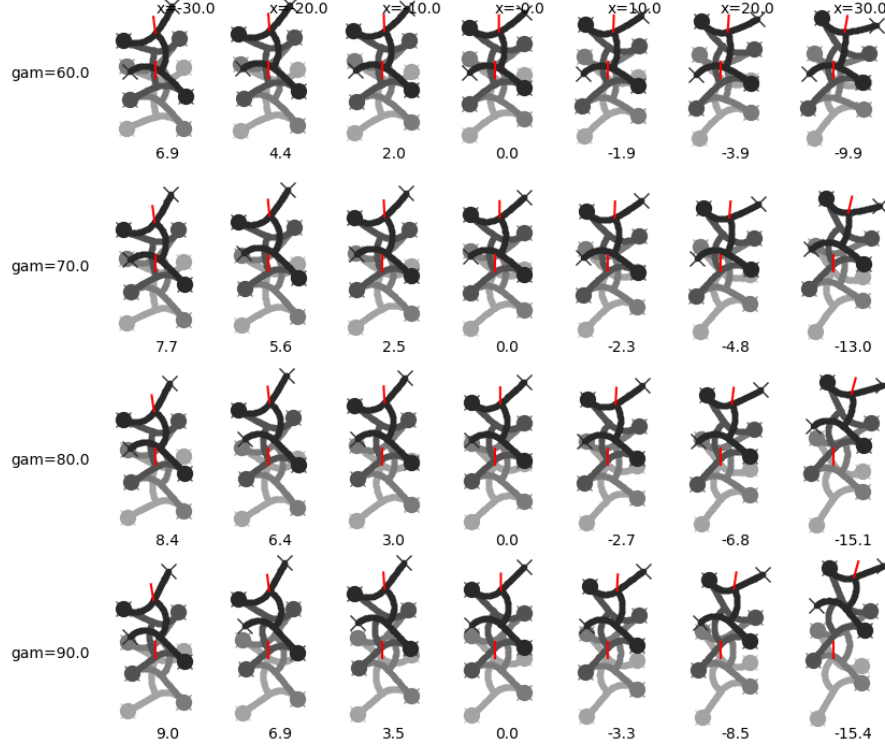
$$\alpha = \begin{bmatrix} 45 - \frac{x_1}{2} \\ 45 + \frac{x_1}{2} \\ x_1 + x_2 \\ 45 - \frac{x_1}{2} \\ 45 + \frac{x_1}{2} \end{bmatrix} \quad (6)$$



- Method:

Simulate for different  $x_1$  and  $x_2$  (in der Abbildung unten ist  $x_1 = \text{gam}$  und  $x_2 = x$ )

- Results für 2 Zyklen:



- Observations:

- Es funktioniert. Der Roboter läuft eine Kurve.
- Kurve ist unsymmetrisch. Rechts klappt besser als links.
- Startpose ist besser für Rechtskurve geeignet.
- Noch nichts über die innere SPannung des Roboters herausgefunden

### 3.3 Approach: Find a reasonable structure

- Src can be found: `analytic_model_2.py`
- Orientierung der Füße soll konstant bleiben:

$$\varphi_0 = \varepsilon + \frac{\alpha_2}{2} - \alpha_0 \quad (7)$$

Da im vorhergegangenen Versuch die asymmetrischen Aktuierung des Torsos schon zu guten Ergebnissen geführt hat, soll dieses Modell beibehalten werden. Allerdings in einer leicht variierten Form.  $x_2$  ist nun ein relatives Maß für die Drehung:

$$\alpha_2 = x_1 + x_2|x_1| \quad (8)$$

Es muss also  $\alpha_0$  so gewählt werden, dass  $\varphi_0$  möglichst unabhängig von  $x_i$  wird. Deshalb wird ein noch unbekannter Term  $x_3$  hinzugefügt. Damit ergibt sich der Biegewinkel des Beines:

$$\alpha_0 = 45 + \frac{x_1}{2} + x_3. \quad (9)$$

Für die Orientierung des Fußes bedeutet das:

$$\varphi_0 = \varepsilon + \frac{x_1 + x_2|x_1|}{2} - \left(45 + \frac{x_1}{2} + x_3\right) \quad (10)$$

$$= \varepsilon - 45 + \frac{x_2|x_1|}{2} - x_3 \quad (11)$$

$$(12)$$

Es wird **angenommen**, dass die Orientierung des Roboters mit der Schrittweite linear wächst (i.e. Der Roboter dreht sich ein wenig zwischen seinen Extremposen):

$$\varepsilon = c_1 x_1 + \varepsilon_0 \quad (13)$$

Mit konstantem Orientierungswinkel  $\varphi = \varphi_0$  ergibt sich somit:

$$\varphi_0 = c_1 x_1 + \varepsilon_0 - 45 + \frac{x_2|x_1|}{2} - x_3 \quad (14)$$

$$x_3 = c_1 x_1 + \frac{x_2|x_1|}{2} + c \quad (15)$$

Unter der **Annahme**, dass  $\varphi_0 \approx \varepsilon_0 - 45$  ist, ergibt sich  $c \approx 0$ . Das meint, die Orientierung ändert sich nur minimal. entspricht also im Wesentlichen der Ausgangskonfiguration. Weiterhin wird **angenommen**, dass für einen fixierter Fuß der Term  $c_1 x_1 \approx 0$  vernachlässigbar ist. Somit ergibt sich für den Biegewinkel des fixierten vorderen linken Beins:

$$\alpha_{0,f} = 45 - \frac{x_1}{2} + \frac{1}{2}x_2|x_1| \quad (16)$$

Wenn das Bein nicht fixiert ist, kann es beliebige Orientierung annehmen. Hierfür wird **angenommen**, dass sich die Drehung des Körpers erst in der nicht fixierten Phase eines Beines in dessen Orientierung auswirkt. Deshalb, wird der Term  $c_1 x_1$  in dieser Phase aktiv. Weiterhin wird **angenommen**, dass  $c_1 = x_2$ . Damit ergibt sich für einen nicht fixierten Fuß:

$$\alpha_{0,\bar{f}} = 45 - \frac{x_1}{2} + x_2 x_1 \quad (17)$$

- Das resultierende Modell sieht so aus:

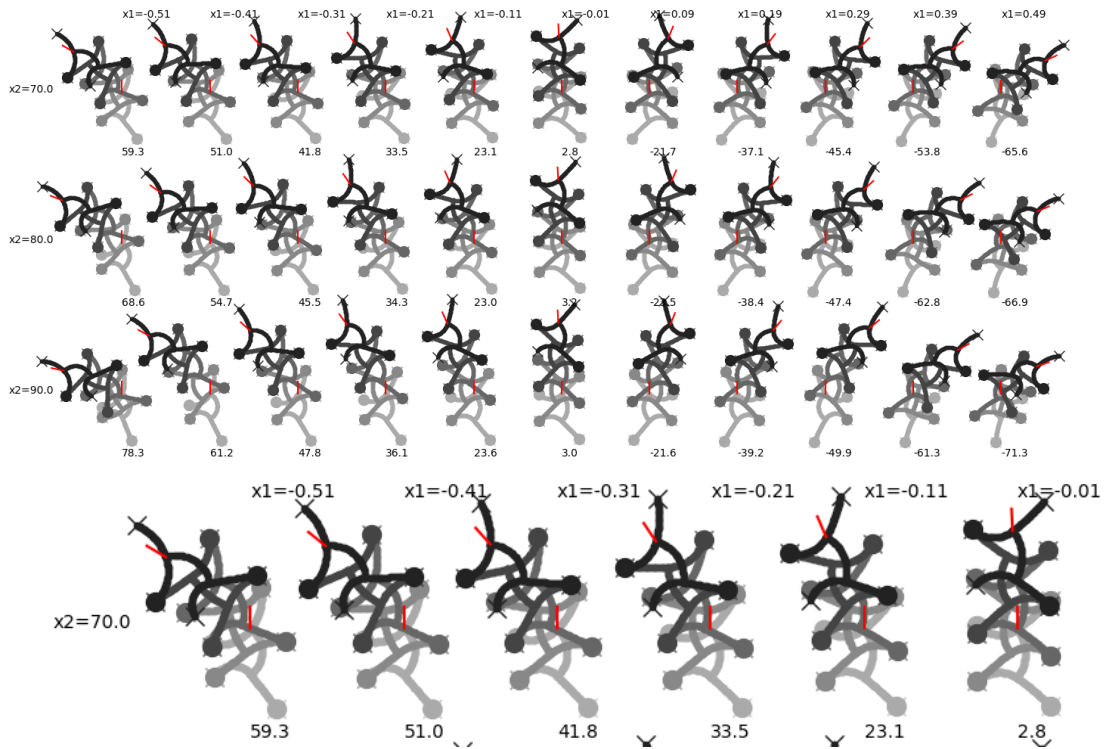
$$\boldsymbol{\alpha} = \begin{bmatrix} 45 - \frac{x_1}{2} + f_0 \frac{1}{2}|x_1|x_2 + \bar{f}_0 x_1 x_2 \\ 45 + \frac{x_1}{2} + f_1 \frac{1}{2}|x_1|x_2 + \bar{f}_1 x_1 x_2 \\ x_1 + |x_1|x_2 \\ 45 - \frac{x_1}{2} + f_2 \frac{1}{2}|x_1|x_2 + \bar{f}_2 x_1 x_2 \\ 45 + \frac{x_1}{2} + f_3 \frac{1}{2}|x_1|x_2 + \bar{f}_3 x_1 x_2 \end{bmatrix} \quad (18)$$

Wobei  $f_i$  den Zustand des Fußes beschreibt:

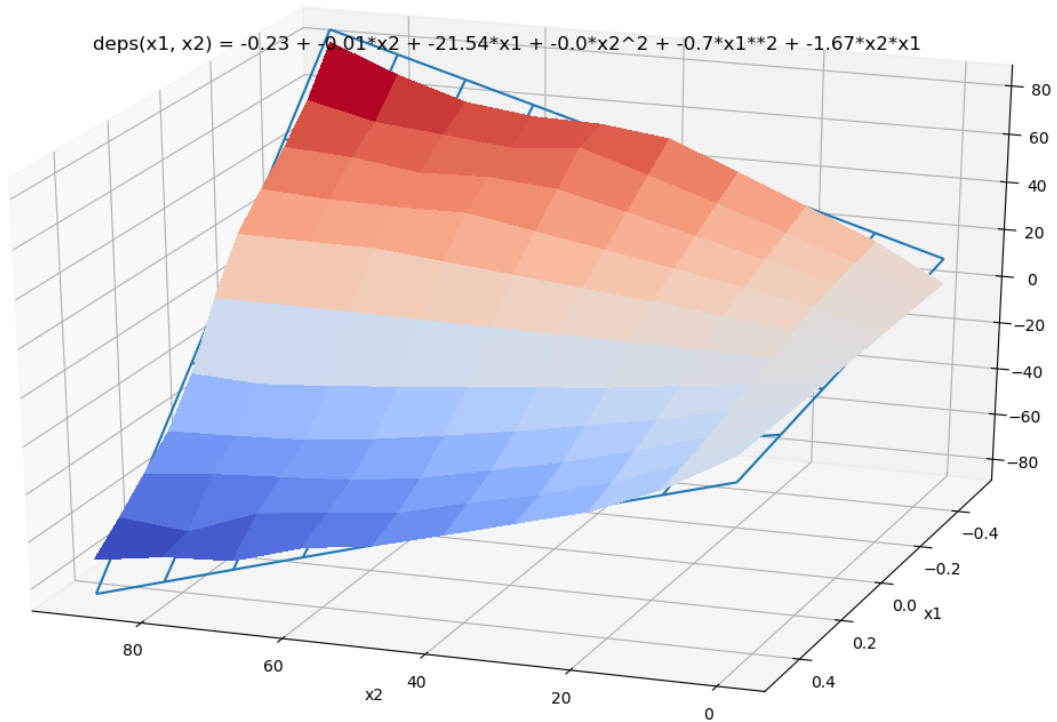
$$f_i = \begin{cases} 1 & \text{if} & \text{foot fixed} \\ 0 & \text{else} \end{cases} \quad (19)$$

$$\bar{f}_i = \begin{cases} 0 & \text{if} & \text{foot fixed} \\ 1 & \text{else} \end{cases} \quad (20)$$

- Results



Delta Epsilon:  $\frac{\Delta \varepsilon}{cycle}(x_1, x_2) = f$



### 3.4 Approach: Optimize Extra leg bending Angle for given extra torso bending

- Src can be found: `analytic_model_3.py`
- Nun soll untersucht werden, welche Extra Leg Bending Angle die innere Spannung des Roboters minimiert.

- Model:

$$\alpha = \begin{bmatrix} 45 - \frac{x_1}{2} + \bar{f}_0 x_3 + f_0 x_4 \\ 45 + \frac{x_1}{2} + f_1 x_3 + f_1 x_4 \\ x_1 |x_2| \\ 45 - \frac{x_1}{2} + f_2 x_4 + f_3 x_3 \\ 45 + \frac{x_1}{2} + f_3 x_4 + f_4 x_3 \end{bmatrix} \quad (21)$$

- Annahme:

Die Extra Biegung  $x_3$  für freie Beine und die Extra Biegung  $x_4$  für fixierte Beine sind abhängig von der Extra Biegung  $x_2$  für den Torso.

Hinter- und Vorderbeine sind nicht symmetrisch, aber kreuzweise symmetrisch: Die Extrabiegung für ein **nicht fixiertes Vorderbein** entspricht der Extrabiegung eines **fixierten Hinterbeins** und andersherum.

- Methode:

Für gegebenes Extra Torso Bending  $x_2$  und gegebenene Torso Biegung  $x_1$  minimiere die Innere Spannung über den Gang mit  $n$  Zyklen aufsummiert:

Gegeben:  $x_1$  Torsobiegung  
 $x_2$  Extra Torsobiegung  
 Gesucht:  $x_3$  Extra Beinbiegung fixiert vorn  
 $x_4$  Extra Beinbiegung fixiert hinten

$$cost(x) = \sum gait(x).stress \quad (22)$$

- Results:

