Simulation, gait generation and embedded control of the Amru5 six-legged robot

Quentin Bombled[1] *and* ***Olivier Verlinden***

A rigid body dynamic model of a six-legged robot with 18 dofs has been developed, including gravity, contact interaction with the ground, actuators and controllers. This model is used as a benchmark to develop a gait algorithm before implementing it on the robot. The latter computes the joint references which represent inputs of the controllers. Success of the real implementation proves the efficiency of the approach.

Key words: *robot, dynamic simulation, DC motor control, decentralized control, gait generation*

Challenge of legged robot

For many years, walking robots are a wide challenge for researchers. Even if they are less fast, energetically greedier and more complex at command level than wheeled robots, they have the ability to get over obstacles and to move on unstructured ground. This aptitude makes them potentially interesting to replace the human in hardly tasks, like demining, exploration in hostile conditions, nuclear site maintenance or building guard. The gait generation and its adaptation to rough environment (collisions detections, holes, ...) is more than ever a present and relevant topic for engineers. A huge number of walking robots exists, each of them having their own structure and their own usefulness. Two main approaches have been envisaged to make them walking (Delcomyn, 2004) : the first one is called the “engineering” approach. The leg motion is a consequence of the references computed by a central agent imposing a motion to the main body of the robot. One of the first implementation was on the DANTE II robot, on the occasion of the Alaskan volcano exploration in 1994. Recently, (Porta et al., 2004) worked in this way on LAURON III and GENGHIS robots. The second approach is “biologically” inspired : initiated by Wilson in 1966 (and is still used nowadays), it is based on the insects observation. Global motion of the robot is here a consequence of the leg motions. If a central agent exists, it just arbitrates when conflicts arise. This progressive reflexion allowed to define types of gait, basis motions of the legs (produced by CPG's for example), structures for control, enhancement of mechanical design and so on. (Parker, 1996) with cyclic genetic algorithms or (Cruse, 2006) with neural networks, and many other research teams tried to mimic the animal behaviour in different ways on walking machines like HANNIBAL, STIQUITO or TARRY II. Even if these implementations were successful, there are very few complete dynamic models of legged robots. Most of them are purely kinematic. Nevertheless, this kind of model can be greatly useful for selecting and designing the controller.

In the following, Section 2 will give details about the dynamic model of AMRU 5. Section 3 describes the gait algorithm. Section 4 outlines the hardware implementation. Finally, Section 5 discusses the results and concludes.

Multibody model of AMRU 5

The robot structure

AMRU 5 (fig. 1) is a six-legged robot with 18 degrees of freedom, initially devoted to demining mission. It has been built by the Royal Military Academy of Belgium, within the framework of the AMRU project (Autonomous Mobile Robot in Unstructured environment). The robot is about 30kg heavy and has an outer diameter of about 1,2m. Each dof is controlled by a DC motor. The leg is based on a pantograph mechanism actuated by 2 DC motors. Hence, foot (extremity of the leg) can be moved independently according to a radial and a vertical motion. Tangential motion is allowed by the rotation about a vertical axis.

[1] *Quentin Bombled, Olivier Verlinden*, Department of Theoretical Mechanics, Dynamics and Vibrations (TMDV), Faculty of Engineering of Mons, Mons, Belgium, quentin.bombled@fpms.ac.be
(Review and revised version 23. 2. 2010)

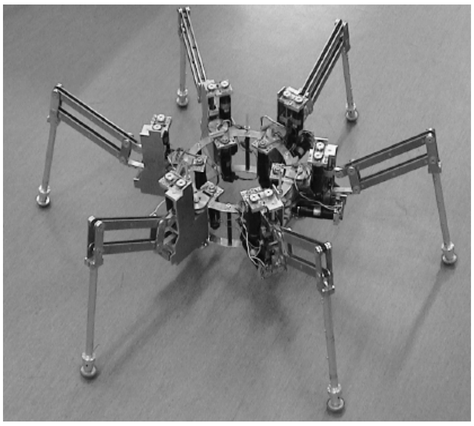

Fig. 1. AMRU 5.

Multibody model

Multibody model of the robot has been set through the generalized coordinates approach. A complete model has been established in (Verlinden, 2005) by means of the C++ library EasyDyn, developed by the department of TMDV. First step is to describe the kinematics at position level of all the bodies, in terms of homogeneous transformation matrices (1).

This tool gives position and orientation of a frame *j* with respect to a frame *i* : $R_{i,j}$ is the rotation tensor matrix describing the orientation of frame *j* with respect to frame *i*, $\{r_{j/i}\}_i$ is the coordinate vector of frame *j* wrt to frame *i,* projected in frame *i.*

$$T_{i,j} = \begin{pmatrix} R_{i,j} & \left(r_{j/i}\right)_i \\ 0\,0\,0 & 1 \end{pmatrix} \tag{1}$$

Second step consists in defining the whole kinematics in terms of velocities and accelerations : a Mupad routine called CAGeM (Computer Aided Generation of Motion) performs the first and second time derivatives of the position and projects them in the global frame. It also creates a C++ application file. After compilation, the user can launch an executable able to build equations of motion using the D'Alembert's principle, which leads to the traditional set of equations (2):

$$M(\underline{q})\underline{\ddot{q}} + \underline{h}\left(\underline{q},\underline{\dot{q}}\right) = \underline{g}\left(\underline{q},\underline{\dot{q}},t\right) \tag{2}$$

where $\underline{q}$ is a column vector gathering the configuration parameters, $M(\underline{q})$ is the mass square matrix, $\underline{h}()$ gathers the contributions of centrifugal and Coriolis forces, and $\underline{g}()$ those of applied forces. These equations are solved following a Newmark integration scheme.

Multiphysics model

Force contact with the ground is modeled through the penetration *p* of the feet into the ground. Important stiffness and damping values are assigned to simulate the soil. Equation (3) represents the normal contact force F_n; tangential forces is expressed according to the latter (4):

$$F_n = Kp^{pk} + Cp^{pd}\left(dp/dt\right) \tag{3}$$

$$\begin{aligned} \underline{F_t} &= -fF_n\left(\underline{v_g}/v_{glim}\right) \quad if\left(\underline{v_g}\right) \le v_{glim} \\ \underline{F_t} &= -fF_n\left(\underline{v_g}/\left(\underline{v_g}\right)\right) \quad if\left(\underline{v_g}\right) > v_{glim} \end{aligned} \tag{4}$$

Because of the great versatility of the EasyDyn framework, classical equations of DC motors and PI controllers have been implemented in the robot model. The control loop of one DC motor is depicted in fig. 2. There is no coupling in the motor control thanks to the pantograph mechanism. Joint tracking trajectories are computed by the gait generation algorithm (see Section 3). The pertubation torque τ_f is caused by friction forces and intermittent contact with the ground.

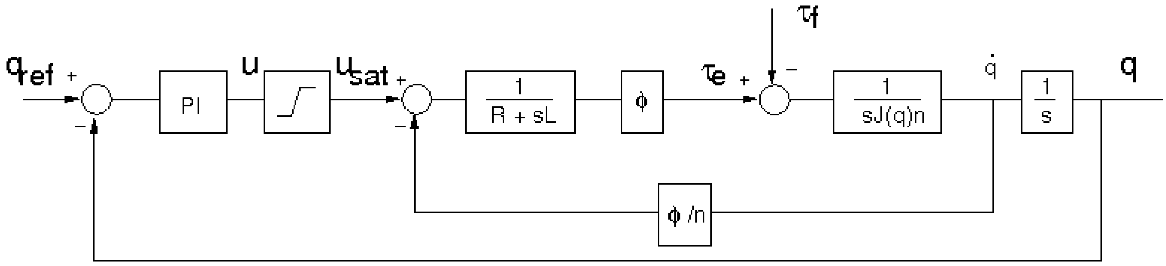

Fig. 2. PI control loop.

Friction effects have also been modeled, by implementing the LuGre model, defined by (5) [Canudas de Wit 1995]:

$$\begin{aligned} F_f &= \sigma_0 z + \sigma_1 (dz/dt) + \sigma_2 v \\ dz/dt &= v - \sigma_0 z(|v|/g(v)) \end{aligned} \qquad (5)$$

where the friction force F_f is function of the relative velocity v of the contact surfaces, and of the deflection of bristles z. Indeed, this model takes into account friction phenomena in pre-sliding phase, where stiffness between asperities are modeled through the $\sigma_0 z$ term. Dynamic of the bristles is described by the *dz/dt* term. These equations have been successfully implemented in the robot model, but with rough values for parameters. Their identification is still under refinement.

Gait Generation

Principle

The fixed tripod gait has been implemented in the simulations (fig. 3). It simply consists of successive swing and support phases of the leg triplets (0-3-4) or (1-2-5).

The following scheme is applied to move AMRU 5: at each time step of the control, the state of the leg is checked. If it is in support phase, then an inverse kinematics algorithm is executed, from the desired trajectory of the main body. If leg is in swing phase, then the same algorithm is applied but according to a fictive motion of the body, which corresponds to the inverse motion of the desired one.

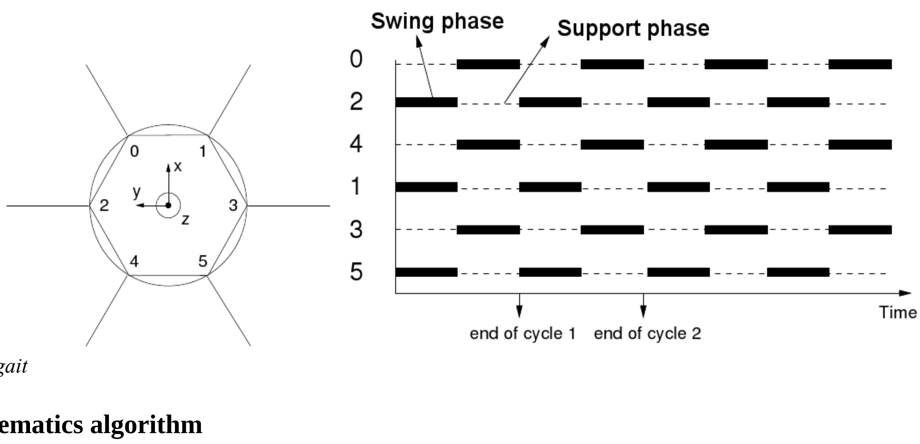

Fig. 3. Tripod gait

Inverse kinematics algorithm

By using the homogeneous transformation matrices, we develop a general algorithm valid for any type of leg structure, under the following assumptions: 1) no sliding at ground interface, 2) flat ground, 3) no obstacles. Fig. 4 shows the principle of the algorithm for one leg:

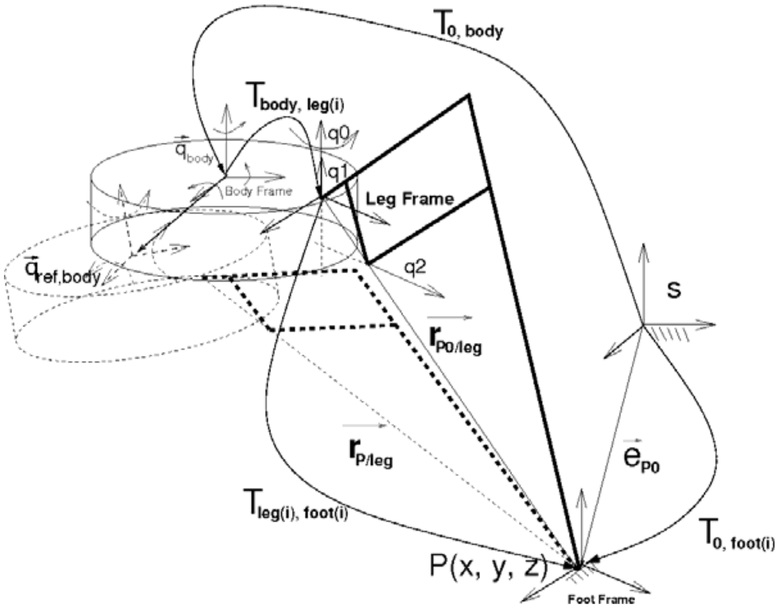

Fig. 4. Inverse kinematics principle

Consider the actual body position in plain line, and the desired position in dotted line. We define the foot position in the local leg reference frame as $\underline{r}_{P/leg}$. It is given by (6):

$$\underline{r}_{P/leg} = \left(T_{0,body} * T_{body,leg}\right)^{-1} * \underline{e}_{P0} \qquad (6)$$

where the homogeneous transformation matrix $T_{0,body}$ defines desired the position and orientation of the body (function of $\underline{q}_{ref,body}$), and $T_{body,leg}$ is a constant, proper to the architecture of the hexapod. Because there is no slip of the foot during the motion, $\underline{e}_{P0}$ (position of the foot in the global reference frame) is a constant. Once $\underline{r}_{P/leg}$ is known*,* a Newton-Raphson algorithm is applied to (7) to determine the leg joint references.

$$\underline{r}_{P/leg} = \underline{f}\left(\underline{q}_{leg}\right) \qquad (7)$$

The same algorithm is applied to the legs in swing phase, except that $T_{0,body}$ is computed from a *reverse motion*. A triangular profile is then assigned to the vertical motion of the leg to allow the swing.

Hardware implementation

Decentralized architecture has been implemented. Each leg is controlled by a *SBC65EC* slave board. It is based on the *PIC18F6627* microcontroller. Each slave controls 3 DC motors actuating the joints. The CPU of the master board is a INTEL DUAL CORE 3GHz, with 1.5Go RAM. The latter handles the gait generation. Communication between master and slaves is bidirectional, and realised thanks to the UDP/IP Ethernet protocol. Control loop is rated at 100 Hz, and precise clock tick is set with the standard timer C library. Master sends a broadcast packet to all the slaves. When PI control is done, position, current and voltage measurements of the motors are sent back to the master.

Results and conclusion

Fig. 5 shows the X reference imposed to the main body (X_{ref} – 0.03 m/s), the reverse motion of the fictive body ($X_{ref,reverse}$), the simulated motion of AMRU 5 (*X motion*) and the leg 0 rotational joint reference computed by the inverse kinematic algorithm. This proves the correct working of the algorithm. Direct implementation on the robot would have been difficult; thanks to the model again, risk of damages at joint level is decreasing because simulation allows to verify if the 18 joints are staying inside their working space. With a direct implementation, the monitoring of the six legs would have been impossible for only one observer. Further researches will lead to friction identification and enhancement of the gait algorithm to lighten the lag between the reference and the real motion of the robot.

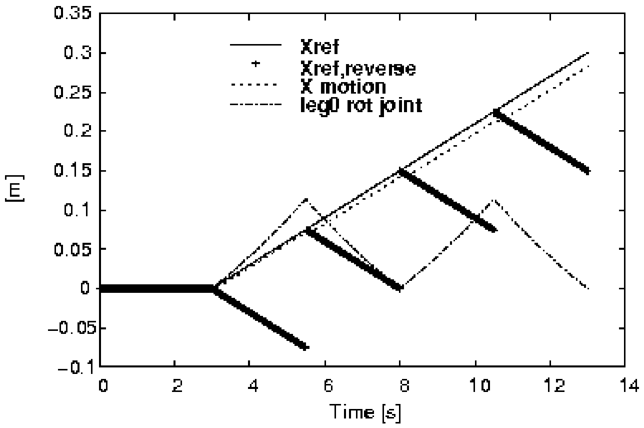

Fig. 5. Motion along X-axis.

Acknowledgement
The authors would like to acknowledge Prof. Y. Baudoin, from the Royal Military Academy, for the lending of AMRU 5.

References

Delcomyn, F.: Insect walking and robotics. In *Annu. Rev. Entomol, 2004, vol. 49, pp. 51-70.*

Porta, J. M. et al.: Reactive free-gait generation to follow arbitrary trajectories with a hexapod robot. In *Robotics and Autonomous Systems*, *2004, vol. 47, pp. 187-201.*

Parker, G. B. et al.: Cyclic Genetic Algorithms for the Locomotion of Hexapod Robots. In *Proceedings of the World Automation Congress, Robotic and Manufacturing Systems*, *1996.*

Cruse, H.: Control of Hexapod Walking in Biological Systems. In *Adaptative Motion of Animals and Machines*, *2006, pp. 17-29. ISBN 978-4-431-31381-6.*

Verlinden, O. et al.: Open source Model of the AMRU 5 hexapod robot. In *International Symposium on Measurement and Control in Robotics*, *2005.*

Canudas de Wit, C. et al.: A new model for control systems with friction. In *IEEE Trans. on Automatic Control*, *1995, vol. 40, pp. 419-425.*