ARAMIES: A FOUR-LEGGED CLIMBING AND WALKING ROBOT

Dirk Spenneberg, Andreas Strack, Jens Hilljegerdes, Heiko Zschenker, Martin Albrecht, Till Backhaus, and Frank Kirchner

University of Bremen, Robotics Lab
Faculty of Mathematics and Computer Science
Bibliothekstr. 1, D-28359 Bremen, Germany
spenneberg@informatik.uni-bremen.de

ABSTRACT

This article describes the four-legged walking integration study ARAMIES and the developed control software approach. Furthermore, we analyze a first walking experiment carried out. We explain how CPG-like rhythmic motion patterns can be produced on the basis of Bezier-splines, resulting in a very simple and flexible way to produce complex walking trajectories which can be modulated in phase, frequency and amplitude. In addition, we describe briefly our new RT-micro-kernel concept for programming behavior-based robots which is used in the ARAMIES project. This microkernel combines properties of real-time operating systems with concept of behavior-based programming.

Key words: Walking Robot, Central Pattern Generators, Modular Mechanical Design, Behavior-Based Microkernel.

1. INTRODUCTION

This paper presents the current state of the robotics project ARAMIES (jointly funded by ESA and DLR), which started in April 2004. Its goal is to develop a four-legged walking robot capable of (semi-)autonomous operation in extremely difficult environments, especially very uneven and steep terrain. An interesting application is the in-situ-investigation of slopes of Martian canyons or craters. These sites have proven to be of high scientific interest but are extremely difficult to access.

The major problem is that conventional wheeled rovers like the currently on Mars operating US-rovers are not able to provide this access. Wheeled systems already have problems with obstacles like very small sand dunes, e.g. the US-rover Opportunity was stuck in May 2005 for over a month in a sand dune of approximately 30cm height.

In various tests, walking robots like the SCORPION have already demonstrated that a multipod-robots programmed with a biomimetic control approach can achieve fast and exceptional robust mobility in such rough outdoor environments(Ref. 1). On the basis of the experiences gained with the SCORPION robots new mechanics and electronics have been built in the first phase of the ARAMIES project.

The result of this first phase is the integration study, which is shown in Fig. 1. It is the precursor for the prototype robot, planned to be assembled till the end of 2005. The integration study has been used for first stress tests of the mechanics and the electronics. Moreover, first tests of the software concepts have been carried out.

2. THE MECHATRONICS

The new mechanical design of ARAMIES is aimed at very flexible kinematics, light weight, robustness,
and high dynamic stability. Also easy maintenance and a high MTBF (mean time before failure) were among the important issues addressed.

In order to test different kinematical setups with various DOF (degrees of freedom), the mechanical and electrical design is focused on maximum flexibility. This is achieved by using only three different mechanical stand-alone units and adaptive components as linkage. Thereby time for changing the kinematical setup and servicing is kept low.

The first and most frequent used mechanical unit is the “standard joint” shown in Fig. 2. It is built of a Faulhaber 38 mm motor/gear combination integrated in a pipe of anodized aluminum. Such units can be connected easily by standard flanges. In combination with some custom made adapters, several extremities with different lengths, different DOF, and shapes can be created. The last standard joint of a leg operates as an ankle or as a wrist, depending on whether the robot is walking or manipulating its environment. In combination with the second stand-alone unit, a 22 mm motor/gear pipe copied from the SCORPION system (Ref. 1), a very simple but efficient foot is designed. These two joints combined with three toes can operate as a foot while standing (see Fig. 4). Additionally, the claws can be used for clinging in sandy or rocky soils.

The limbs can be attached directly to the corpus via custom made adapters(cf. Fig. 1). Furthermore, an additional shoulder joint (the third mechanical unit) has been built in order to increase the flexibility of the design (cf. Fig. 2). If the shoulder unit is integrated it can turn around the whole leg. Because the corpus is currently a frame designed with system strut profiles, different sizes of the system can be realized with low effort. All parts together result in a highly flexible robot construction system with very simple means to test out different kinematical setups.

An overview of the technical details of the system is shown in Tab. 1. The peak torque values can be applied for at least one second, and the values for continuous torque express the maximum continuous load where thermal stability is still provided. The average power consumption corresponds to normal walking operation, while the peak power consumption is a theoretical value for the power, which would occur when all motors are producing their maximum torque.

Table 1. Technical data of ARAMIES. The values in parentheses correspond to a new motor type we are currently testing.

<table>
<thead>
<tr>
<th></th>
<th>Length (700 mm)</th>
<th>Width (450 mm)</th>
<th>Height (600 mm)</th>
<th>Weight (28 kg)</th>
<th>38 mm Joint:</th>
<th>22 mm Joint:</th>
<th>Accumulator:</th>
<th>Power Consumption:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Torque (cont. / peak)</td>
<td>Rotational Frequency</td>
<td>Capacity / Voltage</td>
<td>Standby</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8 (13) / 17 (26) Nm</td>
<td>1 Hz</td>
<td>9 Ah / 24 V</td>
<td>~ 20 W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 / 3 Nm</td>
<td>0.5 Hz</td>
<td>Average while walking with current motion pattern</td>
<td>~ 100 W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Theoretical peak</td>
<td>2800 W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The future work on the mechanics will deal with improving stability and weight. In addition to the new mechanics, new electronics have been developed, too.
It is designed to be very modular regarding physical installation and upgrading.

The electronics consist of 5 different components:

1. The Microcontroller Board: The phyCORE MPC565 is an ready-to-insert Single Board Computer populated with Motorola’s PowerPC MPC565 microcontroller and external FLASH-ROM (up to 4Mb) and SRAM (up to 16Mb).

2. The FPGA Board: It includes an XCV600E FPGA from XILINX. It features nearly 1,000,000 XILINX gates. Furthermore, it supports differential signaling which is necessary for the LVDS bus we use to control the motor board with minimum cabling. The FPGA board is responsible for acquiring information from the motor sensors and controlling the DC motors. The controller in the FPGA allows to set the maximum motor voltage which can be used to ensure that the average current drawn by our motors stays within its specification.

3. The Carrier Board: This board serves as a back-plane board, where both, micro-controller board and FPGA board, will be plugged in. It makes all the inputs and outputs of the MPC565 microcontroller available through different connectors around the board and also includes accelerometers and gyroscopes (cf. Fig. 5).

4. The PC-104 Microprocessor Board: This is an extension board making use of the PC-104 bus. It is not yet employed, but in the future it will be used for the higher levels of control. Furthermore, it allows adding further capabilities to the system like GPS, Ethernet, GSM, and a hard disk.

5. The Motor/Sensor Boards: These boards are capable of driving 4 DC Motors of up to 5 A peak current each. Furthermore, they are able to read position, current, and encoder data of the motors. Additional ports are available making it possible to add pressure (in the foot tips) or force sensors in the future. The FPGA board can support up to eight of these Motor/Sensor Boards, thus enabling the management of up to 32 DC motors at a time. At the moment 20 motors are used.

3. THE MICROKERNEL

Behavior-based programming is a very efficient and elegant way of programming autonomous or semi-autonomous systems. It is especially suited for implementing bio-inspired approaches because most approaches assume that all processes(mechanisms) responsible for the behavior of a system are executed in parallel, which is inspired by the massive parallelism in biological systems. Unfortunately, this can be only emulated on single sequential CPU systems. In concepts like Behavior Language (an extension of the subsumption architecture) (Ref. 2), or Process Description Language (PDL) (Ref. 3) and their derivations, this is achieved by using a clocked single-loop for the behavior processes and writing their influences on the actual hardware all at once, e.g., at the end of the loop.

While being well studied these concepts are seldom used in environments where hardware constraints are tight as they are in an extraterrestrial rover. Therefore problems arising from weak computational power and external disturbances of the program flow have been addressed only lightly. To avoid a loop overrun the loops are typically much longer than the maximum required time to execute all processes. This allows later extension and prevents loop overruns but leads to a lot of idle CPU time which could be used more effectively.

Additionally, the above mentioned behavior programming models do not provide an easy mechanism to allow behavior processes to run on different frequencies, which is desirable if they , e.g., deal with hardware of different reactivity. Finally, out-of-order, e.g., asynchronous, execution of certain processes like reflexes are not foreseen in these concepts.

Real-time operating systems on the other hand offer high reactivity and satisfy real-time needs but would reduce the remaining computational resources of our system. Therefore we developed our own microkernel labeled M.O.N.S.T.E.R which features hard- and soft-periodic processes\(^1\), preemptive reflexes, behavior processes, and hardware drivers on different execution frequencies, online-adaptation of these execution frequencies, competition for hardware resources between the behavior processes, and a background process.

In fact the alterations necessary to achieve these fea-\(^1\)That is processes with hard real-time constrains and those without it

![Figure 5. The phyCORE board and the XILINX FPGA board mounted on the carrier board](image-url)
tures are quite few. To achieve the desired tolerance for the case when behavior process execution takes longer than the estimated loop period, a mechanism to simulate the ideal case - that is when all processes could be executed on time – is required. First, some mechanism to decouple the single-loop from the actual execution time of the included behavior processes is necessary, which is achieved by using a timer interrupt with a fixed system frequency.

Now let \( q_i \) be the quantity behaviors, let further \( inf_{b,i} \) the influence of behavior process \( b \) on that quantity and \( w_{b,i} \) the weight assigned to that influence, then the new value \( q(t) \) at time step \( t \) is computed as follows:

\[
q_i(t) = \sum_{b=0}^{n} w_{b,i}(t) \cdot inf_{b,i}(t)
\]

\[
w_{b,i}(t) = \begin{cases} 
  w_{b,i}^{set}(t), & \text{if } b \text{ active in step } t \\
  w_{b,i}(t-1) \cdot \text{dec}(b, i), & \text{else}
\end{cases}
\]

If a process \( b \) was not executed in time-step \( t \) its weight is decayed, otherwise the weight is assigned to its influence will be used. Furthermore a background process has been introduced which will be executed whenever there is spare CPU time, e.g., when process execution takes shorter time than the fixed loop cycle.

This concept however comes with its costs. Behavior processes have to written in a fashion allowing them to be delayed. This is done by taking the actual timeslice passed since the last execution of a behavior into account when calculating new values. But these adaptations arise quite naturally when one understands behavior processes as difference equations. As an example our motion production process uses Bezier curves to describe the joint movements and in dependence of the passed timeslice it will move along these Bezier curves further or shorter. For more information on the M.O.N.S.T.E.R. microkernel see (Ref. 4).

4. THE RHYTHMIC MOTION CONTROL

In our bio-inspired control concept, rhythmic motion is produced by models of “Central Pattern Generators” (CPG). These are combined with posture control primitives. In addition, the developed control approach features a reflex model, which allows to deal with sudden disturbances from the environment (see (Ref. 5)).

A simple model for the functionality of biological CPGs consists of a controller-module and a unit to produce rhythmic trajectories in the joint angle space. For the controller unit already existing standard control approaches (e.g. PID) are suited. The module for the generation of the rhythmic trajectories can be realized by using third order Bezier-Splines. Bezier curves have the advantages of being smooth and to dispose of a controllable gradient at the start- and the end-position of the curve. This is advantageous for DC-motor control.

A Bezier polynomial is described with the following equation:

\[
P = PO \cdot t^3 + P1 \cdot 3 \cdot t^2 \cdot (1-t) + P2 \cdot 3 \cdot t \cdot (1-t)^2 + P3 \cdot (1-t)^3
\]

PO to P3 are points which are describing the curve, P0 and P3 are supporting points and P1 and P2 are control points of the curve. These coherence is illustrated in Fig. 6. A CPG-Pattern \( P \) can be described as a function of partwise fitted together cubic Bezier-polynomials.

A part \( X \) of a CPG pattern is described by the four coefficients of the Bezier-polynom \( k_n(X) \), its length \( l(X) \in \mathbb{N}_0 \) on the x-axis, the phase offset \( \theta(X) \in [0,1] \), its scalability \( S(X) \in \{0,1\} \) and a optional subpart-list, if the part is constructed by subparts. Subparts are an option for defining new patterns on the basis of existing ones but will not be considered in the further text.

A complete rhythmic pattern is described by a list of parts, where the end of the list points to the start of the list. The position-algorithm computes to every time-step \( t \) for the current Part \( X_a \) of the whole pattern \( P \) (consisting of \( n \) parts and with a offset \( \theta \) the following equation:

\[
pos(t) = y(t + \theta) \mod l_P
\]

\[
\begin{align*}
\text{where } y(x) &= \sum_{i=0}^{3} k_i(X_a) \cdot x \\
\text{and } x \in X_a &= [x_0(a), x_1(a)] \\
\text{and } X_a \in P &= \{X_1, X_2, ..., X_n\} \\
\text{and } l_P &= x_1(n) - x_0(1)
\end{align*}
\]
The above description for setting up and computing rhythmic trajectories allows a very efficient computation in addition to a high flexibility. With this rhythmic motion description we can formulate a trajectory only by supporting points and their gradients. To get an even more compact way of describing trajectories, we distinguish two types of supporting points, extreme points, where \( y'(x) = 0 \), have the type \( T(X) = 0 \) and all other points are of type \( T(X) = 1 \). The gradient of the non-extreme points \( (T(X) = 1) \) is set to the gradient of a straight line connecting its two direct neighbor points.

The phase offset and the scalability of each part can be chosen freely. Typically the phase offset is only used for the whole pattern, the scalability is normally used only for a part of the pattern. An example of a pattern generated with this method is presented in Fig. 7. The following parameters are used for this pattern \((X,Y,Type,Scalability)\) : \( \theta = 0 \), \( X_0 = (0,0,1,0) \), \( X_1 = (5,40,0,0) \), \( X_2 = (10,0,1,1) \), \( X_3 = (30,-20,0,1) \), \( X_4 = (50,0,1,1) \).

The shown example is a "Lateral Leading"-pattern, found at the basalar joints of lobsters. The pattern has a swing-period of 10 time steps and a stance period of 40. The whole step-period is 50 timesteps. The swing-amplitude is 40 degrees and the stance-amplitude is -20 degrees.

A rhythmic motion pattern can be changed in its period, its amplitude, and its phase. Fig. 8 shows an example, where the whole pattern is newly scaled at time step \( x = 50 \). Noticeable is that the first part does not scale because it was configured as non-scalable: \( S(X_0) = S(X_1) = 0 \).

The first part is the swing phase. In most animals, especially in invertebrates, the swing period changes only slightly when the step period changes. In contrast, the stance period is linearly correlated to the step period. Overall, this results in high stability of the walking system since because of this feature and the phase offset between the legs, the invertebrate has as many legs on the ground as possible at a given speed. By introducing the Scalability-Feature in our model, we can easily transfer this feature to our robot. The same can be done with the amplitude, as shown in 9. Recapitulating this cpg-model allows the production of rhythmic and smooth motion patterns on the basis of Bezier-splines, which can be described very compactly by their supporting points and their types. These patterns can be modulated in their phase, amplitude, frequency. The scalability of parts of a pattern can be set.

The usefulness of this cpg-model for producing motion patterns for the 4-legged ARAMIES robot is shown in fig. 10. This plot shows the trajectory of the knee of all four legs. The trajectory and the phase offset is nearly identical to data from in-depth experiments on the standard walking gait of cats found in (Ref. 6). This pattern consists of the following support points \((X,Y,Type,Scalability)\) : \((0,0,0,0), (20,80,0,0), (40,20,0,1), (70,30,0,0), (100,0,0,0)\).

In this approach also the combination of CPG-patterns or a smooth fading between different simultaneously active patterns is possible. How this
is done in real animals is still under investigation, but a possible model would be the following. The activation of the hitherto pattern is decreasing over a certain period and the activity of the new pattern is increasing. As long as more than one pattern is active, the current position is computed from the positions of both pattern weighted with their activity. More details on this approach can be found in (Ref. 5).

5. FIRST STEPS ON 4-LEGGED CPG-BASED WALKING

We conducted walking experiments with the configurable integration study shown in Fig. 1. For the tests we used legs as depicted in Fig. 11, where we removed the shoulder joint in order to loose some weight while maintaining mobility.

The hip and knee joints are controlled by CPGs inspired by results of biological research on the walking of cats (Ref. 6). The positions of these joints during three steps (depicted in Fig. 12(a) and 12(b)) resemble the movements of a cat. Due to different sizes and lengths of the limbs slight modifications had to be made to achieve a stable walking behavior of our integration study. The diagrams show the generated CPG as the “desired” positions and the “actual” positions that really occurred during the experiment. There is only one place in the stance phase of each step where both differ significantly, namely the end of the decreasing part of the hip positions. The difference occurred when the leg had contact to the ground and thus supported the weight of the robot. Here, the joint was not able to pull the robot forward as much as the CPGs patterns suggested. The ankle and foot joints were not driven by CPGs, but controlled in regard to the other joint positions. The goal was to keep the foot part of the leg parallel to the ground if possible. In order to do so we calculated the angle of the ankle depending on the measured angles of the hip and knee joints. The foot joint remained mostly straight. In this case the foot as a whole was parallel to the ground. Only when the ankle had to move farther than it could physically, the foot joint moved also so that at least the toes of the foot where parallel to the ground. This behavior is expressed in the curves shown in Fig. 12(c) and 12(d).

In (Fig. 13) one can see that clearly that the knee-joint performs most of the work. The other three joint feature comparatively low energy consumptions. The prominent peaks of diagram 13(a) correspond directly to the previously discussed problem of the hip joint to follow the CPG in the stance phase.

6. DISCUSSION

The results of the experiment presented above suggest a number of further steps in the development of our four-legged walking behavior. Considering the important aspects of power consumption and mechanical stress it is desirable to slightly modify the motion behavior such that the knee is less utilized. However, it will still remain the most stressed part of the leg. Paying regard to this fact, we intend to use a motor with almost the double tolerable torque than the standard motor described in sec. 2.

Currently we are using a static P-Controller (programmed in the FPGA) to drive the motors, but we are progressing to a PID-Controller which can be reprogrammed online. This will enhance the problem of the hip following the CPG. Due to the re-programmability, we can adapt to different load
Figure 12. Desired and actual joint positions of the left front leg during walking.

Figure 13. Motor currents during walking.

states. For example, we can choose small coefficients in the swing phase of the system, in order to have a very soft contact when the foot touches the ground or an obstacle. And we can stiffen the leg again in the stance phase by using larger values for the coefficients when the leg is under load. This might already work well with a re-programmable P-Controller.

Furthermore, we will start to integrate the values from the gyroscope and the tilt sensor integrated into the carrier board to improve the walking and to compensate for tilting of the system, which occurs due to the high weight when a leg is lifted. We intend to use additional posture control behaviors to adjust the direction of the movement of the robot as well as its tilt. The posture control is already integrated in our control architecture (see Ref. 5). As a simple concept, one can think of the posture control as a system which adds a value
to the position value given by the cpg-models. The posture control applies an offset to the rhythmic motion control. Thus it is possible to change the posture while the system is walking. This feature was already presented in the SCORPION robot, which was thereby able to walk up steep uneven slopes (up to 35%) or to walk up steps while keeping the body in parallel to the ground.

7. OUTLOOK

The first prototype robot, which will have reduced weight, a new closed corpus, and legs with different motor/gear-combinations in response to the different load on the joints, will be assembled till the end of 2005. In 2006 we will focus on the additional behavior processes for stable walking in rough terrain. At the end of the project in Nov./Dec. 2006 a deployment in a realistic environment which resembles a Martian slope is planned.

ACKNOWLEDGMENTS

The presented work is sponsored by the Deutsches Zentrum fuer Luft und Raumfahrt (DLR grant no. 50JR0561) and European Space Agency (ESA contract. 18116/04/NL/PA). Furthermore we like to thank all members of the Robotics Lab for their contributing work, especially Dr. Stefan Bosse for his work on the electronics.

REFERENCES


