3.9.1.5 Inverse kinematics

After the desired leg position is calculated, it is necessary to calculate the required leg joint angles to reach that position. Therefore it is necessary to determine the necessary joint angles to reach a given robot relative target position. This is called inverse kinematics problem.

In general the inverse kinematics problem is a set of non-linear equations, which can often be solved numerically only. In the given case it is possible to derive a analytical closed form solution for the inverse kinematics for one leg of the robot.

Forward kinematics solution. First a solution to the forward kinematics problem is given. This is used in solving the far more difficult inverse kinematics problem.

The forward kinematics problem is the calculation of the resulting foot position for a given set of joint angles.

The foot position relative to the shoulder joint (x, y, z) can be determined using a coordinate transformation. The origin of the local foot coordinate system is transformed into a coordinate system which origin is the shoulder joint.

In the following a simplified model of the robot's leg is applied in which this transformation is composed of the following sub-transformations:

1. clockwise rotation about the y-axis by joint angle q_1
2. counterclockwise rotation about the x-axis by joint angle q_2
3. translation along the negative z-axis by upper limb length l_1
4. clockwise rotation about the y-axis by joint angle q_3
5. translation along the negative z-axis by lower limb length l_2

In homogeneous coordinates this transformation can be described as concatenation of transformation matrices:

$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = Rot_y(-q_1)Rot_x(q_2)Trans\begin{pmatrix} 0 \\ 0 \\ -l_1 \end{pmatrix} Rot_y(-q_3)Trans\begin{pmatrix} 0 \\ 0 \\ -l_2 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{3.44}$$

$Rot_{x/y}(\alpha)$ means a counterclockwise rotation around the x/y-axis of angle α and $Trans\begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}$ a translation of the vector (t_x, t_y, t_z).

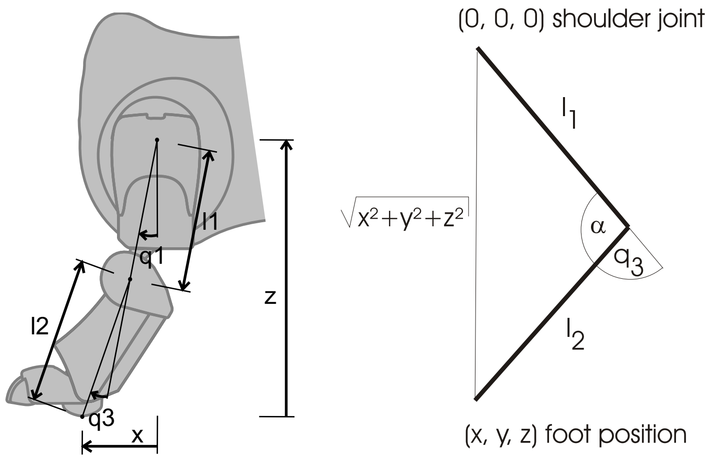


Figure 3.46: leg side view, calculation of knee joint q_3 via law of cosine

This is equivalent to:

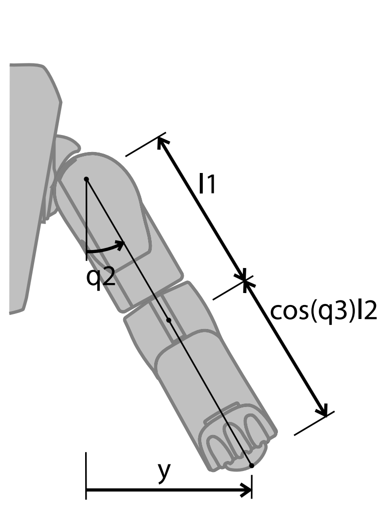
$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(q_1) & 0 & -\sin(q_1) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(q_1) & 0 & \cos(q_1) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(q_2) & -\sin(q_2) & 0 \\ 0 & \sin(q_2) & \cos(q_2) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$
$$\begin{pmatrix} \cos(q_3) & 0 & -\sin(q_3) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(q_3) & 0 & \cos(q_3) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -l_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{3.45}$$

Matrix multiplication results in

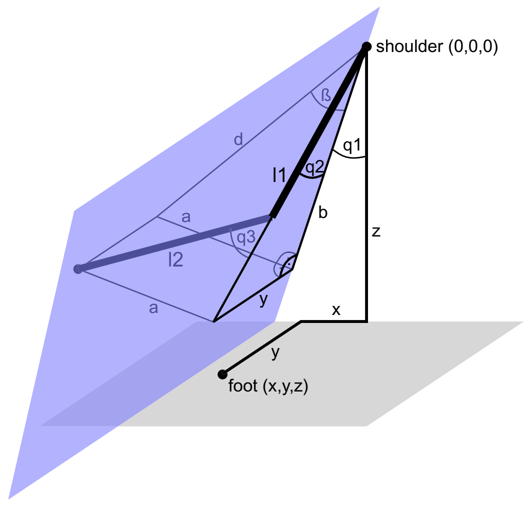
$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(q_1)\sin(q_3)l_2 + \sin(q_1)\cos(q_2)\cos(q_3)l_2 + \sin(q_1)\cos(q_2)l_1 \\ \sin(q_2)l_1 + \sin(q_2)\cos(q_3)l_2 \\ \sin(q_1)\sin(q_3)l_2 - \cos(q_1)\cos(q_2)\cos(q_3)l_2 - \cos(q_1)\cos(q_2)l_1 \\ 1 \end{pmatrix}. \tag{3.46}$$

This equation (and all of the following) is correct only for the left fore leg. But due to the symmetry of the coordinate systems of the four legs, only the signs differ in the calculation for the other legs. Thus when calculating the position of a right foot the y-coordinate has to be negated, for a hind foot the x-coordinate. Furthermore the lower limb length l_2 is slightly larger for the hind legs.

Calculation of knee joint angle q_3. To solve the inverse kinematics problem first of all the knee joint angle q_3 is calculated. As the knee joint position determines how far the leg is stretched, the angle can be calculated from the distance of the target position (x, y, z) to the shoulder joint.



a)



b)

Figure 3.47: a) leg front view, calculation of shoulder joint q_2 b) leg model, calculation of shoulder joint q_1 with several helping variables

According to the law of cosine (cf. Fig. 3.46)

$$\cos\alpha = \frac{l_1^2 + l_2^2 - x^2 - y^2 - z^2}{2l_1l_2} \tag{3.47}$$

with upper limb length l_1 and lower limb length l_2.

With

$$|q_3| = |180^\circ - \alpha| = \arccos\frac{x^2 + y^2 + z^2 - l_1^2 - l_2^2}{2l_1l_2} \tag{3.48}$$

the absolute value of the first joint angle is calculated.

The inverse kinematics problem always has two solution, as there are two possible knee positions to reach a given target. These two solutions are selected via the sign of q_3. With respect to the joint limitations the positive value is used due to the larger freedom of movement for positive q_3.

Calculation of shoulder joint q_2. Plugging the result for q_3 into the forward kinematics solution allows determining q_2 easily. According to equation (3.46) (geometrically apparent cf. Fig. 3.47a)

$$\begin{aligned} y &= \sin(q_2)l_1 + \sin(q_2)\cos(q_3)l_2 \\ &= \sin(q_2)\left[l_1 + \cos(q_3)l_2\right]. \end{aligned} \tag{3.49}$$

Consequently

$$q_2 = \arcsin\left(\frac{y}{l_2\cos(q_3) + l_1}\right). \tag{3.50}$$

Since $|q_2| < 90^\circ$ determination of q_2 via arc sine is satisfactory.

Calculation of shoulder joint q_1. Finally the joint angle q_1 can be calculated. According to equation (3.46) (cf. Fig. 3.47b)

$$\begin{aligned} x &= \cos(q_1)\sin(q_3)l_2 + \sin(q_1)\cos(q_2)\cos(q_3)l_2 + \sin(q_1)\cos(q_2)l_1 \\ &= \cos(q_1)\sin(q_3)l_2 + \sin(q_1)\left[\cos(q_2)\cos(q_3)l_2 + \cos(q_2)l_1\right]. \end{aligned} \tag{3.51}$$

When defining

$$a := \sin(q_3)l_2, \tag{3.52}$$

$$b := -\cos(q_2)\cos(q_3)l_2 - \cos(q_2)l_1 \tag{3.53}$$

and

$$\beta := \arctan\left(\frac{a}{b}\right), \tag{3.54}$$

$$d := \sqrt{a^2+b^2} \quad \left(= \frac{b}{\sin(\beta)}\right), \tag{3.55}$$

so that

$$a = d\cos(\beta), \qquad\qquad b = d\sin(\beta), \tag{3.56}$$

equation (3.51) simplifies to

$$\begin{aligned} x &= \cos(q_1)a - \sin(q_1)b \\ &= d\left[\cos(q_1)\cos(\beta) - \sin(q_1)\sin(\beta)\right] \end{aligned} \tag{3.57}$$

which can be transformed to

$$x = d\cos(q_1+\beta). \tag{3.58}$$

Hence

$$|q_1+\beta| = \arccos\left(\frac{x}{d}\right). \tag{3.59}$$

The sign of $q_1+\beta$ can be obtained by checking the z-component of equation (3.46). As in equations (3.51)-(3.58) this results in:

$$\begin{aligned} z &= \sin(q_1)\sin(q_3)l_2 - \cos(q_1)\cos(q_2)\cos(q_3)l_2 - \cos(q_1)\cos(q_2)l_1 \\ &= \sin(q_1)\sin(q_3)l_2 - \cos(q_1)\left[\cos(q_2)\cos(q_3)l_2 + \cos(q_2)l_1\right] \\ &= \sin(q_1)a + \cos(q_1)b \\ &= d\left[\sin(q_1)\cos(\beta) + \cos(q_1)sin(\beta)\right] \\ &= d\sin(q_1+\beta). \end{aligned} \tag{3.60}$$

As $d > 0$, $q_1+\beta$ is of the same sign as z. Hence if $z < 0$ the calculated value of $q_1+\beta$ has to be negated.

After subtraction of β the last joint angle q_1 is computed.

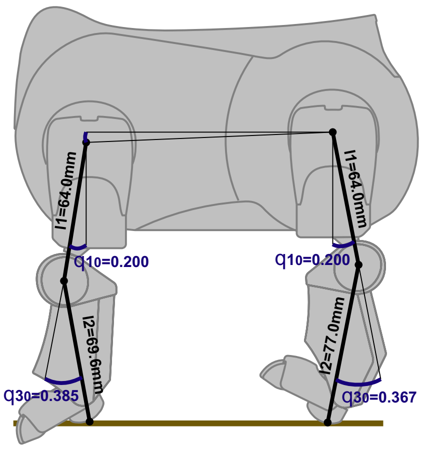


Figure 3.48: Sending an angle of 0 degrees to all joint motors results in vertical legs but not in angles of 0 degrees between the joints.

Anatomy Corrections. The above calculations assume that a vertically stretched leg results in an arc of 0 degrees at all joints. Whereas this might be true for a wire netting model, it is incorrect for an Aibo. Adding constant offsets as in Fig. 3.48 compensates for this difference.

3.9.1.6 Gait Evolution

The task to find a fast and effective parameter set describing the walk becomes more and more difficult with an increasing number of parameters. Finding the fastest possible walk using a walking engine with n parameters means to find the representation of the fastest walk in an n-dimensional search space. For a large number n this is not feasible by trying different parameter combinations by hand. Two different approaches to optimize the gait parameters were used in the GermanTeam:

Localization-Based Fitness. The major problem in autonomous learning approaches for walking is to find a way to measure the speed of the resulting walk reliably. This approach uses the self-locator of the robot to determine the walking speed.

First the robot stands on the field and localizes itself. After being well localized, the robot walks for a fixed period of time into a specified direction with maximum speed. Then it stops and localizes it self again. By calculating the distance between the starting and ending point on the field, the robot can calculate the speed of the walk. To increase the reliability of this kind of measurement, we let the robot walk three times with the same parameter set and take the average of the three measured speeds.

For the optimization process we used a simple $(1+1)$ Evolution Strategy [5]. An individual is represented by a set of walking parameters. Its fitness is discovered by the corresponding speed of the robot while walking on the field. Since this approach does not need any external hardware it is easily possible to adapt the walking parameter set to the actual environmental conditions.