University of Twente

EEMCS / Electrical Engineering
Robotics and *Mechatronics*

Design of a Dynamic Free Gait-generator for a Quadruped Walking Robot

M. (Michiel) van der Coelen

BSc Report

Committee:
Prof.dr.ir. S. Stramigioli
D. Dresscher, MSc
Ir. G.A. Folkertsma

October 2013

Report nr. 026RAM2013
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Summary

Quadruped robots are preferable over their wheeled counterparts because of their increased maneuverability over inconsistent terrain. A quadruped robot is being designed as part of the ROSE project. The project described in this text focuses on the design of a gait-generator for that robot. The goal of this gait-generator is to provide the robot with timing for the transfer of its feet and target locations where the feet should be placed. This has to be done while maintaining static stability of the robot at all times. The basis for the gait-generation is the velocity vector with which the robots center of mass (COM) moves.

Background knowledge is provided on homogeneous matrices, support-patterns, stability, and a specific gait called the wave-gait. Distinctive values used in the gait-generator include the kinematic margins of the legs, the fore and back longitudinal absolute stability margins, the standard fore foothold adequacy, and the transfer distance of the robots center of mass. The methods for calculating these values are explained in detail.

The design of the gait-generator operates on a small cycle which is contained in a state-machine. The three states are: 1. waiting until a leg can be lifted, 2. calculating the target position for the foot of that leg, and 3. waiting for the transfer of the foot to complete. The calculation of the target position is done by evaluating a 15 by 15 square of evenly spaced points around the current foot position. A set of limiting areas is constructed, based on the distinctive values mentioned earlier, in order to enforce continued stability and speed of the robot. The number of candidate footholds is reduced by overlaying these areas. From the remaining points an optimum target position for the transferring foot is then selected.

The gait-generator has been designed to allow future expansions to enable omni-directional movement over inconsistent terrain. The design is implemented in 20-sim. Simulations of the gait-generator model show a stable gait is generated for backwards and forwards motion. The gait-generator is also capable of handling limited crab-angles. A list of recommendations for improvement and further development is also presented.

Contents

1 Introduction

1.1 Context

Legged robots can be more manoeuvrable than their wheeled counterparts when it comes to locomotion over difficult terrain. A major component of legged locomotion is remaining stable throughout the walking motion. Robots with any number of legs are possible. For instance, hexapod robots have been built that are capable of stable locomotion (McGhee and Iswandhi, 1979; Erden and Leblebicioğlu, 2008; Inagaki et al., 2006). Minimizing the number of legs reduces mechanical complexity and weight. A minimum of four legs is required to retain the ability for statically stable locomotion (McGhee and Frank, 1968). This reduction comes at the cost of increased complexity of the gait-generator that controls the locomotion.

As part of the ROSE project (Dresscher and Stramigioli, 2012), a quadruped robot will be designed and realized. This robot will have to traverse difficult terrain (dikes) and carry sensitive measurement equipment. In the project that is the subject of this text, a design is made and implemented for a gait-generator that is intended to be used on the ROSE-robot. The main function of this gait-generator is to generate coordinates for the robots feet in order to provide mobility while maintaining static stability of the robot.

1.2 Problem statement

The robot will use a form of impedance control (Arevalo and Garcia, 2012; Yoneda et al., 1994) for its joints. The movement will be controlled by virtually pushing or pulling on the center of the robots body. The goal of the gait-generator is to provide timing and end-positions for the movement of the feet, which will enable the body to execute its intended movement without losing stability of the robot.

1.3 Requirements

The final version of the gait-generator will have to be able to respond to difficult terrain, misplaced feet, and obstacles that will be encountered. The scope of this project however, is reduced to flat terrain without obstacles. The following requirements are set:

- The provided gait maintains static stability of the robot.
- The provided gait must be capable of maintaining a constant speed of the center of the robots mainbody.
- The center of the mainbody must be able to move in any direction.
- The algorithm should be expandable to include terrain adaptability.
- The algorithm should be expandable to include obstacles (forbidden zones).

The gait-generator has to be compatible with the dynamic robot-model that is developed in 20-sim (Controllab Products, 2008). To achieve this compatibility, the gait-generator is also developed in 20-sim.

1.4 Related work

Many of the basic measures and definitions required for evaluating gaits are covered in (McGhee and Frank, 1968) and (McGhee and Iswandhi, 1979). To facilitate the required motion of the quadruped, 5 options for gait-generation were considered:

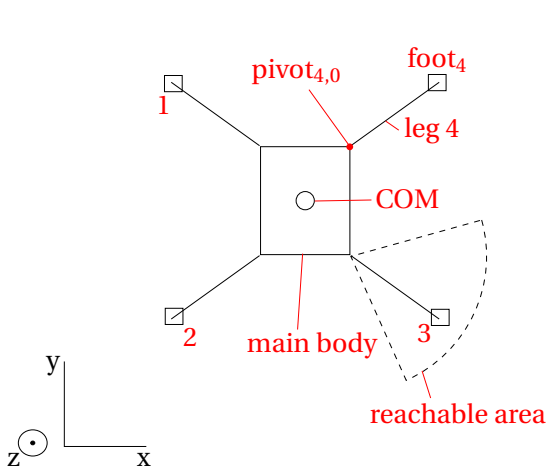
1. An algorithm where a pre-generated tree structure containing leg movement sequences is searched for a suitable candidate at every step (Pal and Jayarajan, 1991). This algorithm does not considerer the terrain but does provide statically stable movement. To facilitate omni-directional movement, the tree structure would become too large so this concept was not explored further here.
2. A decentralized form of gait-generation where a standard wave gait is realized with a set of oscillators which then have their energy modulated based on the required speed and other environmental factors (Inagaki et al., 2006). This system is focussed on decentralizing the gait-generation in order to create changeability of legs, ease of maintenance and fault-tolerance. The system was tested using robots with 6, 8 and 10 legs. It has not been tested for quadrupeds.
3. A circular wave-gait (Hirose et al., 1986), where the robot moves with constant speed along an arc with a variable radius. While this does provide the capability of constant speeds and omni-directional movement, it does not include the possibility of excluding certain areas for foot-placement and was mostly designed with flat terrain in mind.
4. A combination of a primary fixed sequence gait and a free dynamic secondary gait (Bai et al., 1999). Using this algorithm, a robot would move with its primary gait until it reaches a deadlock and then switch to the secondary gait until the robot has reached a state where the primary gait can be continued. This switching of gaits introduces a stuttering motion since the robot is stopped at each switch which makes it unfavourable for carrying measurement equipment.
5. A free crab gait (Estremera and de Santos, 2005). With this gait, the orientation of the body is independent of the direction of motion. This design was made specifically with irregular terrain and forbidden zones in mind and is focussed on maintaining constant speeds. Real-world tests of this system were done with the SIL04 robot (De Santos et al., 2003).

	1	2	3	4	5
Quadruped	yes	no	yes	yes	yes
Constant speed	yes	yes	yes	no	yes
Straight motion	yes	no	no	yes	yes
Omni-directional motion	no	no	yes	yes	yes
Terrain adaptability	none	slight	slight	none	good
Forbidden zones	no	no	no	yes	yes

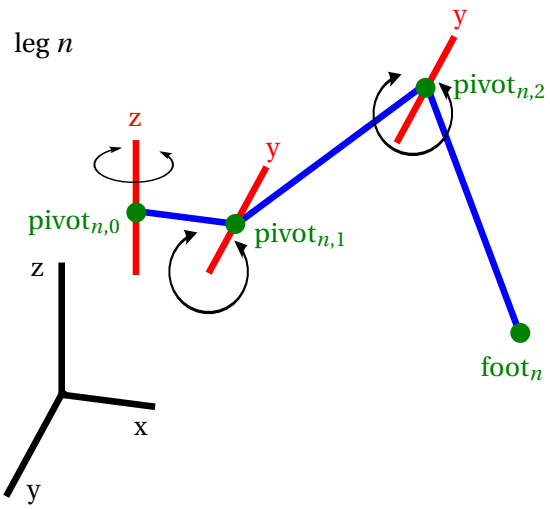
Table 1.1: Summary of the 5 gait-generation options that were considered.

A summary of the gait-generation options listed above, is shown in table 1.1. Based on these properties the choice was made to base the design for the gait-generator on the free crab-gait option.

1.4.1 Robot-model



(a) A simple top-down perspective of the provided robot-model. $\text{pivot}_{4,0}$ is where leg 4 with a setup shown in Figure 1.1b is connected to the main body. COM refers to the center of mass of the robot.



(b) The leg-setup of the provided robot-model. There are three degrees of freedom. n denotes the leg-number [1..4].

Figure 1.1

Figure 1.1a shows a top-down view of the provided robot-model. It uses a form of impedance control for its joints (Arevalo and Garcia, 2012; Yoneda et al., 1994). The legs have three degrees of freedom as shown in Figure 1.1b. The three pivot points of the leg will all have a limit once they're physically realized. This causes a limited reachable area for the foot as shown in Figure 1.1a. The coordinate system shown, will be used throughout this text.

1.5 abbreviations

Table 1.2 lists some of the abbreviations that are commonly used. A detailed explanation of these can be found in chapter 3.

Abbreviation	Meaning
COM	Center Of Mass of the robot.
LT	Transfer Leg. Leg for which the foothold is calculated.
NLT	Next Transfer Leg. Leg that is likely to be transferred next.
ASM	Absolute stability margin.
LASM	Longitudinal Stability Margin.
LASMB	Longitudinal Stability Margin (Backwards)
LASMF	Longitudinal Stability Margin (Forwards)
KM	Kinematic Margin
TD	Transfer Distance of the COM.
ELD	Effective Lifting Distance.
ESD	Effective Stability Distance.
SFFA	Standard Fore Foothold Adequacy.

Table 1.2: Common abbreviations

1.6 Report organization

The outline of this report is as follows: Chapter 2 provides background knowledge needed to understand later chapters. Chapter 3 contains an analysis of the models and values used in the design. This design is specified in chapter 4. The simulations and their results are detailed in chapter 5 and a conclusion can be found in chapter 6, together with recommendations for further work and improvement suggestions.

2 Background

This chapter covers the general background knowledge and terminology that is used in later chapters. Also included are some notes on the notations used throughout this text.

2.1 General

All calculations in this text take place on the plane perpendicular to the gravity. In the used coordinate system this is the xy-plane, where the gravity is along the negative z-axis.

The projection of the center of mass (COM) of the robot is a single point on the xy-plane. There is also a circular safety margin around the COM with radius ASM_{min}. the significance of ASM_{min} is covered in chapter 3.2. The projection of the COM together with its safety margin is represented in figures as a point with a circle around it, denoted by "COM + ASM_{min}". In this text, unless it is explicitly mentioned otherwise, COM can be taken to mean the projection of the COM onto the xy-plane.

2.2 Indices

As was already mentioned in chapter 1.4.1, n denotes a leg index from 1 through 4 and is often used as a subscript to indicate any of the four legs. for example: foot_n refers to the foot connected to the leg with index n.

LI and LJ are similarly used to specify arbitrary leg-indices, while LT and NLT refer specifically to the currently and next transferring leg respectively. LR, LF, RR, and RF refer to the left-rear, left-front, right-rear and right-front leg respectively. The indices that these four refer to will change based on which direction the robot is moving in, since this is independent to the orientation of the body. Figure 2.1 shows an example of this change.

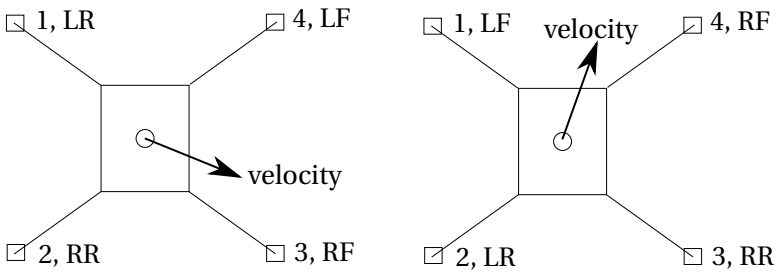


Figure 2.1: An example of the change in referred indices when the direction of movement changes.

For the calculation of the standard fore foothold adequacy, the letters V, W and U are used to refer to the next three legs that will be transferred according to the wave-gait (see section 2.3). The transfer order for the wave-gait is LT, V, W, U and then LT again.

2.3 Wave gait

Previous work has shown that the wave gait has the best stability out of all possible periodic gaits for a legged vehicle (McGhee and Frank, 1968). In this text a clockwise wave-gait leg-order used by (Estremera and de Santos, 2005) is adopted:
Left-Rear leg -> Left-Front leg -> Right-Rear leg -> Right-Front leg.

2.4 Homogeneous matrices

All coordinates used in the calculations are represented with 3D homogeneous matrices. A Homogeneous matrix (H-matrix) can represent a reference frame, or the position and orientation of a rigid body. It is a 4x4 matrix as shown in equation 2.1.

$$\mathbf{H}_P^a = \left[\begin{array}{ccc|c} & \mathbf{R} & & \begin{matrix}P_x\\P_y\\P_z\end{matrix} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \qquad \mathbf{H}_{-P}^a = \left[\begin{array}{ccc|c} & \mathbf{R} & & \begin{matrix}-P_x\\-P_y\\-P_z\end{matrix} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \tag{2.1}$$

Where $\mathbf{H}_P^a$ is the H-matrix describing the position and orientation of a frame P relative to frame a. $\mathbf{R}$ is a 3x3 matrix describing the relative orientation of the frame, and P_x, P_y and P_y represent the coordinates of the position of the frame relative to frame a. $\mathbf{H}_{-P}^a$ has an the same orientation as $\mathbf{H}_P^a$, but its relative position to frame a is inverted as shown in equation 2.1.

When a specific value of a H-matrix is needed, it will be referenced with a row and column index as illustrated by the example in equation 2.2

$$P_x = \mathbf{H}_P^a[1,4] \tag{2.2}$$

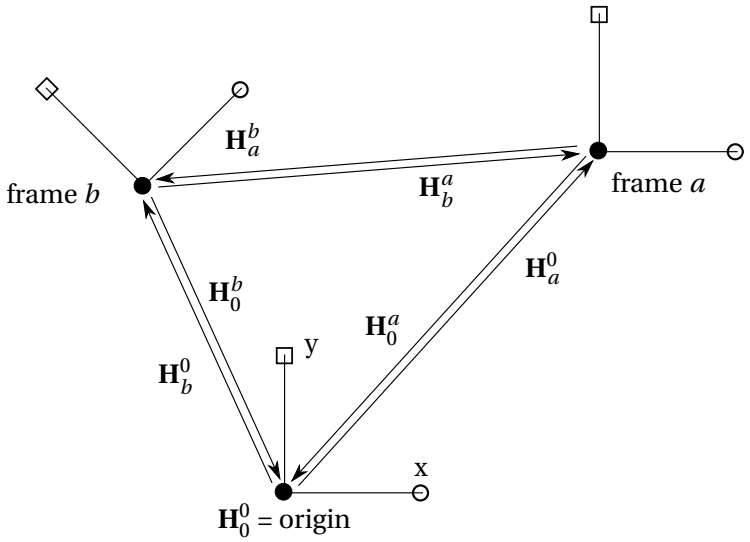


Figure 2.2: An example of the H-matrix notation using example-frames a and b. The arrows indicate the H-matrix used to transform from one frame (superscript) to another (subscript).

The origin of a coordinate system is referred to as frame 0. Figure 2.2 shows two example frames a and b. Transforming from one frame to another is done by multiplying the H-matrices. Equation 2.3 shows the expression for transforming from frame a to frame b:

$$\mathbf{H}_a^b = \mathbf{H}_0^b \cdot \mathbf{H}_a^0 \tag{2.3}$$

This style of notation allows two forms to be used when expressing an inverse H-matrix, as illustrated by equation 2.4

$$\mathbf{H}_a^0 = \left(\mathbf{H}_0^a\right)^{-1} \tag{2.4}$$

Equation 2.5 shows the notation used for the distance of a frame P to the origin.

$$\text{distance from } P \text{ to origin} = |\, \mathbf{H}_P^0 \,| \tag{2.5}$$

2.5 Stability

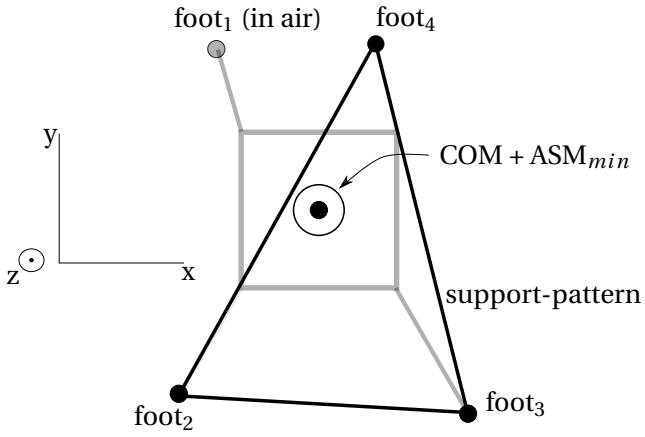


Figure 2.3: The support-pattern. It is the convex hull of all feet that are supporting the robot. When the projection of the COM is inside the support-pattern, the robot is stable. In this example the robot is stable

The convex hull of all feet [1] that are supporting the robot (i.e. are in contact with the ground) is called a support-pattern (Hirose et al., 1986). In order for the robot to be statically stable, the projection of the center of mass (COM) onto the xy-plane must be inside the support pattern at all times (McGhee and Iswandhi, 1979; Hirose et al., 1986).

[1] In this case the circumference of the smallest continuous area that includes all the supporting feet and the lines connecting them

3 Analysis

This chapter will cover the defining variables and values used in the design and the methods that were developed to calculate them. As mentioned in the previous chapter, all calculations take place in the xy-plane.

3.1 Kinematic Margin

The Kinematic margin (KM) refers to the distance the COM can travel in its forward direction until a specific leg is at its physical limit (McGhee and Iswandhi, 1979). It is a scalar value that will be visually represented as a line with a length of KM in the direction opposite to the COM movement and starting at the foot. During movement of the COM the foot of a leg is assumed to stay in the same location while the rest of the leg moves with the COM. The reachable area for a leg is defined as the area where the KM is positive (equation 3.1).

$$KM >= 0 \tag{3.1}$$

The kinematic margin is dependent on the reachable area that is defined by the limits of the leg. The leg setup described in the previous chapter leads to a reachable area for the foot (on the 2D ground plane) in the form of a circular sector. For a foot outside this sector the KM is defined as -1. Three points are used to define the reachable area sector. They are the three corner points as shown in figure 3.1: $\mathbf{H}^0_{\text{pivot}_{n,0}}$, the point where leg n is connected to the robot body, $\mathbf{H}^0_{\text{HL1}}$ and $\mathbf{H}^0_{\text{HL2}}$, two parameters used to further define the limits of the leg.

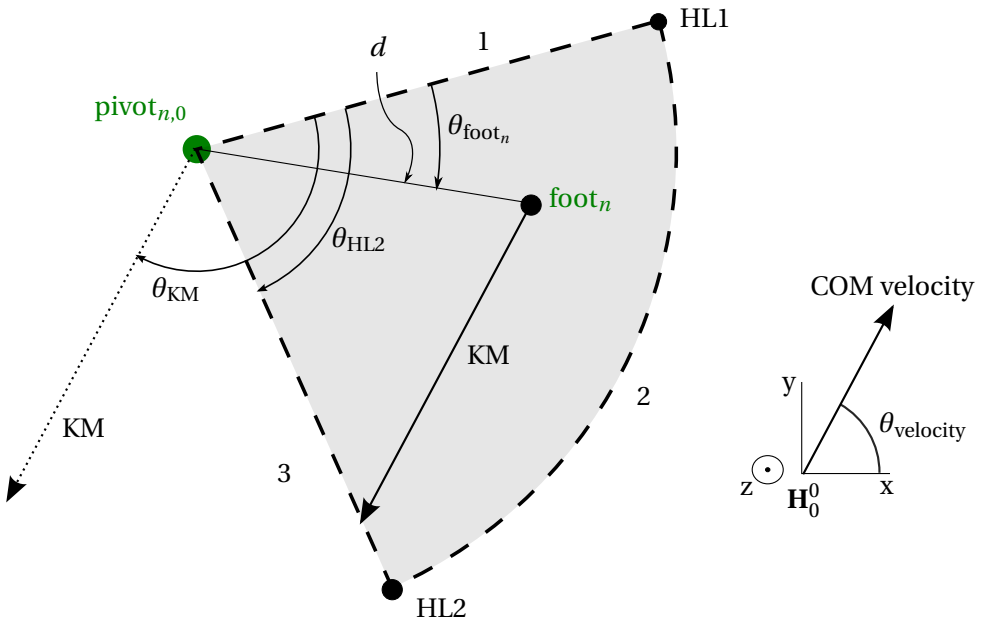


Figure 3.1: Visualization of the Kinematic margin (KM) and the angles that are used to determine which edge of the reachable area will be reached.

Figure 3.1 shows an example case. Numbers 1 2 and 3 indicate the edges of the reachable area (highlighted in gray). the KM-line is in the opposite direction of the COM velocity. θ_{KM} is the angle that the KM-line makes with edge 1. θ_{HL2} is the angle between edge 1 and 3. θ_{foot_n} indicates the angle between the line from pivot$_{n,0}$ to foot$_n$, and edge 1. The angles shown in the figure are considered positive in the direction of their arrows. d is the distance between the pivot and the foot of the leg. θ_{velocity} is the angle of the COM velocity.

the COM moves in the direction of the COM velocity. Which causes the reachable area to translate in the same way. This is modelled by having the foot move in the opposite direction. After KM meters, the foot will transition out of the reachable area.

As mentioned earlier, a foot outside the reachable area has a KM of -1. Therefore the calculation of KM is only required when the foot is inside the reachable area. The first step in calculating KM is making sure the foot is within the reachable area.

$$\begin{aligned} d &\leq \left| \left(\mathbf{H}^0_{\text{pivot}_{n,0}}\right)^{-1} \cdot \mathbf{H}^0_{\text{HL1}} \right| \\ \theta_{\text{foot}_n} &\leq \theta_{\text{HL2}} \end{aligned} \tag{3.2}$$

Equation 3.2 specifies the two conditions that have to be met in order for the foot to be inside the reachable area: The leg cannot extend further than the radius of the reachable area sector and the line from the pivot to the foot (i.e. the leg) must have an angle that makes it lie within the reachable area.

Given that the foot is in a valid position, the next step is to determine which edge is going to be hit. Based on the angles from Figure 3.1, three cases can be distinguished. They are shown in equation 3.3:

$$\text{edge} = \begin{cases} \text{1. line from pivot}_{n,0} \text{ to HL1} & \theta_{\text{KM}} < 0 \\ \text{2. arc from HL1 to HL2} & 0 \leq \theta_{KM} \leq \theta_{HL2} \\ \text{3. line from pivot}_{n,0} \text{ to HL2} & \theta_{\text{KM}} > \theta_{\text{HL2}} \end{cases} \tag{3.3}$$

In case 2 the arc is reached. Case 1 and 3 occur when the sidelines are reached and result in almost identical calculations. To reduce the complexity of the calculations, a new reference frame T is introduced. In this frame, the foot is at the origin and the COM velocity is along the negative x-axis. The construction of this frame is shown in equation 3.4

$$\mathbf{H}^0_T = \begin{bmatrix} cos(-\theta_{\text{velocity}}) & -sin(-\theta_{\text{velocity}}) & 0 & \mathbf{H}^0_{\text{foot}_n}[1,4] \\ sin(-\theta_{\text{velocity}}) & cos(-\theta_{\text{velocity}}) & 0 & \mathbf{H}^0_{\text{foot}_n}[2,4] \\ 0 & 0 & 0 & \mathbf{H}^0_{\text{foot}_n}[3,4] \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

3.1.1 arc-case

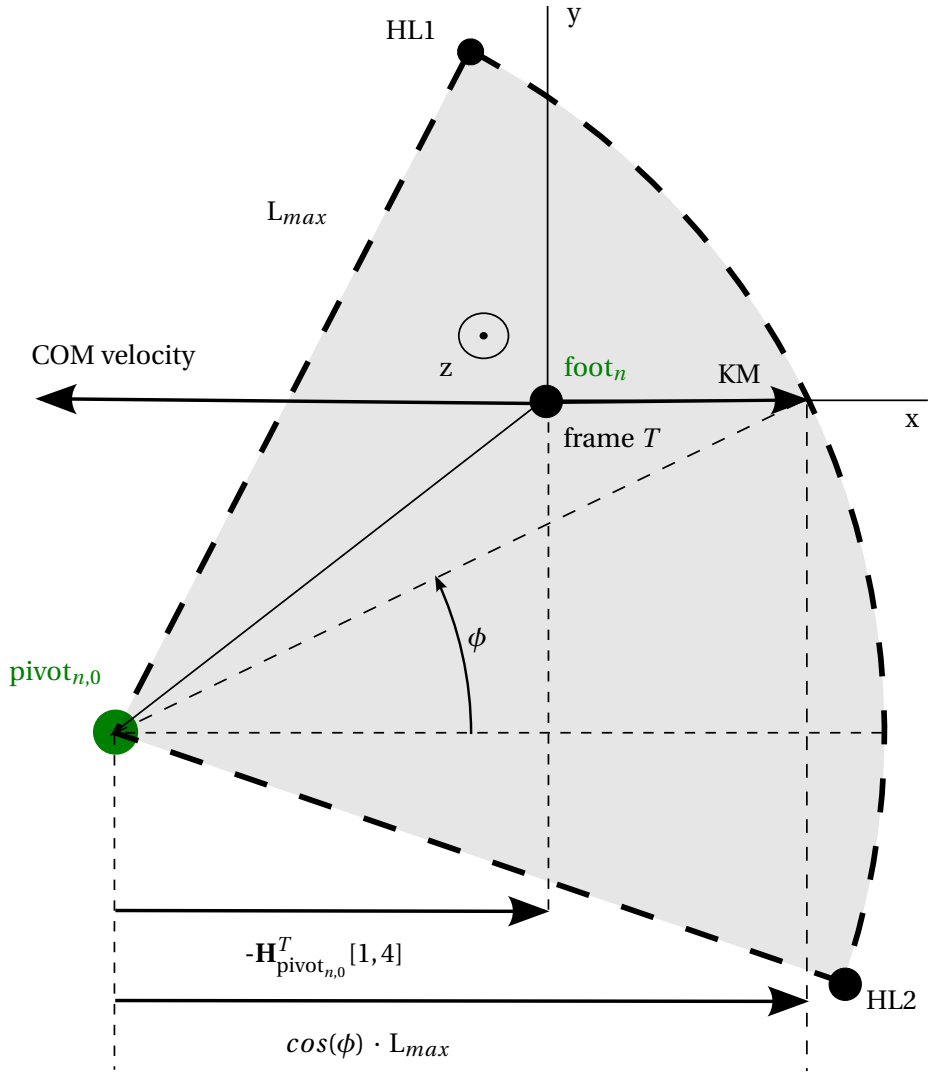


Figure 3.2: The system from Figure 3.1 using frame T as origin. in this frame, the foot is at the origin and the COM velocity is along the negative x-axis

Figure 3.2 shows the point system used in Figure 3.1, rotated so the x and y-axis of frame T are horizontal and vertical. A different direction for the COM velocity is used so the arc-case occurs. L_{max} is the maximum length of the leg. The position of the foot coincides with the origin of frame T and the KM is along its x-axis. Angle ϕ is used in the calculation of KM, which is detailed in equation 3.5.

$$\begin{aligned} L_{max} &= \left| \left(\mathbf{H}^T_{pivot_{n,0}}\right)^{-1} \cdot \mathbf{H}^T_{HL1} \right| \\ \phi &= sin^{-1}\left(\frac{-\mathbf{H}^T_{pivot_{n,0}}[2,4]}{L_{max}}\right) \\ KM &= cos(\phi) \cdot L_{max} + \mathbf{H}^T_{pivot_{n,0}}[1,4] \end{aligned} \tag{3.5}$$

This KM value is the distance from the foot to the arc with radius L_{max} centered at the pivot. Because the leg can only reach outward a maximum of L_{max}, KM can never be larger than the value for the arc-case, regardless of θ_{KM}.

3.1.2 sideline-cases

For the other two cases from equation 3.3, the side-edges of the sector are reached. To find the distance to the edge, frame T is used as origin, see Figure 3.3.

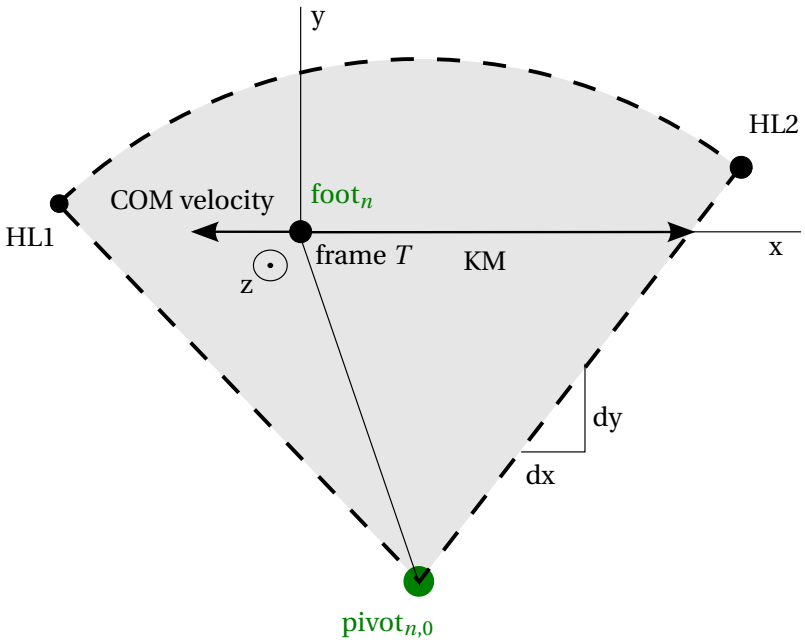


Figure 3.3: The system from Figure 3.1 with frame T as origin. In this frame the foot is at the origin and KM is along the x-axis.

KM is equal to the x-coordinate where the edge crosses the x-axis. dependent on the case, edge 1 or 3 should be considered. The calculation of KM is shown in equation 3.6 for case 3.

$$\begin{aligned} \frac{dx}{dy} &= \frac{\mathbf{H}^T_{\text{HL2}}[1,4] - \mathbf{H}^T_{\text{pivot}_{n.0}}[1,4]}{\mathbf{H}^T_{\text{HL2}}[2,4] - \mathbf{H}^T_{\text{pivot}_{n.0}}[2,4]} \\ KM &= \frac{dx}{dy} \cdot \mathbf{H}^T_{\text{pivot}_{n.0}}[2,4] + \mathbf{H}^T_{\text{pivot}_{n.0}}[1,4] \end{aligned} \tag{3.6}$$

A special case occurs when the arc crosses the x-axis before the side-line does. As mentioned earlier, the KM can never be larger than the one calculated with the arc-case method. An example of such a case is illustrated in Figure 3.4.

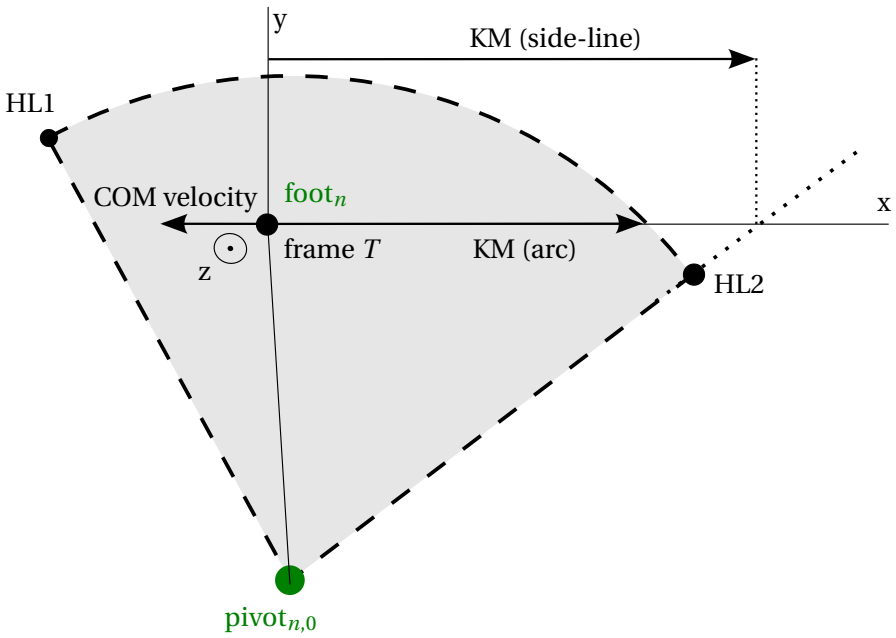


Figure 3.4: A special case for the calculation of KM. The arc crosses the x-axis before the side-line does. KM (side-line) shows the outcome of equation 3.6 and KM (arc) is obtained from equation 3.5.

3.2 Longitudinal Absolute Stability Margin

A key value in determining how stable the robot is, is the Absolute Stability margin (McGhee and Iswandhi, 1979). The *Longitudinal* Absolute Stability Margin LASM is defined by (Estremera and de Santos, 2005). This value indicates the distance that the COM has to travel in its forward direction until an edge of the support-pattern is reached. The forward direction is independent of the body orientation. For ease of use, the distinction is made between the distance to the forward edge (LASMF), and the backward edge (LASMB) of the support pattern.

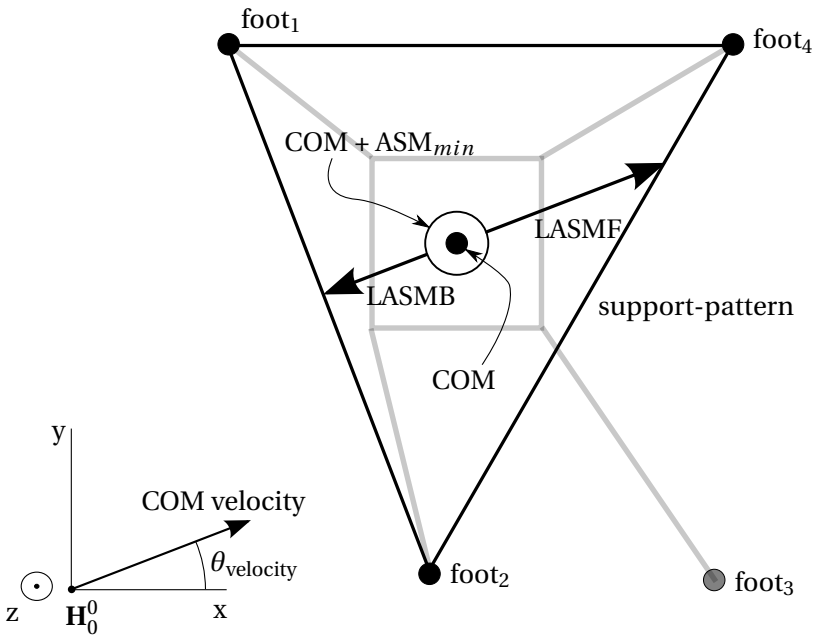


Figure 3.5: visualization of LASMF and LASMB. they represent the distances from the COM to the front and back edges of the support-pattern respectively. The direction of motion is taken into account

Figure 3.5 shows the LASMB and ASMF for a specific direction of movement with foot_3 in the air. foot_1, foot_2, foot_3, and foot_4 indicate the positions of the feet. θ_{velocity} is the angle that the COM velocity makes with the x-axis. When the COM is inside a support-pattern, LASMB will be negative (the COM would have to move backwards to reach the back edge) and LASMF will be positive.

If the legs of the robot are assumed to be massless, then the COM can be taken as the center (of mass) of the main-body. To account for errors due to this assumption, a circular safety zone around the COM is introduced with radius ASM_{min} (Estremera and de Santos, 2005). This zone also helps in preventing other external disturbances to the robot from causing instability.

Similar to the calculation of the kinematic margin, A new reference frame T is introduced. this frame is constructed as show in equation 3.7. The frame has its origin at the COM and is rotated so the COM velocity lies along the x-axis.

$$\mathbf{H}_T^0 = \begin{bmatrix} cos(\theta_{\text{velocity}}) & -sin(\theta_{\text{velocity}}) & 0 & \mathbf{H}_{\text{COM}}^0[1,4] \\ sin(-\theta_{\text{velocity}}) & cos(\theta_{\text{velocity}}) & 0 & \mathbf{H}_{\text{COM}}^0[2,4] \\ 0 & 0 & 1 & \mathbf{H}_{\text{COM}}^0[3,4] \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.7}$$

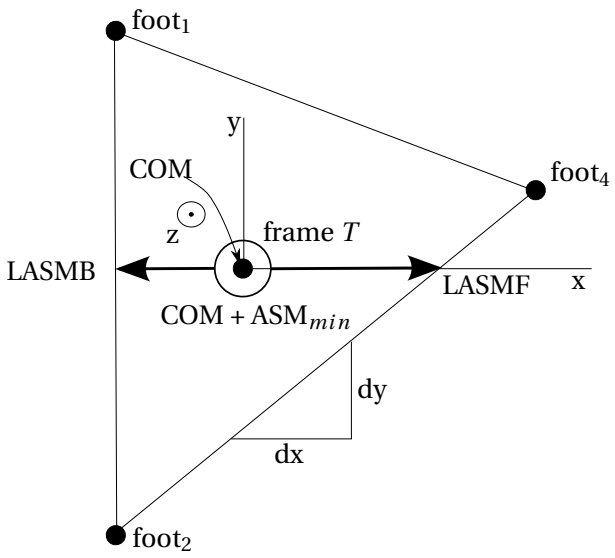


Figure 3.6: the system from Figure 3.5 with frame T as origin. The transformation places the COM at the origin and rotates the other points around it so the LASMF (and the COM velocity) is along the x-axis.

The transformed system is shown in Figure 3.6. LASMB and LASMF can be found as the two x-coordinates where the support-pattern crosses the x-axis. Equation 3.8 and 3.9 calculate the LASMB and LASMF respectively, for the specific case shown in Figure 3.6.

$$\begin{aligned} \frac{dx}{dy} &= \frac{\mathbf{H}^T_{\text{foot}_2}[2,4] - \mathbf{H}^T_{\text{foot}_1}[2,4]}{\mathbf{H}^T_{\text{foot}_2}[1,4] - \mathbf{H}^T_{\text{foot}_1}[1,4]} \\ \text{LASMB} &= -\frac{dx}{dy} \cdot \mathbf{H}^T_{\text{foot}_1}[2,4] + \mathbf{H}^T_{\text{foot}_1}[1,4] + ASM_{min} \end{aligned} \tag{3.8}$$

Note that the LASMB is negative when the COM is inside the support-pattern.

$$\begin{aligned} \frac{dx}{dy} &= \frac{\mathbf{H}^T_{\text{foot}_2}[2,4] - \mathbf{H}^T_{\text{foot}_4}[2,4]}{\mathbf{H}^T_{\text{foot}_2}[1,4] - \mathbf{H}^T_{\text{foot}_4}[1,4]} \\ \text{LASMF} &= -\frac{dx}{dy} \cdot \mathbf{H}^T_{\text{foot}_4}[2,4] + \mathbf{H}^T_{\text{foot}_4}[1,4] - ASM_{min} \end{aligned} \tag{3.9}$$

A special case arises when the COM is outside the support-pattern and the COM velocity is directed in such a way that the COM will never enter the support-pattern. In this case the LASMB is defined as arbitrarily large positive and the LASMF as arbitrarily large negative.

For cases different from figure 3.6, the specific feet in the equations will differ. The correct selection of the feet to be used should be based on the orientation of the body and the COM velocity. The LASMB and LASMF can be calculated for many cases. To distinguish between these, the notation in equation 3.10 is used.

$$\text{LASMF}^{LJ}_{LI}(P) \tag{3.10}$$

Where LJ refers to the leg that has its foot in the air and LI the leg which foot is placed at P.

3.3 Standard Fore Foothold Adequacy

The Standard Fore Foothold Adequacy is a value that is used to measure how well a new foothold for a front-leg enables the continuation of the wave-gait. Consider the currently transferring leg LT, which will be placed at P. If the wave-gait is used, the two legs that will be transferred after LT are the legs contra-lateral to LT. They are named V and W. The leg that was transferred before LT (according to the wave-gait) is named U. The order of legs being transferred is now: LT, V, W, U. The SFFA is now defined as the maximum possible LASMF_V^W at the instant when leg V can be lifted with stability (Estremera and de Santos, 2005). That is, the distance from the COM projection to the line connecting the new footholds for foot_{LT} (P) and foot_V at the time when the COM projection has entered the support-apttern that excludes foot_V. At the time of this calculation, the new foothold for foot_{LT} is being considered and the next foothold for foot_V is not known. In order to find the SFFA, a limit for the later placement of foot_V is defined, based on the kinematic margin of leg U.

3.3.1 Defining the limiting line B

The distance the COM must travel before leg V can be lifted is described by $\text{LASMB}_{LT}^V(P)$. The situation is illustrated by Figure 3.7.

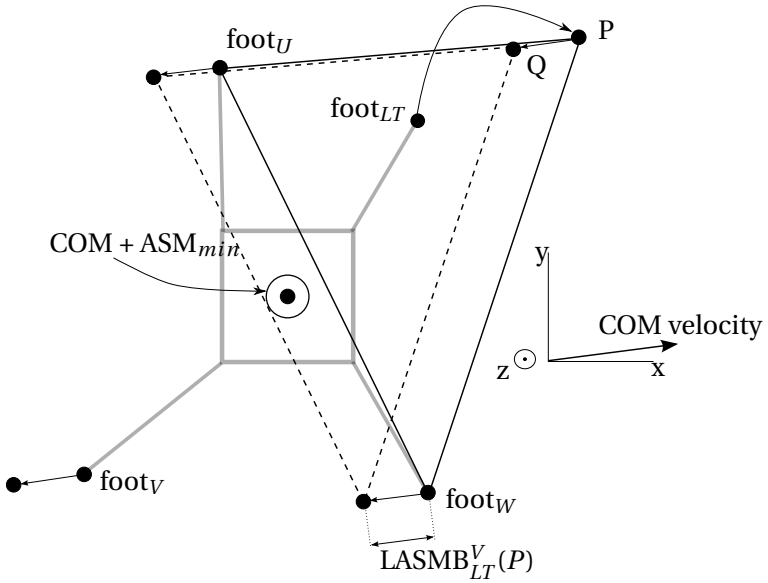


Figure 3.7: $\text{LASMB}_{LT}^V(P)$ is the distance the COM must travel with foot_{LT} placed at P, before the support-pattern that excludes foot_V is entered. The movement of the COM is replaced by movement of the feet in the opposite direction.

The placement of foot_V will define the back-edge of the support-pattern that needs to be entered before leg U can be lifted with stability. Leg U should not reach its kinematic limit before the new support-pattern is entered. This imposes a limit on how far forwards the new foothold of foot_V can be set. Figure 3.8 shows the situation after the transfer of foot_{LT} has completed and the COM has moved $\text{LASMB}_{LT}^V(P)$. At this time, the distance that can be travelled before leg U fails, can be written as KM_U - $\text{LASMB}_{LT}^V(P)$ (KM_U calculated at the time when foot_{LT} was transferred).

Moving the COM an additional KM_U - $\text{LASMB}_{LT}^V(P)$ results in the position of the COM where the maximum distance was travelled before leg U reaches its limit. This position is named $\mathbf{H}_{COM'}^0$ in Figure 3.8. Since leg U has to be transferred before this position is reached (or fail), this COM position (with safety margin ASM_{min}) has to be inside the support-pattern where leg U can be lifted with stability. The back-edge of this support-pattern that lies furthest from the original COM position (i.e. has the highest LASMF_V^W) can be constructed by taking foot_{LT}, foot_W and a

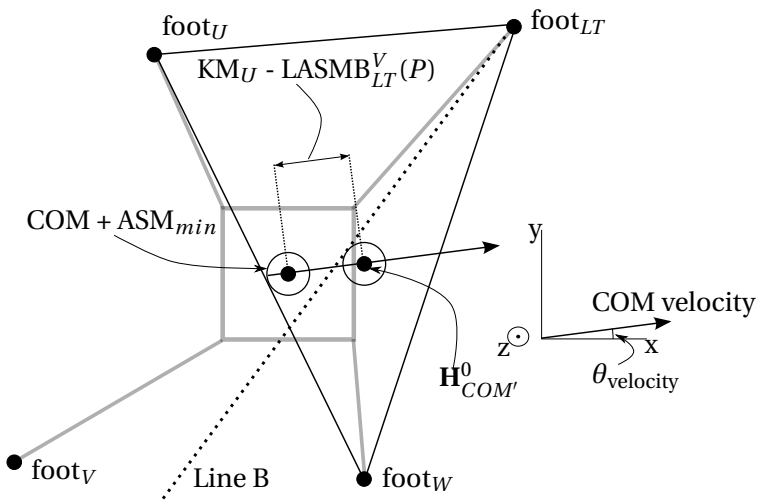


Figure 3.8: The continuation of Figure 3.7. The transfer of foot$_{LT}$ has completed and the COM has moved forwards by LASMB$^V_{LT}(P)$. Leg V can now be lifted with stability. Line B crosses foot$_{LT}$ and is tangent to the the circle with radius ASM$_{min}$ centered at the position where the COM will be after moving an additional KM$_U$ - LASMB$^V_{LT}(P)$.

point on a line crossing foot$_{LT}$ and running tangent to the circle with radius ASM$_{min}$ centered at $\mathbf{H}^0_{COM'}$. This line is referred to as line B (see Figure 3.8). If a foothold for foot$_V$ would be chosen beyond line B, the resulting support-pattern that would enable the lifting of leg U, can not be entered before leg U reaches its kinematic limit. Placing foot$_V$ on top of line B will result in the maximum LASMF^W_V which was defined as the SFFA.

3.3.2 Calculating the Standard Fore Foothold Adequacy

To calculate the SFFA, a reference frame T is used. The frame is constructed in a way that places the COM at the origin and the COM velocity along the x-axis. This frame was also used to calculate the longitudinal absolute stability margins. Its construction is shown in equation 3.7 in section 3.2.

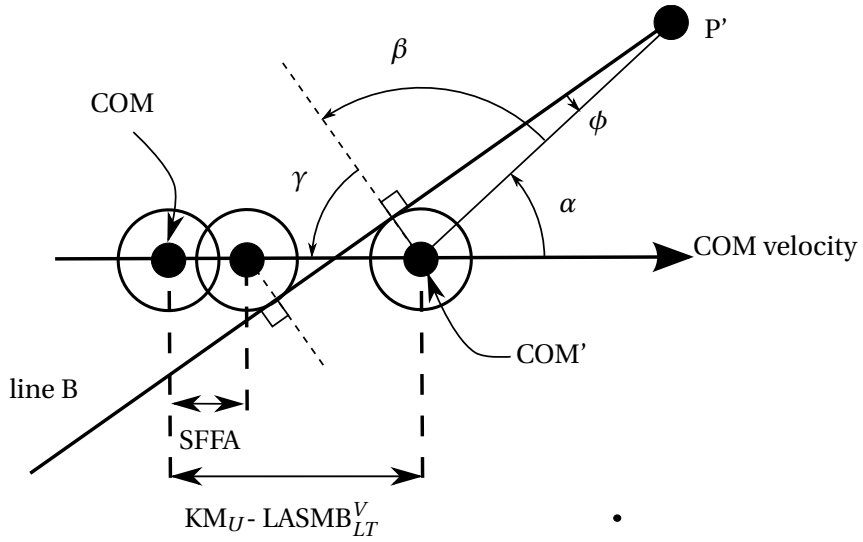


Figure 3.9: The angles needed for the SFFA calculation. Frame T is used so the COM velocity is along the x-axis and the origin is at the COM. P' is the position of foot$_{LT}$ as shown in Figure 3.8 (or Q from Figure 3.7)

Figure 3.9 illustrates how the SFFA can be calculated geometrically. The angles are considered positive in the direction of their arrows. The complete evaluation of SFFA is shown in equation 3.11:

$$\begin{aligned}
\phi &= sin^{-1}\left(\frac{ASM_{min}}{|\,\mathbf{H}^{P'}_{COM'}\,|}\right) \\
\alpha &= abs\left(atan2\left(\mathbf{H}^T_{P'}[2,4] - \mathbf{H}^T_{COM'}[2,4]\ ,\ \mathbf{H}^T_{P'}[1,4] - \mathbf{H}^T_{COM'}[1,4]\right)\right) \\
\beta &= \frac{\pi}{2} - \phi \\
\gamma &= \pi - \alpha - \beta \\
\text{SFFA} &= \text{KM}_U - \text{LASMB}^V_{LT} - 2 \cdot \frac{ASM_{min}}{cos(\gamma)}
\end{aligned} \tag{3.11}$$

Where $\mathbf{H}^T_{P'}$ is the position of the foothold that was selected for foot$_{LT}$, moved back by LASMB$^V_{LT}(P)$ similar to Q in Figure 3.7 or foot$_{LT}$ in Figure 3.8. $\mathbf{H}^T_{COM'}$ is the position of the COM moved forwards by KM$_U$- LASMB$^V_{LT}$. The value for alpha is taken absolute, this is done to account for the change in sign when the other front foot is considered.

3.4 Transfer Distance

The Transfer Distance $TD_{LI}(P)$ is defined as the distance that the COM will travel in the time it takes for an arbitrary leg LI to transfer its foot to a specific location P (Estremera and de Santos, 2005). Since the actual trajectory of a leg is not known, and might be dependent on the physical implementation of the robot, a simplified model for the trajectory is used. Figure 3.10 shows a (simplified) transfer trajectory for a leg as seen from the side. The foot is lifted straight off the ground with speed V1, then moved horizontally to a point above the new foothold and is then lowered straight down with speed V3. Speeds V1, V2 and V3 are scalars. The amount that the foot is lifted above the ground is referred to as the 'ground_clearance'. While this model assumes straight lifting and landing of the leg, the calculation of the TD is still valid when certain alternative trajectories are used, as long as a specific value for V1, V2, and V3 can be derived. For example, for a parabolic trajectory in the xz-plane, V1 and V3 can be considered to be the speed in the z-direction and V2 to the speed in the x-direction.

The minimum value of TD, TD_{min}, is obtained when a foot is lifted to a height of 'ground_clearance' and immediately placed down again.

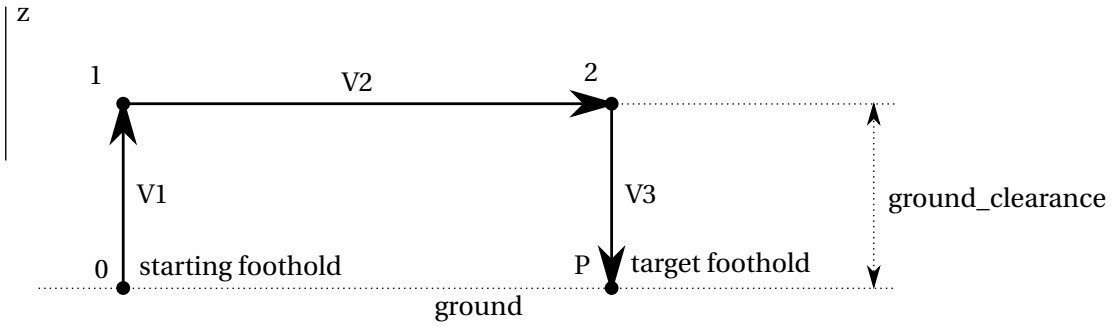


Figure 3.10: Transfer Distance (side-view) A foot is lifted from point 0 to point 1 with speed V1. The speed between point 1 and 2 is V2. The landing from point 2 to the final foothold P is done at speed V3.

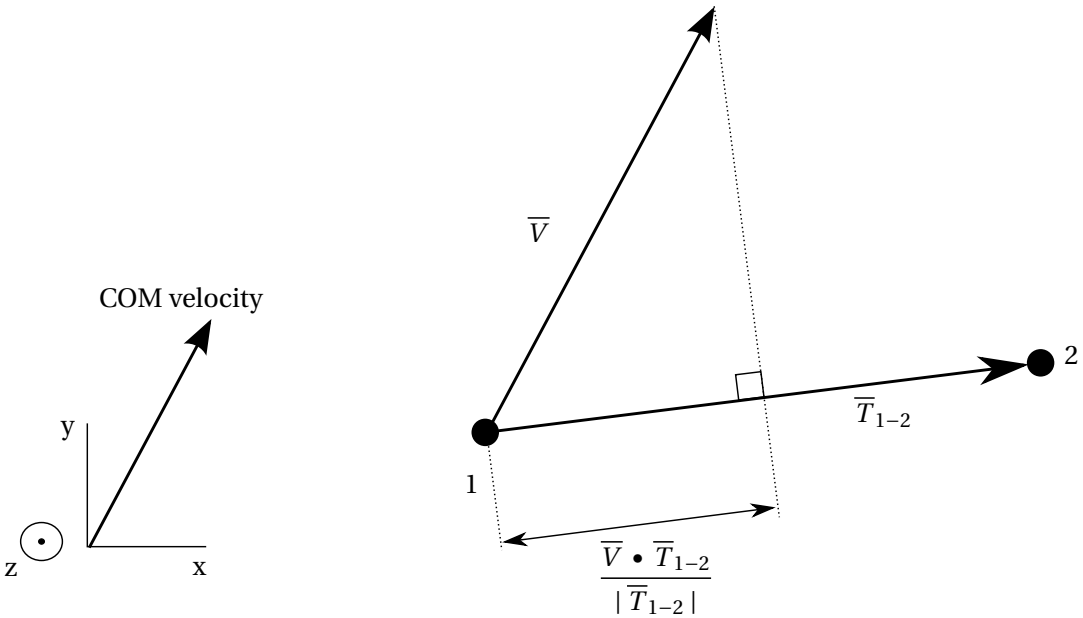


Figure 3.11: Transfer Distance (top-down view). The robot will continue to move during the transfer of a foot, so the actual speed that a foot has between point 1 and 2 from Figure 3.10 is dependent on the robots speed and direction $\overline{V}$. $\overline{T}_{1-2}$ is the vector from point 1 to point 2.

Speed V2 is the result of actuation in the leg. When calculating the transfer distance, the velocity of the COM should also be taken into account (Figure 3.11). Equation 3.12 details the calculation of the transfer distance TD (the distance the COM travels during the transfer of a foot).

$$\begin{aligned} V_{\text{COM}} &= \left|\,\text{COM velocity}\,\right| = \left|\,\overline{V}\,\right| \\ \text{TD}_{min} &= \frac{V_{COM}}{V1 + V3} \cdot 4 \cdot \text{ground\_clearance} \\ V_{\text{transfer}} &= V2 + \frac{\overline{V} \cdot \overline{T}_{1-2}}{|\,\overline{T}_{1-2}\,|} \\ \text{TD} &= \text{TD}_{min} + \frac{|\,\overline{T}_{1-2}\,|}{V_{transfer}} \cdot V_{COM} \end{aligned} \tag{3.12}$$

Where point 1 and 2 are taken from Figure 3.10,
V_{COM} is the speed of the COM, $\overline{V}$ is the COM velocity vector started at point 1 and $\overline{T}_{1-2}$ is the vector from point 1 to point 2. The notation of the transfer distance where the foot of leg LT is transferred to position P is shown in equation 3.13:

$$\text{TD}_{LT}(P) \tag{3.13}$$

3.5 Effective lifting distance

The Effective Lifting Distance is an expansion on the longitudinal absolute stability margin that will be used in the design. When the foot of the currently transferring leg LT is ready to be transferred to position P, the effective lifting distance is the distance that the COM will have travelled before an arbitrary leg LJ can be lifted. i.e. the distance the COM travels before the transfer of LT is complete and the support-pattern that allows the lifting of LJ has been entered (Estremera and de Santos, 2005).

As detailed in section 3.2, the distance to the required support-pattern to lift leg LJ is LASMB_{LT}^{LJ}. It is possible that this distance is negative because the COM is already in this support-pattern. Or the transfer of the foot of leg LT to P takes longer than the time needed to enter the support pattern. This can be summarized in a single statement shown in equation 3.14:

$$\mathrm{ELD}_{LT}^{LJ}(P) = max\left(\mathrm{LASMB}_{LT}^{LJ}(P)\ ,\ \mathrm{TD}_{LT}(P)\right) \tag{3.14}$$

Where $\mathrm{ELD}_{LT}^{LJ}(P)$ is the Effective Lifting Distance that the COM covers from the time when leg LT is lifted to the time when leg LJ can be lifted. $\mathrm{TD}_{LT}(P)$ is the distance covered by the COM during the transfer of the foot of leg LT to P.

3.6 Effective Stability Distance

the Effective stability distance is a measure of how well the placing of the foot of the currently transferring leg LT at P enables the complete transfer of another arbitrary leg LJ. Its value is the distance that the COM can travel with leg LJ in the air, before the robot looses stability or the kinematic limit of leg LT is reached (Estremera and de Santos, 2005).

$$ESD_{LT}^{LJ}(P) = min\left(LASMF_{LT}^{LJ}(P),\ KM_{LT}(P)\right) - ELD_{LT}^{LJ}(P) \tag{3.15}$$

The travel distance available until stability is lost with leg LJ in the air and the foot of leg LT placed at P is $\mathrm{LASMF}_{LT}^{LJ}(P)$. The kinematic margin of leg LT after placing its foot at P is $\mathrm{KM}_{LT}(P)$. Before leg LJ can be lifted, the effective lifting distance needs to be covered. This leads to the expression for the effective stability distance shown in equation 3.15.

4 Design

This chapter covers the design of the gait-generator and its implementation in 20-sim. This design is based on one detailed in (Estremera and de Santos, 2005). The goal of the gait-generator is to provide the robot-model with actions that should be taken in order to remain stable while moving.

4.1 Inputs and outputs

The gait-generator provides this action in the form of an index for a leg that should be moved and the position where that legs foot should be placed. To do this, the gait-generator needs some information from the robot-model. Part of this is variable during operation and should be up to date at all times:

- the location of all feet.
- the location and velocity of the COM projection on the xy-plane.
- an indication that the current transfer of a leg has finished.

The rest is contained in static parameters:

- the HL1 and HL2 points relative to any pivot.
- leg-transfer speeds V1,V2,V3.
- the location and orientation of the pivots.
- The minimum allowable stability margin: ASM_{min}.

Table 4.2 and 4.1 summarize these values. There is no specific output to indicate *when* a leg should be moved. A change in leg_index and H_foot implies that a leg should be moved immediately.

Name	Type	Description
leg_index	Integer	Index of the leg for which the position is given in H_foot (1..4)
H_foot	H-matrix (4x4)	Recommended target position for the foot of the leg with index I_leg

Table 4.1: Outputs of the gait-generator. Lengths are in meters

Name	Type	Description
		Variables
P1	H-matrix (4x4)	Foot position of leg 1
P2	H-matrix (4x4)	Foot position of leg 2
P3	H-matrix (4x4)	Foot position of leg 3
P4	H-matrix (4x4)	Foot position of leg 4
Pcom	H-matrix (4x4)	Position and orientation of the center of mass on the 2D plane
velocity	vector (3x1)	speed and direction of movement for the center of mass
b_transferring	boolean	True when a leg is currently being transferred
		Parameters
HL1	H-matrix (4x4)	Position of HL1 (chapter 3.1) relative to a pivot$_{n,0}$.
HL2	H-matrix (4x4)	Position of HL2 (chapter 3.1) relative to a pivot$_{n,0}$.
V1	Scalar	Speed of a foot when a leg is lifting it (chapter 3.4).
V2	Scalar	Speed of a foot between lifting and landing (Chapter 3.4).
V3	Scalar	Speed of a foot when it's landing (chapter 3.4).
ground_clearance	Scalar	Distance that a foot is lifted from the ground when transferring
ASM_min	Scalar	Minimum allowable stability margin.
rel_pivot10	H-matrix	Position and orientation of pivot$_{1,0}$ relative to Pcom.
rel_pivot20	H-matrix	Position and orientation of pivot$_{2,0}$ relative to Pcom.
rel_pivot30	H-matrix	Position and orientation of pivot$_{3,0}$ relative to Pcom.
rel_pivot40	H-matrix	Position and orientation of pivot$_{4,0}$ relative to Pcom.

Table 4.2: inputs to the gait-generator. Lengths are in meters, speeds in meters per second

4.2 The state-machine

The gait-generator operates a small state machine depicted in Figure 4.1.

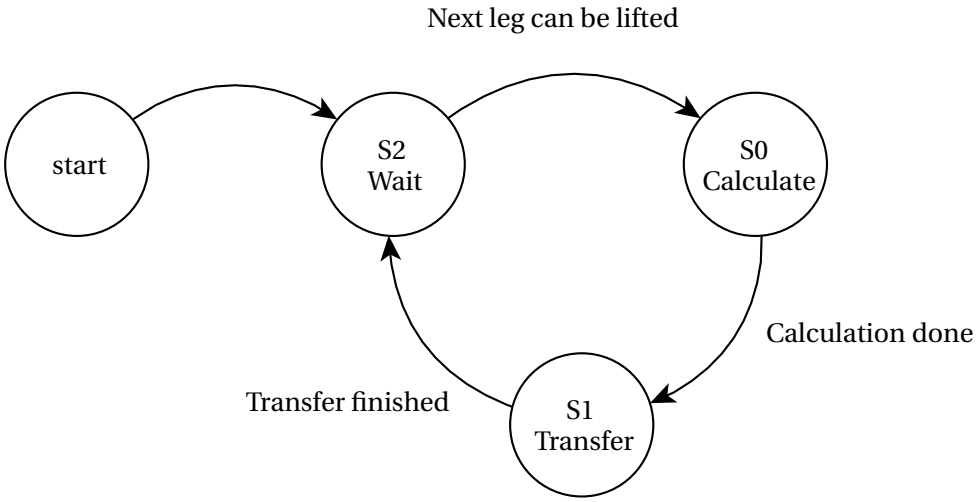


Figure 4.1: The main gait-generator state-machine

4.2.1 State 0: calculate

In this state leg LT is ready to start its transfer. The gait-generator has to calculate where to transfer it to, and in the process determine the preferred next leg to be lifted (NLT). Once the calculation is done, the output values of the gait-generator are set, indicating the foot should be transferred, and the state-machine enters S1.

4.2.2 State 1: transfer

Leg LT is currently transferring. Nothing has to be done until the b_transferring input becomes false. S2 is entered when this happens.

4.2.3 State 2: wait

This is the starting state for the state-machine. All feet are on the ground. A check (section 4.3) is made for the ability to lift a leg with stability. the NLT selected in S0 has preference during these checks. Once a leg can be lifted, its index is stored as LT and the transition is made to S0.

4.3 Leglifting check

Before the leglifting check is made, the order of preference of the legs is established. The NLT selected in S0 is always the top preference. After NLT come all four legs in ascending order of kinematic margin:

$$\text{Order of leg preference} = [\ \text{NLT}\ ,\ \text{lowest KM} \ldots \text{highest KM}\ ]$$

Each leg on the list is considered separately. The current candidate leg in question is referred to as leg LC. If both of the following conditions are satisfied, then that leg can be lifted while maintaining stability:

1. The distance to the new support-pattern must be reachable before any of the other legs reach their kinematic limit:

$$\text{LASMB}^{LC} \quad < \quad \text{lowest KM}$$

2. When leg LC is in the air, the COM should be inside the new support-pattern:

$$\begin{aligned} \text{LASMB}^{LC} &< 0 \\ \text{LASMF}^{LC} &> 0 \end{aligned}$$

Statement 1 will automatically be true if statement 2 is satisfied. It is still included because it allows earlier detection of failure for the gait-generation: if all candidate legs fail on condition 1 then the robot will not be able to lift any leg with stability before a kinematic limit is reached and the robot will have to stop. This functionality is not yet available in the current implementation.

When the first statement is true, it is still possible that none of the legs can be lifted at the current moment. The robot will have to wait for either the COM to move, or the COM velocity to change. Another possibility is that when NLT can't be lifted, but another leg can. In this case the other leg will only be selected if there is a proper reason for distrusting the NLT that was set in S0. In the current implementation proper reasons are:

- There is no NLT set (i.e. start of the system)
- The definitions of front and back legs changed due to a change in the direction of movement since S0.

If none of these occur then only NLT is considered.

4.4 Foothold calculation

To calculate a valid foothold for the foot of leg LT, an evaluation of 225 possible footholds is done. The points are spread evenly across a 15 by 15 square and the current foothold is at the center. The size of the area can be set with a parameter in the 20-sim model. To judge a point P, from the evaluation area, as a foothold, some limits are imposed on the values calculated for that point. Each of these limits generates a restricting area wherein the foothold should lie. Which of the limits should be applied is dependent on whether a leg is front or rear and left or right. The limits for the different areas are covered below, followed by the operation of the NLT-selection.

4.4.1 Area O

When placing the foot of leg LT at P, Area O consists of the points where the expression in equation 4.1 is true:

$$\text{KM}_{LT}(P) > \text{KM}_{min} \tag{4.1}$$

Where $\text{KM}_{LT}(P)$ is the kinematic margin of leg LT when its foot is placed at P, and KM_{min} is the lowest KM in the system without considering KM_{LT} (leg LT is going to be moved). Plainly stated, after placing the foot of leg LT at P, the new kinematic margin of leg LT should be higher than all other kinematic margins in the system. Since KM has been defined as -1 when P is outside the reachable area (chapter 3.1), this statement automatically excludes those footholds. This area also prevents the reduction of total kinematic margin. Area O should also contain a check for obstacles or leg-collisions that would prevent P from being a valid foothold. This is not currently implemented though. This condition has to be met for every foothold, always.

4.4.2 Area D

When placing the foot of leg LT at P, Area D consists of the points where the expression in equation 4.2 is true:

$$\mathrm{TD}_{LT}(P) < \mathrm{LASMF}^{LT} \tag{4.2}$$

Where $\mathrm{TD}_{LT}(P)$ is the distance that the COM will travel during the transfer of the foot of leg LT to P (chapter 3.4), and LASMF^{LT} is the distance the COM can move forwards before the support-pattern without the currently transferring leg LT is exited (chapter 3.2). This area makes sure that during the transfer of the foot of leg LT to P, the COM does not move outside of the support-pattern. This condition has to be met in order for the robot to remain stable so it's required for all valid points.

4.4.3 Area A

When placing the foot of leg LT at P, Area A consists of the points where the expression in equation 4.3 is true:

$$\mathrm{ESD}_{LT}^{NLT}(P) > \mathrm{TD}_{min} \tag{4.3}$$

Where $\mathrm{ESD}_{LT}^{NLT}(P)$ is the effective stability distance that can be covered by the COM when the foot of leg LT is placed at P and the next transferring leg NLT is in the air (chapter 3.6). TD_{min} is the minimum transfer distance (chapter 3.4). Suppose the foot of LT is placed at P and the COM has entered the support-pattern that excludes NLT. The distance that the COM can travel with leg NLT in the air before loosing stability should be more than the distance that will be covered when just lifting and landing a foot. For adjacent legs this area indicates that NLT can be lifted immediately after LT has been placed. This area is only used when LT and NLT are adjacent.

4.4.4 Area B

When placing the foot of leg LT at P, Area B consists of the points where the expression in equation 4.4 is true:

$$\mathrm{ELD}_{LT}^{LJ}(P) < \mathrm{KM}_{min} \tag{4.4}$$

Where $\mathrm{ELD}_{LT}^{LJ}(P)$ is the effective lifting distance when transferring the foot of the currently transferring leg LT to P, before another leg LJ can be lifted (chapter 3.5). KM_{min} is the lowest KM in the system. None of the remaining legs should reach their kinematic limit before the leg LJ can be lifted. For this area, leg LJ is always the rear leg that is closest to LT (without being LT). This means that when LT is one of the rear legs, LJ is not the next leg to be lifted when the wave-gait is assumed, but the one after that. This rear leg is considered instead of NLT because placing the foot of leg LT at P forms the back-edge of the support-pattern that should be entered before LJ can be lifted. KM_{min} should not consider LT or NLT since those KMs will be increased by leg-transfers before that rear leg (LJ) can be lifted. This area is only useful when a rear leg is being transferred.

4.4.5 Area C

When placing the foot of leg LT at P, Area C consists of the points where the expression in equation 4.5 is true:

$$\mathrm{SFFA}_{LT}(P) < 2 \cdot \mathrm{TD}_{min} \tag{4.5}$$

Where $\mathrm{SFFA}_{LT}(P)$ is the standard fore foothold adequacy for the foothold P of the currently transferring leg LT (chapter 3.3), and TD_{min} is the minimum transfer distance (chapter 3.4). When LT is a front leg, this statement generates the area of footholds that would allow two consecutive transfers of the contra-lateral legs. This Area can be ignored if LT is not a front leg, or if the wave-gait order is not used since, in that case, it's not certain that the following two transfers will be the contra-lateral legs.

4.4.6 NLT and target foothold selection

As explained in the previous sections, not all areas should be used in all cases. The ones that are applicable, are combined. The result is stored as a 15 by 15 array with a 1 where a foothold is inside all required areas and a 0 otherwise. Figure 4.2 illustrates an example of this.

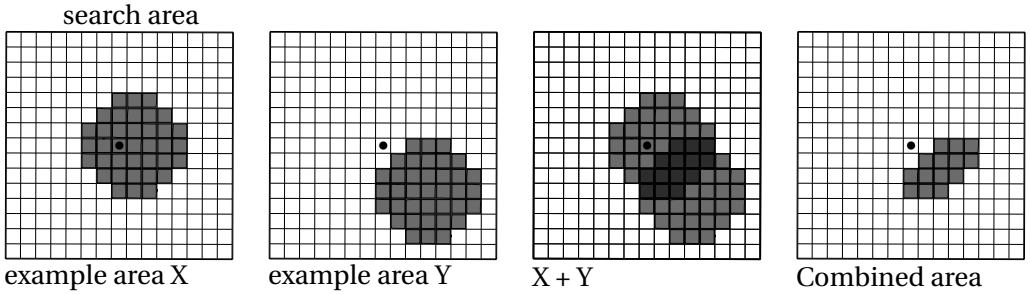


Figure 4.2: An example of combining two limiting areas for the footholds named X and Y (1's are indicated with a darker color). The resulting Combined area on the far right shows the collection of all footholds that appear in X as well as Y.

Areas A and B are dependent on the choice of NLT. Because this design is implemented in 20-sim, there appears to be no suitable option to procedurally generate an area for a specific NLT, check it and then either continue or recalculate for the next NLT in the list. The areas are therefore calculated with each of the four most preferred NLT from the list of section 4.3 (generated again with the new LT). Four combined areas are generated. The NLT with its resulting area, that will be used in further foothold selection, is the first NLT of the ordered list that produces a combined area array where one or more footholds are valid.

Once an array with valid footholds is available, the best foothold should be selected. Many factors can be considered for this selection. Due to limited time available for this project however, the choice was made to optimize for kinematic margin. i.e. the foothold that results in the highest KM is chosen as target foothold for LT.

4.5 20-sim implementation

The implementation that was made in 20-sim is a single model that contains the inputs and outputs specified in section 4.1. It can be roughly divided into five numbered parts as shown in Figure 4.3.

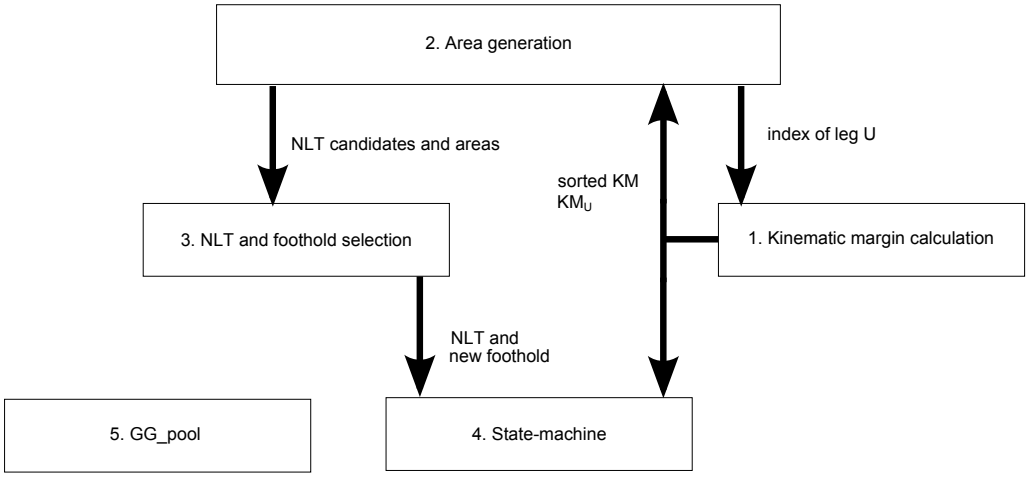


Figure 4.3: The 20-sim model divided into five parts. The arrows indicate the most important information transfers between them.

The parts are:

1. *Kinematic margin calculation*: the kinematic margins for all four feet at their current positions are calculated, these values are then sorted in ascending order. the kinematic margin for leg U, used in the calculation of the SFFA, is also available as output of this part.
2. *Area generation*: The five areas A,B,C,D, and O are generated here.
3. *NLT and foothold selection*: In this part, the areas are combined, and the best NLT with its area, is chosen. From these, the best foothold for leg LT is selected.
4. *State-machine*: This part contains the implementation of the state-machine.
5. *GG_pool*: This is a single submodule that houses the global variables used throughout the model.

The complete 20-sim model is shown in appendix A. The following section covers all of the submodels (and their submodels, if any).

4.5.1 Selector modules

The selector modules are indicated with a red border in 20-sim and appear in all parts shown in Figure 4.3. The only function of these modules is to select one of the values from their inputs or the global pool based on a leg index and provide them as input for another module. For instance, the 'select_P' module outputs the position of foot 1 through 4, based on the index provided at the input. All selector modules also except 0 at their index input, which is constructed to mean the index of the currently transferring leg LT.

4.5.2 The global variable pool

The module named "GG_pool" (part 5 in Figure 4.3) contains the variables, constants and parameters that are commonly used throughout the model. Table 4.2 already shows most of them. The additional ones that are only used internally are summarized in table 4.3. The 'array_size' is defined as a constant and not as a parameter. This is done because the value is also hardcoded into many of the submodel interfaces.

Name	Type	Description
		Constants
array_size	Integer	Number of rows (and columns) in the search area. Set to 15.
		Parameters
search_coverage	real	Length of a side of the search area.
		Variables
HL11	H-matrix (4x4)	HL1 of leg 1
HL12	H-matrix (4x4)	HL2 of leg 1
HL21	H-matrix (4x4)	HL1 of leg 2
HL22	H-matrix (4x4)	HL2 of leg 2
HL31	H-matrix (4x4)	HL1 of leg 3
HL32	H-matrix (4x4)	HL2 of leg 3
HL41	H-matrix (4x4)	HL1 of leg 4
HL42	H-matrix (4x4)	HL2 of leg 4
pivot10	H-matrix (4x4)	Absolute position and orientation of $pivot_{1,0}$
pivot20	H-matrix (4x4)	Absolute position and orientation of $pivot_{2,0}$
pivot30	H-matrix (4x4)	Absolute position and orientation of $pivot_{3,0}$
pivot40	H-matrix (4x4)	Absolute position and orientation of $pivot_{4,0}$
LT	Integer [1...4]	Index of LT

Table 4.3: Additional global variables stored in GG_pool. These are in addition to the variables listed in Table 4.2.

4.5.3 Parray

Inputs: (H-matrix) P_in
Outputs: (45x15 Array) output
This module is shown as a blue box with white text in the full model in Appendix A and resides in part 2 of Figure 4.3. It outputs an array that contains all the coordinates of the points in the search area around a foot at P_in. Figure 4.4 shows the search area that is generated. The search

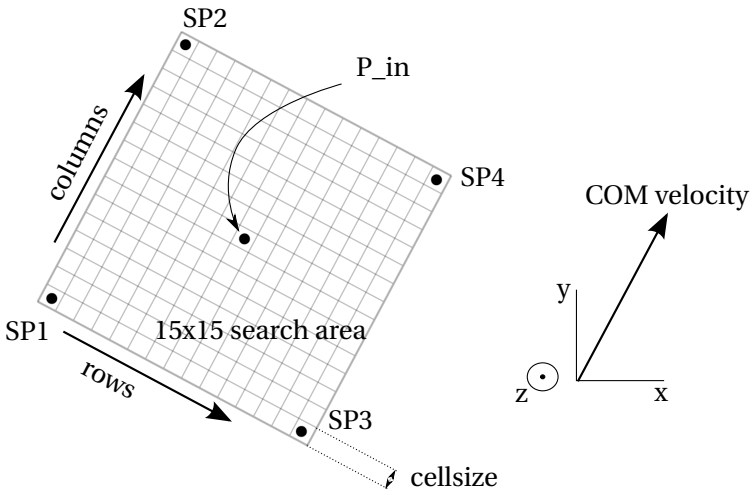


Figure 4.4: the search area for a new foothold: Parray, Its a 15x15 array aligned to the COM velocity and centered at a provided position P_in. SP1 through SP4 are the corner coordinates.

area is rotated so the columns of the matrix are aligned with the COM velocity. The coordinates of all 225 points are stored in the output matrix as shown in equation 4.6:

$$\text{output} = \begin{bmatrix} SP1_x & \dots & \dots & \dots & SP2_x \\ SP1_y & \dots & \dots & \dots & SP2_y \\ SP1_z & \dots & \dots & \dots & SP2_z \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ \vdots & \ddots & P\_in_x & \iddots & \vdots \\ \vdots & \dots & P\_in_y & \dots & \vdots \\ \vdots & \iddots & P\_in_z & \ddots & \vdots \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ SP3_x & \dots & \dots & \dots & SP4_x \\ SP3_y & \dots & \dots & \dots & SP4_y \\ SP3_z & \dots & \dots & \dots & SP4_z \end{bmatrix} \tag{4.6}$$

The coordinate value are stored as vectors instead of H-matrices to save some memory space.

4.5.4 Calculator modules

These modules are indicated by a blue border in 20-sim and are present in parts 1 through 4 from Figure 4.3. They handle the calculation of the values explained in chapter 3: $\text{KM}_{LI}(P)$, $\text{LASMB}_{LI}^{LJ}(P)$, $\text{LASMF}_{LI}^{LJ}(P)$, $\text{TD}_{LT}(P)$, $\text{ESD}_{LI}^{LJ}(P)$ and $\text{ELD}_{LI}^{LJ}(P)$. The naming scheme used in the 20-sim model (Appendix A):
"(name)_(subscript)_(superscript)". an "_P" is added if the value is calculated for all P in Parray.

4.5.5 Area modules

Each area described in section 4.4 is evaluated in one of the orange-bordered modules. They appear in parts 2 and 3 shown in Figure 4.3. Their output is a 15x15 array of integers. A 1 indicates that point P on that index in Parray is insides that particular area. A 0 means that it is not. A value of -1 (at any index) denotes that there are no valid points for that area in Parray. This is done so the 'select_best_nlt' module can easily determine if an array has valid points in it.

4.5.6 sorter_wLT

inputs: (real) KM_1, (real) KM_2, (real) KM_3, (real) KM_4
outputs: (4x2 array) KM_list_ordered
The sorter_wLT module takes KM 1 through 4 and places them in ascending order in the 4x2 output array. The second column is the KM value and the first is the index of leg for which that value was calculated. KM_{min} can be taken directly from the array at the top row, second column.

4.5.7 determine_order

This module outputs a 4x1 array with leg indices. From first to last these indices stand for the left-rear, left-front, right-rear, right-front legs. This array is used in modules that need to know which legs are front and which are rear. Figure 4.5 displays the sectors for the angle of the COM velocity that are used to determine the order. The current implementation in 20-sim

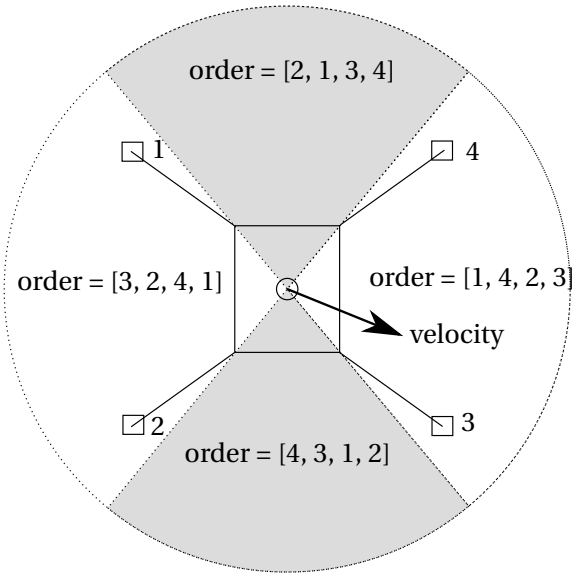


Figure 4.5: Leg indices for [left-rear, left-front, right-rear and right-front] based on the sector where velocity lies.

uses a set of IF-statements to determine which sector should be used and then outputs the corresponding order-array. A single function is possible where the entries in the order-array are inc/decremented (and wrapped around so 5=1 and 0=4) based on the angle of the COM velocity in relation to the sectors.

4.5.8 order_NLT

The list of preferred NLT used for the repeated calculation of area A and B is generated here. The output is a 4x1 array with the indices of the legs:

$$\text{output} = [\ \text{leg NLT}\ ,\ \text{lowest KM}\ \ldots\ \text{highest KM}\ ]$$

4.5.9 select_best_nlt

This module takes all combined areas and their corresponding NLTs and outputs the first combination with valid footholds according to the list of preferred NLT.

4.5.10 select_best_ij

From the combined area that was selected by select_best_nlt, this module selects the row and column indices i and j that correspond to the valid foothold with the highest kinematic margin.

4.5.11 looper

The entire state-machine is contained in this module, including the leglifting check. It is the only module besides GG_pool itself, that writes to the global variable pool.

5 Evaluation and testing

The 20-sim implementation of the design has been tested as a stand-alone model without the robot-model attached. This was done to avoid long simulation times due to the constraints in the robot-model.

A 3D representation of a test robot has been made in the 20-sim 3D scene (Figure 5.1). This representation makes it easy to spot when stability is lost or a foot is placed outside its reachable area.

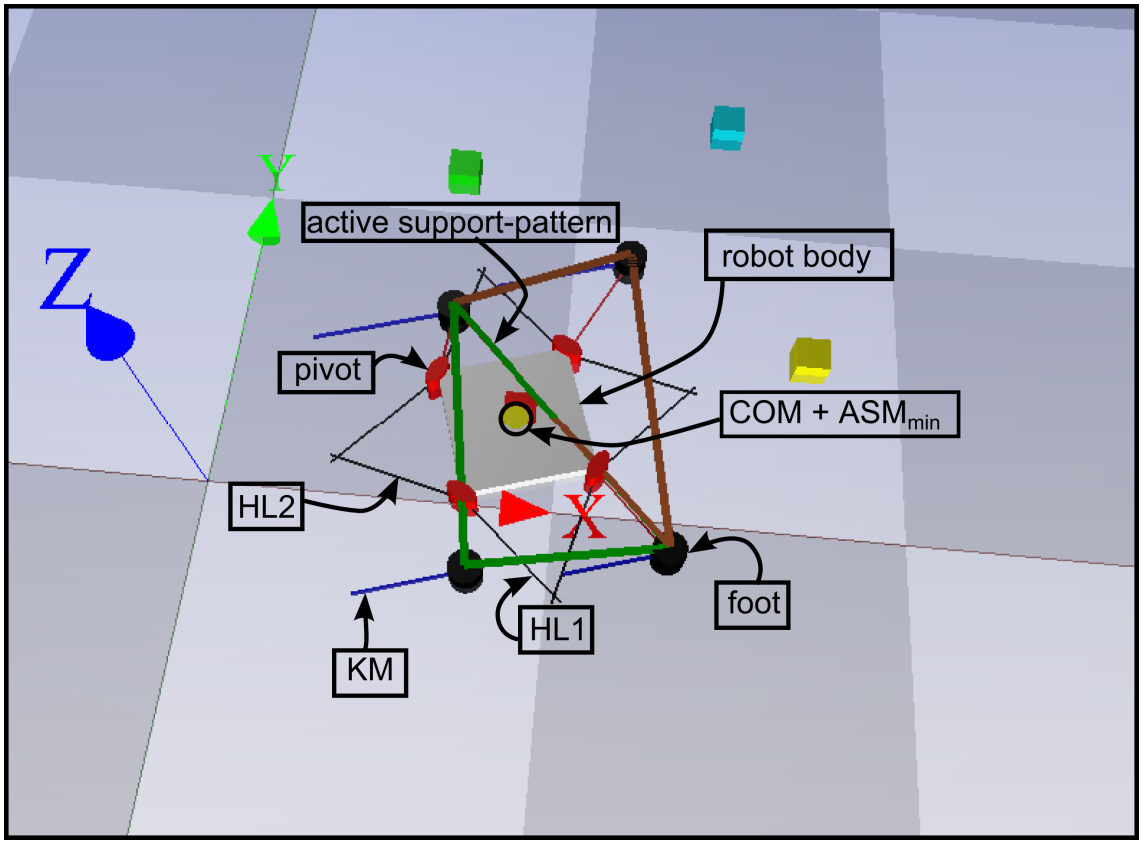


Figure 5.1: 3D representation of the test robot

The named items in Figure 5.1 are:

- *foot*: A black cylinder represents the position of a foot of the robot.
- *active support-pattern*: This is the support-pattern the the COM projection of the robot is currently in. It is shown as a green triangle.
- *pivot*: The four pivot$_{n,0}$ are indicated with a red cylinder. To show the orientation of the pivot, the cylinder is slightly stretched in the pivots x-direction.
- *robot body*: A grey block represents the robots body. The four legs are connected at the pivots on the four corners.
- *HL1 and HL2*: These are lines from a pivot to its respective limit defining points: $\mathbf{H}_{HL1}^{pivot_{n,0}}$ and $\mathbf{H}_{HL2}^{pivot_{n,0}}$.
- *KM*: This is a line with the length of the kinematic margin for its leg. The line is in the opposite direction of movement and starts at the foot. It is the same representation of KM as used in the analysis in chapter 3.1.
- *COM + ASM$_{min}$*: A yellow circle around the COM with radius ASM$_{min}$. This circle should always be inside the active support-pattern in order for the robot to be statically stable.

5.1 Simulation parameters

Table 5.1 shows the parameters that are constant for all tests. Some notes on the impact of these parameters:

- ASM_{min}: reducing this value will make it easier to get a stable gait, and reducing it to 0 will allow a stable gait at almost all crab-angles.
- *V1*/*V2*/*V3*: Increasing the leg speed improves the maximum speed.
- *HL1*: The length of HL1 is interpreted as the maximum reach of the leg.
- *search_coverage*: increasing this value beyond the size of the reachable area for a foot will do nothing but decrease the resolution of the valid foothold area. Decreasing it so its smaller than the reachable area will effectively limit the quadrupeds step-size.

20-sim Simulated the models discrete system at 2000 Hz with the Backwards Differentiation Formula method.

name	value(s)
ASM_min	0.04
ground_clearance	0.01
search_coverage	0.09
V1	5.0
V2	5.0
V3	5.0
HL1 and HL2	$\begin{bmatrix} 1 & 0 & 0 & 0.4 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0.4 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
rel_pivot_10 and rel_pivot_20	$\begin{bmatrix} \frac{1}{2}\sqrt{2} & -\frac{1}{2}\sqrt{2} & 0 & -0.2 \\ \frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} & 0 & 0.2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2}\sqrt{2} & -\frac{1}{2}\sqrt{2} & 0 & -0.2 \\ -\frac{1}{2}\sqrt{2} & -\frac{1}{2}\sqrt{2} & 0 & -0.2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
rel_pivot_30 and rel_pivot_40	$\begin{bmatrix} \frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} & 0 & 0.2 \\ \frac{1}{2}\sqrt{2} & -\frac{1}{2}\sqrt{2} & 0 & -0.2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} & 0 & 0.2 \\ -\frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} & 0 & 0.2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Table 5.1: Parameters used in all tests

5.2 Simulation results

The results of the simulations done with the parameters specified in the previous section, are presented in the following subsections.

5.2.1 Constant forward speed

In this test the body was rotated around the z-axis with a set of angles starting at -π and ending at π with 0.1 increments. the velocity was set constant at 0.1 m/s and with the same angle as the body rotation. For all angles the resulting gait was stable. Equation 5.1 shows the starting H-matrix of the COM projection and the COM velocity used.

$$\begin{aligned} \mathbf{H}^0_{COM} &= \begin{bmatrix} cos(\alpha) & -sin(\alpha) & 0 & 0 \\ sin(\alpha) & cos(\alpha) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ \text{COM velocity} &= \begin{bmatrix} s \cdot cos(\alpha) \\ s \cdot sin(\alpha) \\ 0 \end{bmatrix} \end{aligned} \tag{5.1}$$

Where α is the angle for the speed and body-orientation. and s is the simulated speed.

For the angles α of 0.1, 0.2 and 0.3 the simulation was ran multiple times with the speed s incremented with 0.05 m/s until the resulting gait was no longer stable. This maximum speed was found at 0.6 m/s.

To confirm that this maximum speed is linearly dependent on the leg speeds, a simulation was done with an angle of 0.1 and double speed (1.2 m/s). The leg speeds and simulation frequency were also doubled to 10 m/s and 4000 Hz respectively. The result of this test was a stable gait while additional tests at speeds of 1.25 and 1.3 m/s did not yield stable gaits.

5.2.2 Non-zero crab-angle

In this test a relative angle of speed to body orientation of θ is introduced. This angle was set to 0.1, 0.2, 0.3, 0.4, 0.5, and 0.6 rad . To find the maximum speed for each of these crab-angles, simulations were done where the speed was incremented with 0.05 m/s starting at 0.1 m/s until the resulting gait was no longer stable. Equation 5.2 shows the expression used for the COM velocity:

$$\text{COM velocity} = \begin{bmatrix} s \cdot cos(\theta) \\ s \cdot sin(\theta) \\ 0 \end{bmatrix} \tag{5.2}$$

Where θ is the crab-angle and s is the simulated speed.

For crab-angles θ of 0.1 through 0.4, the maximum speed s was found to be 0.5 m/s. This is slightly lower than what was found for a crab-angle of 0 in section 5.2.1. The cause of this was observed to be the kinematic limits of the legs, which are rotated with the body, resulting in a generally lowered maximum KM. Crab-angles higher than 0.4 rad did not produce a stable gait at any of the simulated speeds.

An observation was made that simply selecting the foothold with the largest KM from the valid positions does not always result in the best foothold.

5.2.3 Variable speeds

In this test, the COM velocity was made variable. Equation 5.3 shows the expression for the COM velocity that was used:

$$\text{COM velocity} = \begin{bmatrix} 0.5 \cdot cos(2\,\pi\,t) \\ 0 \\ 0 \end{bmatrix} \tag{5.3}$$

The gait-generator can handle this and switches the leg-order around when the speed crosses 0, resulting in the opposite gait while remaining stable.

5.2.4 Variable velocity angles

in this test the attempt was made to make the robot walk along a circle. The expression for the COM velocity in equation 5.4 results in a circular trajectory for the COM at constant speed of 0.5 m/s.

$$\text{COM velocity} = \begin{bmatrix} 0.5 \cdot cos(2\,\pi \cdot 5\,t) \\ 0.5 \cdot sin(2\,\pi \cdot 5\,t) \\ 0 \end{bmatrix} \tag{5.4}$$

In this setting the gait-generator fails to provide a stable gait. Even when the radius of the circle is increased and the speed lowered. This is because the values of LASMB/F and KM assume straight motion of the COM and the feet positions are based on those values.

5.2.5 General observations

Some general observations that were made during the less structured tests of the model:

- Reducing ASM_{min} to 0 enables gait-generator to provide a stable gait for all crab-angles. This might be, in part, because the ASM_{min} is only considered in the LASMB/F modules.
- Moving the wave-gait NLT to the bottom of the preference list in the 'select_best_nlt' module will still cause the wave-gait to be generated when the crab-angle is 0.

6 Conclusion and recommendations

6.1 conclusion

A model has been designed and built in 20-sim that functions as a gait-generator for a quadruped. Simulations of the model have shown that it is capable maintaining stability when moving the robots center of mass in its forward or backwards direction while maintaining the input speed. Crab-angles of up to 0.5 rad are possible with reduced speed. The maximum speed is dependent on the speed, and size parameters of the legs. There is also a lower limit to the reachable area for the legs before stable movement is no longer possible.

The output gait is dynamically generated and takes all up-to-date information into account at every step. Because the model uses the transfer distance of the COM rather than a time measure, it will provide a stable gait at any speed below the maximum.

The model has been designed so it can be expanded to include forbidden zones for the feet to land by excluding them from the limiting area O (see chapter 4.4.1)

Terrain adaptability can be introduced by improving the TD (transfer distance) modules to include the uncertainty of the terrain. The core principles of the design do not change when the terrain becomes more challenging.

6.2 Recommendations

The following subsections detail a multitude of areas in the model that could still use some work. They are presented in descending order of impact/improvement.

6.2.1 Failure detection

During the S2 state in the looper (state-machine) module, there is a possibility to detect that a stable gait can no longer be generated. If this is detected then a signal could be sent to the robot to reduce speed or stop entirely. This will improve the chances for a stable gait in difficult scenarios. This upgrade should be implemented in the part of the looper submodule where the leglifting check is handled. Comments have been left in the submodule code at this position.

6.2.2 Circular gait or turning

To make the gait-generator truly omni-directional a turning mode or a circular gait should be implemented. The second option can be achieved by updating the KM and LASMB/F calculation to include the angular velocity of the COM. A way to achieve 'on the spot' turning would be to have the gait-generator select footholds that are in the center of the reachable area when the speed is below a certain threshold. This would allow the robot to turn by moving the COM projection into the four different support-patterns at low speed while rotation the body.

6.2.3 Foothold selection

The foothold selection that occurs in the select_best_ij submodule only takes the kinematic margin of the foothold into account. The simulations show that this is often not enough and more information should be considered here before deciding on one of the valid footholds. Improved optimization rules for the selection are already detailed in (Estremera and de Santos, 2005).

6.2.4 Optimizations

The model has been going through a continuous test-update loop and has not been optimized. There are a few debugging variables left in the submodels and most of the programming can be rewritten in a more efficient form. Other languages like C/C++ provide much more flexibility in building, designing and debugging a code-based model. 20-sim has to ability to load external libraries and interact with them through functions. Rebuilding the model in C/C++ and providing library function for 20-sim will vastly improve the debugging options for the gait-generator.

6.2.5 Obstacles and leg-collisions

Area O can be used to overlay an area that excludes forbidden zones from the valid footholds, effectively preventing the robot from placing feet there. When the kinematic limits of the leg are set in such a way that leg-collisions are possible then area O should be the place to detect and exclude those footholds.

6.2.6 Floating point accuracy issues

The positions of the feet are stored as H-matrices relative to the origin rather than to the COM. This will cause a loss of accuracy when the robot moves very far from the origin. A good solution would be to store the feet positions relative to the COM and update all submodules to use the relative positions. (this will actually also improve simulation speed since some transformations will be simplified)

6.2.7 Minimum absolute stability margin

The ASM_{min} that is used in chapter 3.2 to calculate the LASMB/F is not considered as a circle but as a simple value that should be added/subtracted. To clarify: it is possible that when the COM has moved a distance of LASMF forward, the circle around the COM with radius ASM_{min} is not completely contained in the support triangle. This should be resolved in the LASMB/F modules. As mentioned in chapter 5, the ASM_{min} should perhaps be considered in other modules as well in order to allow stable gaits at larger crab-angles.

6.2.8 Favour of non-wave-gaits

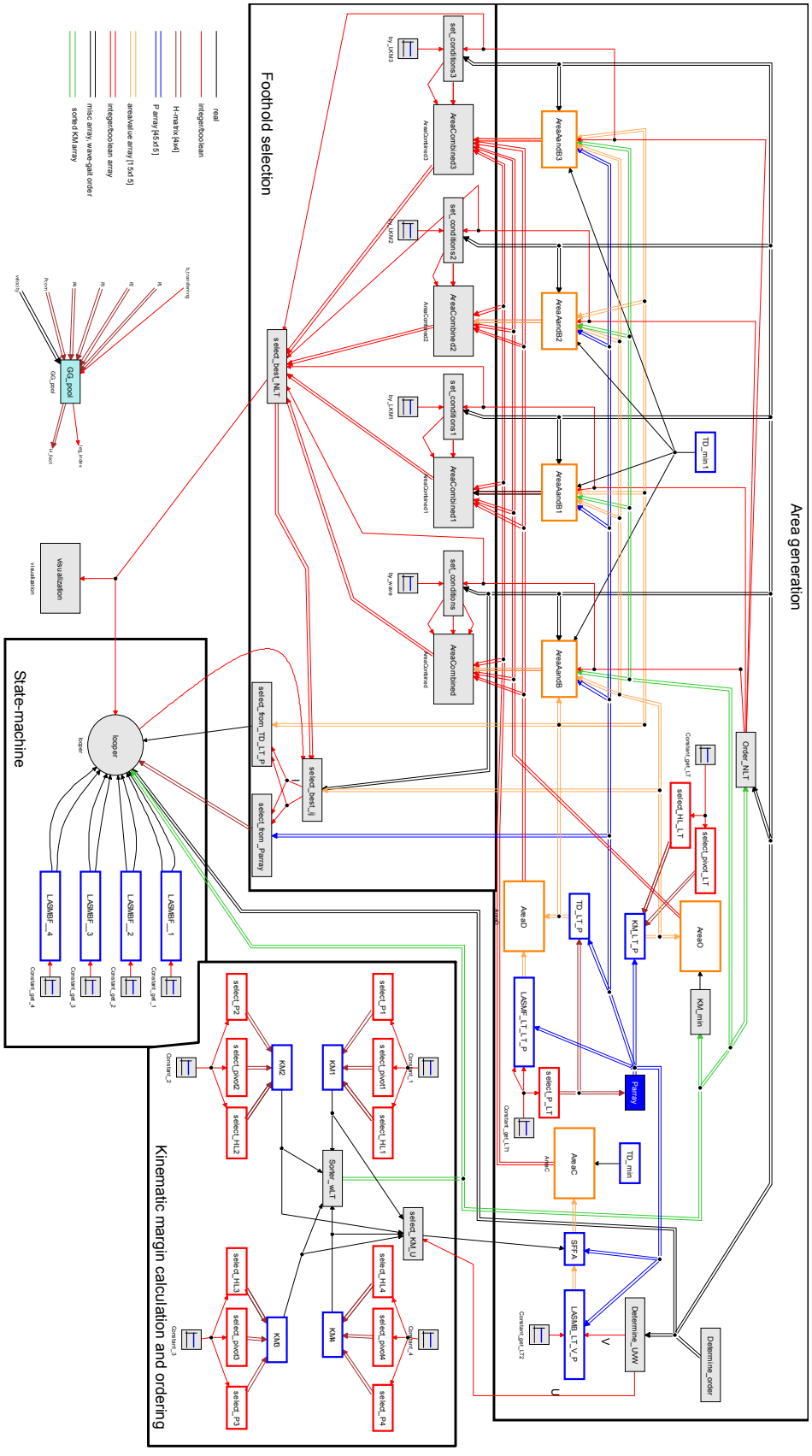
To allow more flexibility, the forced favour of the wave-gait can be removed from the select_best_nlt submodule and leg-lifting check in the looper submodule. Together with an improved foothold selection this will provide the gait-generator with more options during movement.

6.2.9 Terrain adaptability

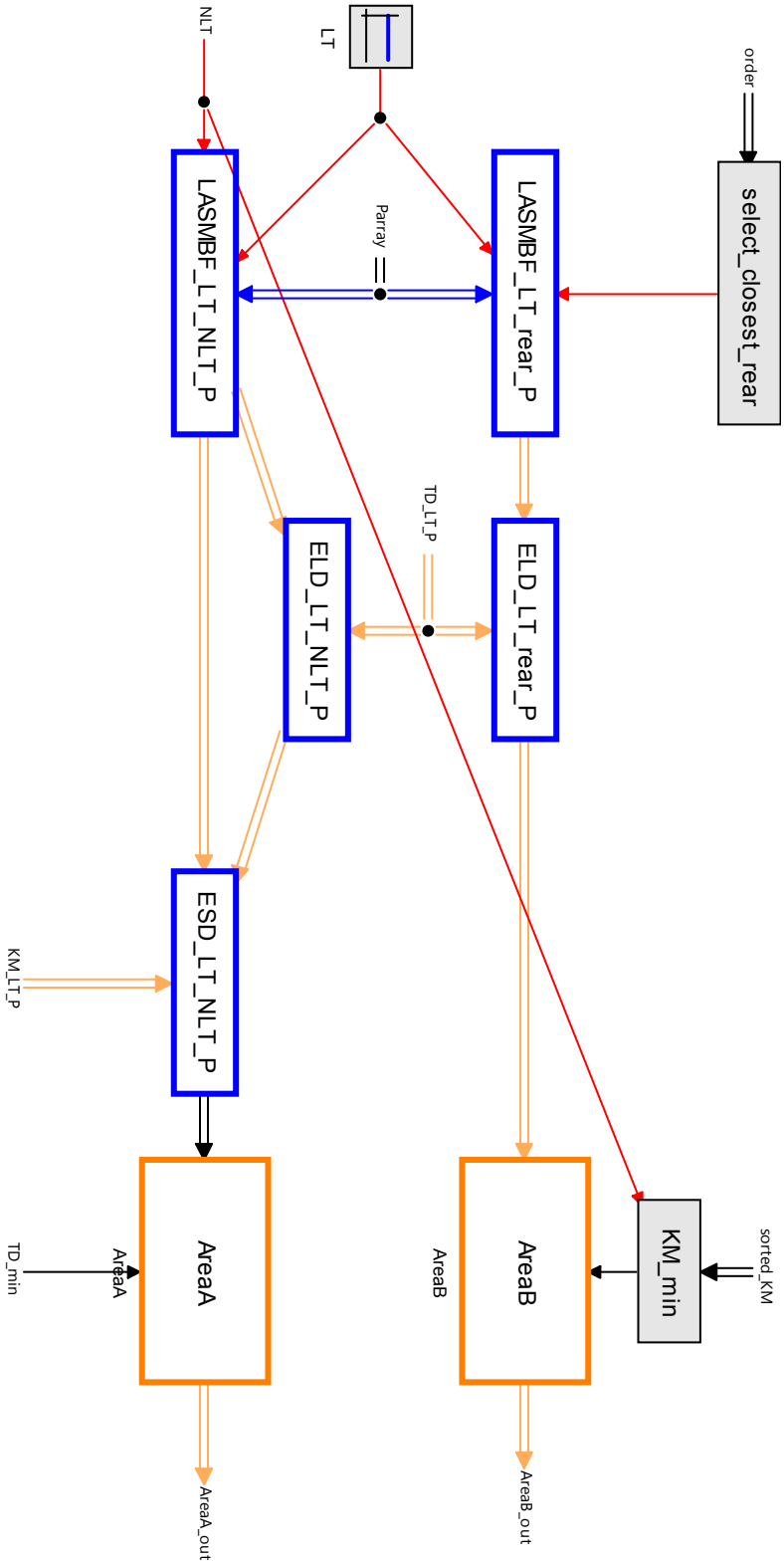
The values for the transfer distance (TD) rely on a very simplistic model of leg-movements. The 20-sim model could be adapted so the values for TD come from a more accurate source like the robot-model itself. When the terrain is inconsistent, leg transfers might be longer or shorter than assumed because the foot was placed higher or lower than expected. A safety margin based on the terrains height uncertainty can be included in the calculation of TD. The model would then effectively be capable of generating stable gaits for inconsistent terrain ((Estremera and de Santos, 2005). Additionally the calculation of the kinematic margin for a leg should consider the z-axis since the reachable area for a foot will change when it is placed higher or lower than the ground-plane at z=0. For full 3D placement of the foot, there is actually a reachable volume instead of an area for which the KM needs to be calculated.

A Appendix 1: 20-sim model

A.1 Full model



A.2 Submodel for Area A and B



Bibliography

Arevalo, J. C. and E. Garcia (2012), Impedance Control for Legged Robots: An Insight Into the Concepts Involved.

Bai, S., K. H. Low and T. Zielinska (1999), Quadruped free gait generation based on the primary/secondary gait, **vol. 17**, no.04, pp. 405–412.

Controllab Products (2008), 20-sim.
http://www.20-sim.com/

De Santos, P. G., J. A. Galvez, J. Estremera and E. Garcia (2003), SIL04: a true walking robot for the comparative study of walking machine techniques, **vol. 10**, no.4, pp. 23–32.

Dresscher, D. and S. Stramigioli (2012), Autonomous robots for dike inspection, unpublished.

Erden, M. S. and K. Leblebicioğlu (2008), Free gait generation with reinforcement learning for a six-legged robot, **vol. 56**, no.3, pp. 199–212.

Estremera, J. and P. G. de Santos (2005), Generating continuous free crab gaits for quadruped robots on irregular terrain, **vol. 21**, no.6, pp. 1067–1076.

Hirose, S., H. Kikuchi and Y. Umetani (1986), The standard circular gait of a quadruped walking vehicle, **vol. 1**, no.2, pp. 143–164.

Inagaki, S., H. Yuasa, T. Suzuki and T. Arai (2006), Wave CPG model for autonomous decentralized multi-legged robot: Gait generation and walking speed control, **vol. 54**, no.2, pp. 118–126.

McGhee, R. B. and A. A. Frank (1968), On the stability properties of quadruped creeping gaits, *Mathematical Biosciences*, **vol. 3**, pp. 331–351.

McGhee, R. B. and G. I. Iswandhi (1979), Adaptive locomotion of a multilegged robot over rough terrain, **vol. 9**, no.4, pp. 176–182.

Pal, P. K. and K. Jayarajan (1991), Generation of free gait-a graph search approach, **vol. 7**, no.3, pp. 299–305.

Yoneda, K., H. Iiyama and S. Hirose (1994), Sky-hook suspension control of a quadruped walking vehicle, in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, IEEE, pp. 999–1004.