A Coginitive Architecture and Simulation Environment for the Ptinto Robot

Pablo Muñoz, María D. R-Moreno
Departamento de Automática, Universidad de Alcalá
28805 Alcalá de Henares (Madrid), Spain
Email: {pmunoz,mdolores}@aut.uah.es

Pablo Gallego, Bonifacio Castaño
Departamento de Matemáticas, Universidad de Alcalá
28805 Alcalá de Henares (Madrid), Spain
Email: bonifacio.castano@uah.es

***Abstract*—In this article we present a cognitive architecture and we describe some algorithms for simulating the movement of a hexapod robot (a six legs spider) who has been designed at the Astrobiology Center (CAB) of the Institute of Aerospace Technology (INTA) in Madrid (Spain). The robot is called Ptinto and its aim is to explore the Tinto river and the surrounding areas.**

The cognitive control architecture wants to control Ptinto in order to make an autonomous exploration of the environment. It is structured as a three layer system, using a planning and scheduling component with long-term memory and basic learning possibilities, and an executor that implements some reactive behaviours.

The simulation suite is specifically adapted to the geometric characteristics and dynamics of this particular robot. It includes both a standalone simulation test-bench (for kinematics designs) and an environment to interact with the control architecture.

***Index Terms*—Autonomous agents; cognitive architectures; planning and scheduling; simulation.**

I. Introduction

The Ptinto robot is being developed for the in-situ analysis of the shallow and inaccessible areas of the Tinto River at Huelva (Spain). This river is a unique place for the study of biodiversity in extreme conditions (high pH and heavy metals). Besides in the last decade the Tinto River and the surrounding areas has been the focus of study of NASA scientists to find similarities to the Martian surface. In that sense, this place has very good conditions to test robots for future use on Mars performance.

Therefore, on the one hand, the Ptinto project aims to create an autonomous and intelligent robot capable of moving over the uneven surface of the Tinto River and whose technology could be used, on the other hand, on future missions to Mars. Legged robots are a useful alternative to the traditional and easier to implement rovers moving on wheels or tracks. The main reason is that rovers are inefficient when ground conditions are unfavorable due to its irregularity, viscosity or instability (e.g. in sandy or boggy regions). Legged robots, however, are able to operate in this type of surfaces where the rovers are useless. For this reason, walking robots are used in many applications because of their motion characteristics: adaptability to all fields, greater ability to overcome obstacles and high mobility.

Spider robots, in particular those endowed with six legs, raise a great number of new and different challenges to those robots that move through rolling mechanisms. In general, the ability to function in adverse terrains is the result of the high number of degrees of freedom. This number is in direct correspondence with the load of joints in his legs that are usually at least three. It is obvious that the control and management of a system of this kind has a very high complexity. Besides the inherent difficulty of efficiently handling a robot of this class, we have to deal with the sensitivity of its mechanisms, engines and multiple joints. This means that, for example, it will be necessary to ensure that the strength twisting their joints do not ever exceed a certain threshold of safety.

The paper is structured as follows. First, a brief review of the state of the art of cognitive architectures is provided. Next, section III presents our proposed cognitive architecture for Ptinto. Then, in section IV the simulator suite is described. Finally some conclusions and future work are presented.

II. Cognitive architectures

In the last years many architectures have been designed for control advanced robotics systems. Many of these architectures take a human as the main source of inspiration for developing the architecture, or even try to model human cognition; so that these systems have become known as cognitive architectures. As described in [1], a cognitive system is that who *exhibits effective behaviour through perception, action, deliberation, communication, and through either individual or social interaction with the environment*. This implies, in general, that a cognitive system must be able to operate on unknown circumstances or events for which the system was not designed. Therefore, it is desirable that a cognitive system is resistant to uncertainty

Typically there are two main cognitivist paradigm: (i) cognitivism based in the representation and processing of symbolic information, and (ii) emergent systems, focused primarily on principles of self-organization. Our work is encompassed in the first group, since we have a single heavy agent capable of handling symbolic representations of the environment.

In [2] considers the capabilities that a cognitive architecture should support. These are basically (i) a long-term memory to store relevant information about beliefs, knowledge and goals of the system; (ii) a well-organized mental structures that are representation of the contents of the memory (often related with human-machine interface -HMI-); and (iii) functional and learning processes which operate on the mental structures. However, depending on the system purpose, a cognitive system must support abilities like recognize and classify elements in its environment, predict its actions effects or reasoning about how to solve a situation.

Currently, there is a lot of projects developing on cognitive systems. For example the Soar framework [3], [4], which seeks to achieve a general cognitive architecture using explicit production rules to govern its behaviour, and problem solving to reach a goal state. It also implements learning and interaction with the environment capabilities. Similar to Soar is ACT-R [5] based on how human cognition works. It uses models that automatically produces a step-by-step simulation of human behaviour that can be compared with experimental results. Another architecture centred around unified amodal representations and mainly dealing with processing of high-level information is ICARUS [6], [7], whose main difference is its focus on the organization, use, and acquisition of hierarchical knowledge structures.

Other cognitive architectures are more focused on performance and integration with robotics, like 3T [8], which uses three levels of abstraction and description languages in order to coordinate a planful activites with real-time behaviours for dealing with dynamic environment. Like 3T is the developments under the PACO-PLUS european project, who made a three-level cognitive architecture [9] that uses stereo vision coordinated with a symbolic planner to operate a six axis robotic arm. With an underemphasized interest of representations and more oriented towards reactive behavior, there is Subsumption architecture [10], [11], that manages the execution flow through interconnected state machines.

III. Proposed architecture

The proposed cognitive architecture for the control of Ptinto hexapod robot corresponds to a hybrid three layers (3T) system, in which top tier will be in charge of the deliberative process, long-term memory and learning process as a function of events that occur in the environment. The middle level or execution system also has a short-term memory, as well as a series of rules that trigger the reactive behaviour implemented, in order to response in a short time to eventual situations that may occur in both the environment and in the internal state of the robot. Finally, the low level or functional level, is responsible for providing the functionality of the robot to the top layer, and relay the information collected by the sensors. At this time, the robot only has three contact sensors on each leg, distributed so that it can detect contact with the ground and obstacles that is unable to overcome. A conceptual vision of that architecture can be seen in Fig. 1.

Next subsections aim to detail the most interest elements of the architecture, that is the knowledge base, the learning process and the reactive behaviour. A detailed information about the technology used for each layer and the models deployed can be found in [12].

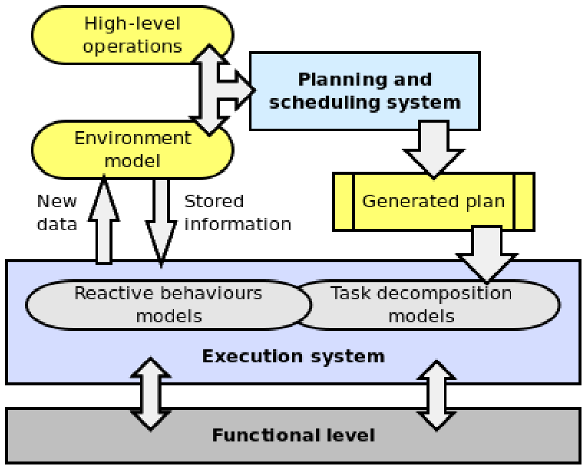


Fig. 1. Proposed architecture preview

A. Knowledge base

In the designed architecture the knowledge base is located in the high layer. The model and operations in the knowledge base are governed by a planning and scheduling system, which will be responsible for obtaining a route that allows the robot to reach one or more interest points. In order to do this, the knowledge base has a three-dimensional model of the environment in which the robot is. This model represents the position of the robot and the obstacles who are known a priori. Also there is the information of the aridity of the terrain point to point, so the robot can adapt his movement to the soil conditions at any time.

For operating with the knowledge base, there is a set of operators modeled as a function of both the environment parameters (distance, altitude, aridity and obstacles), and the constraints under the robot is (available energy or capacity to overcome certain terrain), so that the planning and scheduling system can operate onto the knowledge base and obtain a route among the current position and the desired interest points. If there are several points to reach, a penalty metric can be defined to allow the robot to decide whether to omit a point, in function of the balance between the cost of achieving that goal and the penalty for no achieving it.

B. Learning process

Since the robot's sensors are limited to three contact sensors placed on each leg, the possibility of learning is limited to the ability of detecting unknown obstacles or updating aridity data of the terrain. Both cases are preceded by an unexpected behaviour on the plan generated by the planning system, and detected by the performance monitoring undertaken by the execution system. Prior to the update of the knowledge base, will take place a reactive behaviour as discussed in the next subsection.

In the case the executor notifies a new obstacle, the system will stop the current plan, because this event means that the current path is cut by an obstacle that does not appear in the knowledge base. This obstacle will be included in the knowledge base and the planning system will seek a new route from the current position and environment status. If a modification is notified about the aridity of the terrain, the high level layer updates the aridity data of the current location, but it does not require to produce a new plan, because typically the reactive behaviour is enough to overcome that eventuality.

C. Reactive behaviour

The executor is in charge of controlling the system, so that is able to read the plan obtained by the planning and scheduling system and break down every action in an ordered sequence of commands executable by the functional layer. It also monitors the permanent command execution with the objective of verifying that the commands are properly executed. But because the environment information may be incomplete or may have changed, the executor has some reactive behaviour to respond quickly (and without interference from the upper layer) to a few unexpected events.

These events fall into two groups: (i) incomplete or incorrect information in the knowledge base and (ii) failures in commands execution. In the first group are those who previously commented: unknown obstacles and changes in aridity. An unknown obstacle implies that the robot stops his movement, updates the knowledge base and requests a new route to the planning system. The second case adjusts the parameters that govern the movement of the legs, so that the robot can continue his movement. Also the executor updates the aridity of the point at which the robot is. Because the modification of the parameters can be ineffective, this behaviour can be performed iteratively if a failure is detected in the new movement (and there is not obstacle detection), ending the iteration when the movement is correct or a parameter reaches its limit. If a parameter reaches the limit, the system determines that at this point is an obstacle and acts accordingly.

For failures not caused by the environment, there might be severe failures of the system, which involve moving the system to a safe state and request human intervention. Another possibility is that the duration of the action execution in progress exceeds the time established by the planning system, in this case the system attempts to determine whether the failure type is as the previously mentioned, treating the event as if it had found an obstacle in another case.

IV. PTINTO SIMULATOR SUITE

In order to test the cognitive architecture, we have developed a simulation environment that is specifically adapted to the geometric characteristics and dynamics of this particular robot. The whole simulation system has been developed using MATLAB [13], [14].

The main problem arising in the operation of legged robots is the following: each time the system of locomotion is placed in a certain position it is necessary to determine, as accurately as possible, the performance of each one of the engines, which power their joints, for a configuration of its feet satisfying a set of constraints. In general, these conditions are the number of points of contact with the ground and maintaining balance throughout the range of motion. This problem, known as inverse kinematics, admits an algebraic approach but its resolution can present a very high complexity.

This means that in all robotic projects, whatever their purpose could be, is always required a prior development of a simulation system. The simulated program would have to consider, in advance, all aspects of the robot behaviour. In the case of robots that move by feet and are designed to work in adverse conditions, the simulation becomes more complex due to the need to simulate, not only the device that is designed, but all the features of the environment where it is going to be used. A common way to address this objective is creating various ground models, starting with very simple ones but that will be progressively more complicated.

Apart from that, a simulation system for the operation and performance of a spider robot must be able to anticipate, solve the problems and check the feasibility of the solutions obtained in all reasonably foreseeable situations that may occur. It is its duty to verify that the tracking, performance and control systems are working adequately according to the needs and expectations of the project.

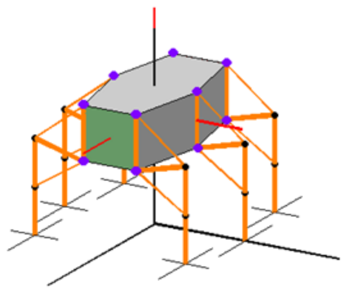
In this section we present a specific simulation system for the elementary movements of Ptinto. We describe its physical and geometrical layout with particular attention to its size, degrees of freedom and range of motion. In the simulation, we use two different reference systems. The global one that would be associated with the ground, and a local one embedded with the robot. Next we establish the two basic types of transformations that relate the positions of the angles of each of its legs and the foot position both directly and vice versa. We also present simulation algorithms for translation and rotation movements on flat ground.

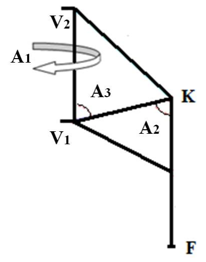
A. Geometric description of the robot

In this section we describe the layout of Ptinto. Figure 2(a) displays a schematic representation of the robot in its resting position. As it can be seen, its body is a hexagonal prism and each one of its legs is located at the prism vertical edges. The vacuum inside the body will be used to place both components that allow for movement and the potential burden to carry in terms of their use. In the rest position, including the legs, the total length is 168 cm, the maximum width 233 cm and its height is 143 cm.

Each one of its six legs consists of two straight sections of fixed length joined by other two sections of varying length. The change in length for each variable component is achieved by a step-by-step motor. These changes allow the leg to go up and down. Furthermore, the connection between the body and each leg has a third engine that changes the angle between the plane of the leg and the body, allowing the leg moves forward or backward.

Figure 2(b) shows a schematic representation of a leg with the denomination of its main elements. It is important to notice that all the elements of each leg are always in the same plane. V_1 and V_2 points are the vertex of the body where the leg is connected and K and F points are named “knee” and “foot”.


(a) Robot



(b) Leg

Fig. 2. Ptinto layout

Thus, the three angles are the three degrees of freedom of a leg and determine its position perfectly. This means A_1, A_2 and A_3 determine the foot coordinates. In the other way around the three foot coordinates can be the three degrees of freedom that bind the angles values.

Table I shows the range of possible values for A_1, A_2 and A_3 angles. The angle A_1 is assigned a positive value if the leg is in an advanced situation relative to the body and negative in the opposite case.

TABLE I
ANGLES RANGE

Angles	A1	A2	A3
Minimum	-30	60	60
Maximum	30	90	90

B. The reference systems

To control the movement, location and spatial orientation of a robot like Ptinto we have to use two different rectangular coordinate system: one of them called "local system" is fixed in the body of the robot and the other known as "global system" is fixed in the ground, that is, the scenario where the robot develops its movements. Both systems are shown in Fig. 3.

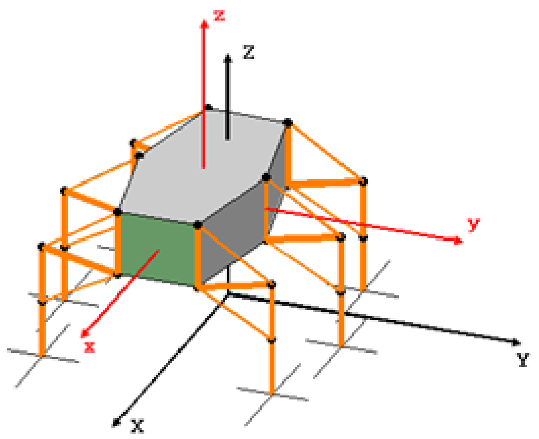


Fig. 3. Reference systems of Ptinto

The local system (shown in lowercase) has its origin of coordinates at the geometric center of the body of the robot and its three perpendicular axes coincide with the three principal directions. The global system (shown in block letters) corresponds to a fixed Cartesian system in the scenario in which the robot moves [15].

In the starting position, also called rest position, each one of the angles is assigned the value of the center of its total range and the robot's feet are supposed to be supported within the $Z = 0$ plane of the global system. Then the global and local coordinate axes are parallel, have the same orientation and the origin of the local coordinate system in relation to the global system is

$$(0, 0, h)$$

where h is the height of the center of the body of the robot over the $Z = 0$ plane in the global system.

In general, the changes and calculations upon the values of the legs angles and the corresponding foot positions are performed in the local system and then transformed into the global system. In general, changes in the values of the angles of the legs and the corresponding relative positions of these are performed in the local system and then move into the global system. This requires always knowing the position of the local origin of coordinates

$$\overrightarrow{Ol} = (Ol_1, Ol_2, Ol_3)$$

and the elements of an orthonormal basis of the local system,

$$\overrightarrow{Lx} = (Lx_1, Lx_2, Lx_3)$$

$$\overrightarrow{Ly} = (Ly_1, Ly_2, Ly_3)$$

$$\overrightarrow{Lz} = (Lz_1, Lz_2, Lz_3)$$

both referenced to the global system.

In this framework if $\vec{R} = (R_x, R_y, R_z)$ and $\vec{r} = (r_x, r_y, r_z)$ are the position vectors of the same point in the global system and local system respectively, there is the following relations between them:

$$\vec{R} = \vec{r} \begin{pmatrix} Lx_1 & Lx_2 & Lx_3 \\ Ly_1 & Ly_2 & Ly_3 \\ Lz_1 & Lz_2 & Lz_3 \end{pmatrix} + \overrightarrow{Ol}$$

and in the other way around:

$$\vec{r} = \left(\vec{R} - \overrightarrow{Ol}\right) \begin{pmatrix} Lx_1 & Ly_1 & Lz_1 \\ Lx_2 & Ly_2 & Lz_2 \\ Lx_2 & Ly_3 & Lz_3 \end{pmatrix}$$

C. Direct geometry

In this section we show how to calculate the three foot coordinates (x_F, y_F, z_F) of any leg of Ptinto when the three A_1, A_2 and A_3 angles are known. To accomplish this aim we need to take into account the sing criterion for A_1 that makes a sing difference between left and right legs.

This calculation is made in the local reference system were we know the three coordinates $(x_{V_1}, y_{V_1}, z_{V_1})$ of the corresponding point of the body and the lengths $R = \left\|\overrightarrow{A1\ K}\right\|$ and $T = \left\|\overrightarrow{K\ F}\right\|$.

Then the coordinates of the knee (x_K, y_K, z_K) for the left legs are:

$$x_K = x_{V_1} + R * \cos\left(\frac{\pi}{2} - \mathrm{A}_3\right) * \sin(\mathrm{A}_1)$$
$$y_K = y_{V_1} + R * \cos\left(\frac{\pi}{2} - \mathrm{A}_3\right) * \cos(\mathrm{A}_1)$$
$$z_K = z_{V_1} + R * \mathrm{sen}\left(\frac{\pi}{2} - \mathrm{A}_3\right)$$

and for the right legs have a difference in:

$$y_K = y_{V_1} - R * \cos\left(\frac{\pi}{2} - \mathrm{A}_3\right) * \cos(\mathrm{A}_1)$$

Then, the foot coordinates are:

for the left legs:

$$x_F = x_K + T * \cos(\pi + (\frac{\pi}{2} - \mathrm{A}_3) + \mathrm{A}_2) * \mathrm{sen}(\mathrm{A}_1)$$
$$y_F = y_K + T * \cos\left(\pi + (\frac{\pi}{2} - \mathrm{A}_3) + \mathrm{A}_2\right) * \cos(\mathrm{A}_1)$$
$$z_F = z_K + T * \mathrm{sen}\left(\pi + (\frac{\pi}{2} - \mathrm{A}_3) + \mathrm{A}_2\right)$$

and for the right ones:

$$y_F = y_K - T * \cos\left(\pi + (\frac{\pi}{2} - \mathrm{A}_3) + \mathrm{A}_2\right) * \cos(\mathrm{A}_1)$$

D. Inverse geometry

In this section we are going to show how to obtain the three A_1, A_2 and A_3 angles, when we know the three foot coordinates:

$$(x_F, y_F, z_F)$$

In this case, called reverse geometry, the three x_F, y_F and z_F foot coordinates are the three degrees of freedom of the leg. These coordinates uniquely determine the values that the three angles should take to get the corresponding foot position.

In the real robot, this foot position will always be feasible. However, in simulated cases, the values of the foot coordinates can be any and even meaningless. In this article we assume that values x_F, y_F and z_F always correspond to a reachable foot position.

A fundamental idea in this calculation is that V_1, V_2, K and F points, which define a leg, are always in a plane surface called "the leg plane."

1) A_1 computation: This angle is simply the angle between the leg plane and the OY axis of the local reference system. It is necessary to obtain A_1 to take into account the criteria we have chosen to select the angle sign. That is, A_1 is considered positive if the leg is advanced in the direction of the OX local system axis. For 1, 2 and 3 legs, on the left side of the robot, the A_1 angle is obtained as:

$$A_1 = \frac{\pi}{2} - \mathbf{atan2}\left(\frac{y_F - y_{V_1}}{x_F - x_{V_1}}\right)$$

where **atan2** is the MATLAB function that returns an angle inside the $[-\pi, \pi]$ interval. In contrast, in the case of the right legs, 3, 4 and 5, the expression for the A_1 angle is:

$$A_1 = \frac{\pi}{2} + \mathbf{atan2}\left(\frac{y_F - y_{V_1}}{x_F - x_{V_1}}\right)$$

2) A_2 calculation: To compute A_2 angle we first obtain the distance D between the foot and its corresponding V_1 vertex.

$$D = \sqrt{(x_F - x_{V_1})^2 + (y_F - y_{V_1})^2 + (z_F - z_{V_1})^2}$$

Thus the A_2 angle is computed as:

$$A_2 = \mathbf{acos}\left(\frac{R^2 + T^2 - D^2}{2\mathrm{RT}}\right)$$

where **acos** is the MATLAB function that returns the angle, in the interval $[0, \pi]$, whose cosine is the value of its argument.

3) A_2 calculation: The A_3 angle calculation can be accomplished in several ways. In this article we have chosen to compute the cutting points of the two following circumferences:

- C_1 centered at V_1 with radius R.
- C_2 centered at F with radius T.

According to the geometry of the legs, these two circles are contained in the "leg plane" and cut at only two points. One of these two points corresponds with the K point of the leg.

Moreover, at this level we assume a two-dimensional coordinate system for the "leg plane". This particular system is centered at the V_1 point, the OX_{lp} axis parallel to the OY axis of the local system and directed toward the outside of the body of the robot and the OY_{lp} axis parallel to the OZ axis of the same local system.

In this new two-dimensional reference system associated with the leg the V_1 vertex has the coordinates $(0, 0)$ and the foot position is represented by the $\overrightarrow{cp}$ vector, where:

$$\overrightarrow{cp} = (cp_1, cp_2) = \left(\sqrt{(x_F - x_{V_1})^2 + (y_F - y_{V_1})^2}, z_F - z_{V_1}\right)$$

Then we calculate the common points of C_1 and C_2 circumferences as the solution of the following algebraic equation system:

$$\left.\begin{array}{c} x^2 + y^2 = R^2 \\ (x - cp_1)^2 + (y - cp_2)^2 = T^2 \end{array}\right\}$$

These system solutions are obtained using the symbolic computation facilities of MATLAB. Besides, we know by the

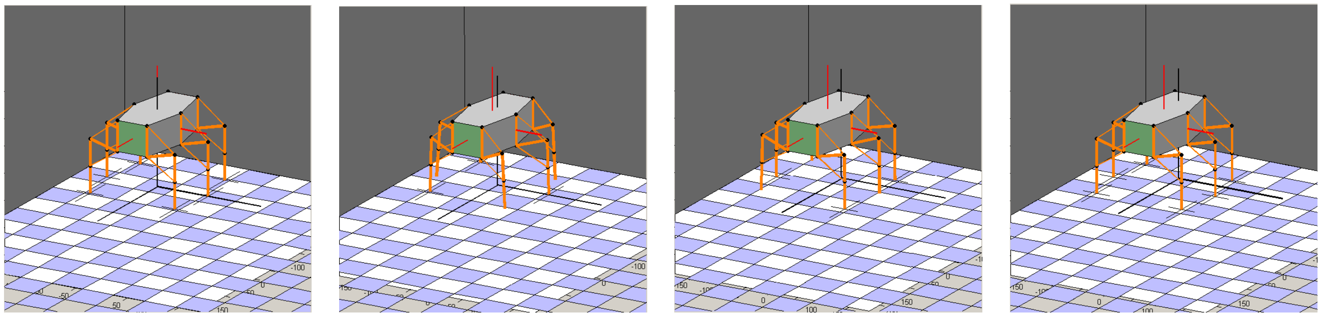

(a) Initial position
(b) Pair legs
(c) Odd legs
(d) Final position

Fig. 4. One step on a horizontal ground

geometric conditions of the problem that there are only two different solutions (x_1, y_1) and (x_2, y_2).

Then, for both we calculate:

$$A_i = \frac{\pi}{2} - \mathbf{atan2}\left(\frac{y_i}{x_i}\right), \qquad i = 1,2$$

and we determine the value of the A_3 angle as:

$$A_3 = min\left\{A_i, \quad i = 1,2\right\}$$

E. Ptinto movements

This section presents the algorithms required to simulate the motion of the robot and represent it by the simulation program.

In general, the method is to establish what should be the final coordinates of the robot elements from a known position and after a particular movement. The final location is determined by the new feet position after moving some of them, step forward or turn a certain angle. In every case, the three A_1, A_2 and A_3 angles, for each leg, are recalculated from the new feet position. In addition, we determine the coordinates of the whole robot points in the local reference system and also set the robot location in the global system after it moves.

1) Movements without changing position: These movements correspond to the movement of one or several feet and that do not involve the displacement of the robot. They include any change of any foot towards a new admissible position.

The movement of any foot can be symbolized by a vector $\vec{m} = (m_x, m_y, m_z)$ in the local system.

Then, from an initial foot location $(x_{F_i},\ y_{F_i}, z_{F_i})$ the final coordinates are obtained as:

$$(x_{F_f},\ y_{F_f}, z_{F_f}) = (x_{F_i},\ y_{F_i}, z_{F_i}) + (m_x, m_y, m_z)$$

Neither the vector $\overrightarrow{Ol}$ nor the local basis $\left\{\overrightarrow{Lx}, \overrightarrow{Ly}, \overrightarrow{Lz}\right\}$ are changed for this movement.

2) Walking on a horizontal ground: This movement involves a change in the vector $\overrightarrow{Ol}$ without changing the local basis $\left\{\overrightarrow{Lx}, \overrightarrow{Ly}, \overrightarrow{Lz}\right\}$. We suppose that Ptinto has three alternate legs (1, 3 and 5 or 2, 4 and 6) resting on the ground and the others ones up. Figure 4 shows this movement.

Then if we symbolize the movement of the robot in the by a vector: $\overrightarrow{d} = (d, 0, 0)$, the new position $\overrightarrow{Ol_f}$ of the center or the robot can be obtained by the previous one $\overrightarrow{Ol_i}$ as:

$$\overrightarrow{Ol_f} = \overrightarrow{Ol_f} + \overrightarrow{d}$$

At the same time, feet on the ground have to be placed backward in order to keep their previous position

$$(x_{F_f},\ y_{F_f}, z_{F_f}) = (x_{F_i},\ y_{F_i}, z_{F_i}) - (d, 0, 0)$$

and the lifted ones moved forward for stepping:

$$(x_{F_f},\ y_{F_f}, z_{F_f}) = (x_{F_i},\ y_{F_i}, z_{F_i}) + (d, 0, 0)$$

When all this changes are made the three A_1, A_2 and A_3 angles, for each leg, have to be recalculated and Ptinto can be displayed at its current location and with its new position.

3) Rotating on a horizontal ground: This movement involves a change in the local basis $\left\{\overrightarrow{Lx}, \overrightarrow{Ly}, \overrightarrow{Lz}\right\}$ without changing the $\overrightarrow{Ol}$ vector. We assume the robot turns α degrees and revolves about the $\overrightarrow{Lz}$ vector. We also suppose that Ptinto has only three alternate legs touching the floor. Figure 5 shows a $90°$ rotation.

From the point of view of the robot, this rotation represents a change in the angles of the leaned legs in such a way that, after making the turn, their feet are in the same point than they were before rotating. This implies, similarly to the case of marching, a rotation movement of Ptinto body and the opposite turn of the feet of the legs touching the ground. In the same way that for walking, the lifted legs make a gyration in the same sense than the robot does.

To make these turns we have chosen the Rodrigues-Hamilton [16] quaternions method. This strategy is simple and can be applied for all rotations occurring during our simulation. In some special cases it may need a higher number of operations that other more direct methods, however, its generality is a very important advantage that compensates for these cases.

The rotation around the vector $\overrightarrow{Lz}$ about an angle α is represented by the quaternion:

$$Q = \left(\cos\left({}^{\alpha}\!/_{2}\right),\ \ \sin\left({}^{\alpha}\!/_{2}\right)\ \overrightarrow{Lz}\right)$$

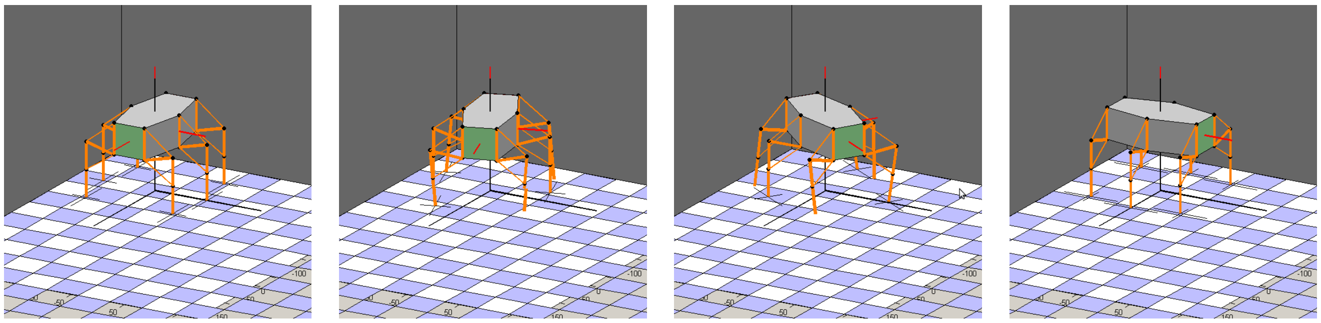

(a) Initial position (b) Rotating (c) Rotating (d) Final position

Fig. 5. Rotating on a horizontal ground

From this quaternion, the $\overrightarrow{Lx_f}$ and $\overrightarrow{Lx_f}$vectors, obtained as result of rotating the $\overrightarrow{Lx_i}$ and $\overrightarrow{Ly_i}$ vectors, can be computed as:

$$\left(0,\overrightarrow{Lx_f}\right) = Q\circ\ \left(0,\overrightarrow{Lx_i}\right)\circ Q^*;\ \ (0,\overrightarrow{Ly_f}) = Q\circ(0,\ \overrightarrow{Ly_i})\circ Q^*$$

where Q^* is the quaternion conjugate and $\circ$ the quaternion product.

Then the feet touching the floor have to be rotated around the same vector $\overrightarrow{Lz}$ and about the angle $-\alpha$. This is done by the quaternion:

$$Q = \left(\cos\left(\frac{\alpha}{2}\right),\ \ \sin\left(\frac{\alpha}{2}\right)\ \overrightarrow{Lz}\right)$$

In that case, the leaned feet coordinates after rotation are $(x_{F_f}, y_{F_f}, z_{F_f})$ and the ones before it are $(x_{F_i}, y_{F_i}, z_{F_i})$, they are related by:

$$\left(0, x_{F_f}, y_{F_f}, z_{F_f}\right) = \left(Q\ ^-\right)\circ(0, x_{F_i},\ y_{F_i}, z_{F_i})\circ\ \left(Q\ ^-\right)^*$$

and, in the same way, for the lifted feet rotation about an angle α,the relation is:

$$\left(0, x_{F_f}, y_{F_f}, z_{F_f}\right) = Q\ \circ(0, x_{F_i},\ y_{F_i}, z_{F_i})\circ\ Q\ ^*$$

After all this transformation we recalculate the A_1, A_2 and A_3 angles and draw the robot on the simulator screen.

V. Conclusion

This work presents a solution to the locomotion problem of a hexapod exploration robot, in particular of the Ptinto prototype. For this we present a cognitive architecture with basic learning possibilities and reactive behaviours models, as well as a suite to simulate the complex kinematics of a six-legged robot with three degree of freedom per leg.

An hexapod robot is a challenge both technically and software. In order to design and test complex kinematics models, we implement the particular geometry of Ptinto into a simulator suite, that allows us to check the behavior of the robot movements. Also this simulator let us to prove the cognitive architecture. This architecture, despite working with few sensors capabilities, is able to handle the robot and update the terrain information on the go, so that Ptinto is able to carry out its mission, which is make exploration missions.

VI. Future work

The basis of the architecture is finished and fully functional, but some behaviours are too simple to control Ptinto in complex areas. In this fact, there could be interesting implement a predictive model for anticipating the stability in the next step, using the data from the knowledge base and the information provided by the position of each leg through the kinematics model of the robot. With this and with more movement models, the robot might be able to select the correct motion task for each terrain in order to exhibit a better locomotion. Also, introduce more sensors will let us expand the learning possibilities.

Related to the simulator, the next thing that has to be done is to elaborate movement algorithms for more complicated terrains. This means not flat grounds and even softer ones. In the first case, when the floor becomes irregular it will be necessary to use more complicated gaits that the tripod ones considered in this paper. On top of that, more elaborated vector transformation should be taken into account and when footprints are not on a plane, some equilibrium aspect would appear.

Acknowledgment

This work is funded by the Castilla-La Mancha project PEII09-0266-6640.

References

[1] D. Vernon, G. Metta and G. Sandini. *A Survey of Artificial Cognitive Systems: Implications for the Autonomous Development of Mental Capabilities in Computational Agents*, IEEE Transactions on Evolutionary Computation, vol.11(2), pp.151-180, April 2007.

[2] P. Langley, J. E. Laird and S. Rogers. *Cognitive Architectures: Research Issues and Challenges*. Cognitive Systems Research, vol.10(2), pp.141-160, 2009.

[3] J. E. Laird, A. Newell, and P. S. Rosenbloom. *Soar: An architecture for General Intelligence*. Artificial Intelligence, vol.33(3), pp.1-64, 1987.

[4] John E. Laird. *Extending the Soar Cognitive Architecture*. In Proceeding of the 2008 Conference on Artificial General Intelligence 2008, 224-235, 2008.

[5] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin. *An Integrated Theory of the Mind*. Psychological Review, vol.111(4), pp.1036-1060, 2004.

[6] P. Langley, K. Cummings and D. Shapiro. *Hierarchical Skills and Cognitive Architectures*. In Proceedings of the Twenty-sixth Annual Conference of the Cognitive Science Society (pp. 779-784). Chicago, IL, 2004.

[7] P. Langley and D. A. Choi. *Unified Cognitive Architecture for Physical Agents*. In Proceedings of the Twenty-first AAAI Conference on Atificial Intelligence. Boston: AAAI Press, 2006.

[8] R. Bonasso, R. Firby, E. Gat, D. Kortenkamp, D. Miller and M. Slack. *Experiences with an Architecture for Intelligent, Reactive Agents*. Journal of Experimental & Theoretical Artificial Intelligence, vol.9(2-3), pp.237-256, 1997.

[9] D. Kraft, E. Baseski, M. Popović, A. Batog, A. Kiær-Nielsen, N. Krüger, R. P.A. Petrick, C. Geib, N. Pugeault, M. Steedman, T. Asfour, R. Dillmann, S. Kalkan, F. Wörgötter, B. Hommel, R. Detry and J. Piater, *Exploration and Planning in a Three-level Cognitive Architecture*. In International Conference on Cognitive Systems (Workshop at the IEEE International Conference on Robotics and Automation), 2008.

[10] R.A. Brooks. *A Robust Layered Control System for a Mobile Robot*. IEEE Journal of Robobotics and Automation, vol.2, pp.14-23, March 1986.

[11] R. A. Brooks. *Intelligence without Representation*. Artificial Intelligence, vol.47, pp.139-159, 1991.

[12] P. Muñoz, M. D. R-Moreno and B. Castaño. *Integrating a PDDL-based planner and a PLEXIL-executor into the Ptinto Robot*. In Proceedings of the 23rd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems: Next-Generation Applied Intelligence (IEA-AIE 2010). Lecture Notes In Artificial Intelligence, pp.72-81 Córdoba, Spain, June 2010.

[13] P. Marchand and T. Holland, *Graphics and GUIs in MATLAB*. Chapman and Hall/CRC, 2002.

[14] The MathWorks, Inc. MATLAB 7: *Creating Graphical User Interfaces*. http://www.mathworks.com/help/pdfdoc/matlab/buildgui.pdf.

[15] V. M. Budanov, *Algorithms of motion planning for a six legged walking machine*. Journal of Mathematical Sciences, Vol 146, No 3, 2007.

[16] W.R. Hamilton. *Elements of Quaternions*. volume I, Chelsea Publishing Company, Thrid edition, 1969.