A framework for robotic rovers' navigation and obstacle avoidance

P. Coraggio[1], A. De Santis[2] and B. Siciliano[2]

Università di Napoli Federico II

[1]Dipartimento di Scienze Fisiche, Via Cinthia, 80136 Napoli, Italy, pcoraggio@na.infn.it
[2] Dipartimento di Informatica e Sistemistica, Via Claudio 21, 80125 Napoli, Italy, {agodesa, siciliano}@unina.it

Abstract

Keywords: reactive/deliberative architecture, mobile holonomic robot, collision anticipation.
In this paper, a deliberative/reactive robotic architecture for spatial exploration in an unstructured, unknown and hostile environment is proposed. In order to better operate in this environment, the robot government system is formed by two independent modules working in a parallel, asynchronous way: a deliberative one and a reactive one. The main purpose of the reactive module is to preserve the robot structure while the deliberative module is responsible of a high level inferential system enabling the robot to perform flexible actions

1. INTRODUCTION

Navigation in unstructured, unknown and potentially hostile environment is still a crucial topic in autonomous robotics. In this case, the robot has no map of its environment and no or less possibility of building it from its sensory data. Moreover, the computational efforts in building maps could result unessential for its task. Classical methods [1], [2], [3] provide sensors-to-action modules in order to guarantee both the robot safety and the performing of task directed action. These modules have to be combined (e.g. in a cooperative or concurrent way), but once fixed this combination, the system can show none or less flexibility in changing this order.

In hostile environments, this flexibility could constitute a necessary feature for the robot to be operative. In order to endow the robot with this peculiarity, a hybrid reactive/deliberative architecture has been designed. In addition to a reactive module devoted to the robot survival with strict real time constraint, a deliberative one has been created. The latter has a twofold activity: based on the sensory data, this module updates some current parameters in order to better manage its current state (a sort of learning mechanism); then, it tries to state some environmental characteristics. In this way, the whole system can improve its performance: through a predictive mechanism, the system can speed up its computation reducing the amount of data processed, focusing on particular states achieved.

2. SYSTEM OVERVIEW

The robot has a mobile holonomic base and a manipulator to pick objects mounted on it, and it is equipped with a camera able to monitoring the environment. Moreover, the robot has a shield enabling it to defend itself from small objects falling on its path. The robot government system is formed by two independent modules working in a parallel, asynchronous way: a deliberative one and a reactive one. The main purpose of the reactive module is to preserve the robot structure, by exploiting data coming from a camera and other sensors (with parameters supplied by the deliberative module). This module focuses its attention to the nearby environment, with special attention to the possible collision with falling objects. The inputs of this module are the sensory data, and some

parameters are computed by the deliberative module. These parameters can be shaped, as we will show later, as a first order logic predicate that can be processed by an inferential engine.

The deliberative module is devoted to long-term computation in the sense that its main tasks are: planning the best trajectory, estimate the impact energy, update some control parameters in order to improve the reactive module performance. The latter is done by making multiple hypotheses on the environment and the state of the robot and testing them with the perceptual data. The most important evaluation is the impact energy of falling objects for an anticipated detection of potentially dangerous collision.

As a first step, the robot plans its best trajectory for a specific assigned task (deliberation); the nearby environment is watched (with special attention to the possible collision with falling objects). By computing the time interval between an object's detection and its estimated impact with the robot, the system is able to decide among different solutions (e.g., satisfy avoidance in an inverse kinematics scheme, see below for details). A sketch summarizing the whole system is presented in fig. 1.

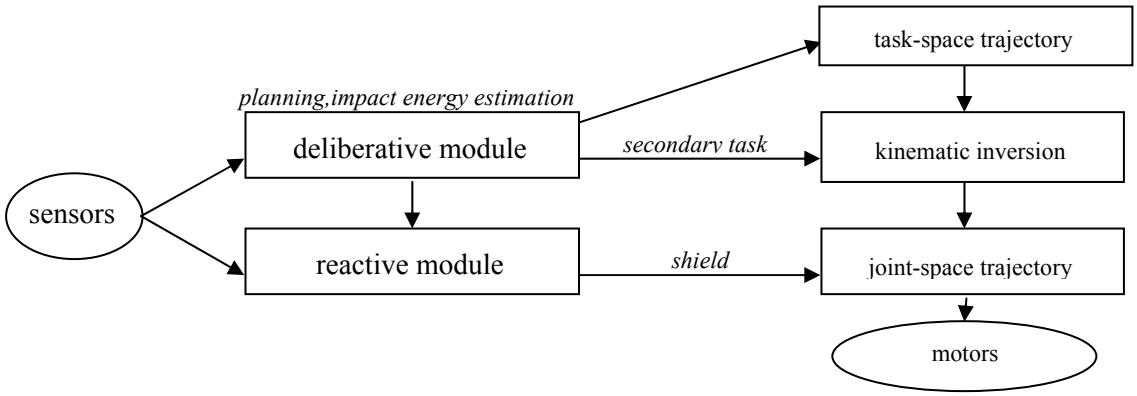

Figure 1 System overview

2.1 THE REACTIVE MODULE

The reactive module's inputs are the sensory data and the information on activate behaviours coming from the deliberative module. It shares an inferential engine with the deliberative module in order to gain advantage from some inferential chain taking place in both modules.

Based on the deliberative module predictions, the reactive module manages the following robot's behaviours in the presence of a potentially dangerous objects colliding with the robot:

1. Don't care: if the impact energy computed is under a certain threshold; it means that the robot has to go on its planned path.
2. Protect yourself with the shield and go on, if the impact energy has a range between a minimal and a low threshold; it means that the robot has to turn and protect itself with the shield. If possible, it has to stop in order to guarantee an optimal equilibrium.
3. Escape, if the impact energy is estimated great enough to potentially damage the robot structure. It means that the robot has to run away from its actual position with the maximum velocity, being careful to avoid other obstacles on the ground.

After each impact, the robot reactive module matches the energy impact estimated with the one computed from sensory data. This information is sent to the deliberative module, enabling it to construct a set of cognitive evaluations about the falling objects and, if necessary, to update its parameters' inference system.

2.2 THE DELIBERATIVE MODULE

As a first step, the robot plans its best trajectory for a specific assigned task (deliberation). A kind of anticipated perception system in the deliberative module is able to compute a not exact evaluation of the impact energy of an object potentially hitting the robot, where the computation is based on the form, colour and velocity of moving objects detected by the sensors, using updatable parameters for the inferences. Notice that these features may be changed depending on the usual falling objects of the environment. By computing the time interval between an object's detection and its estimated impact with the robot, the system is able to decide among the following options

- Satisfy this obstacle's avoidance as a secondary task in its inverse kinematics scheme with functional redundancy, if enough time is available to execute its primary task (e.g. to pick up some sample from an unknown planet).
- Compute a new trajectory in the task space (the deliberative module plans a new optimal path) in order to avoid the impact and to decide dropping its primary task, if it is no longer compatible with the eventual impact of dangerous objects.
- If there is not enough time to plan a new trajectory, activate the reactive module (with parameters established by the deliberative module) taking into account the possibility of impact with other obstacles on the ground.

2.3 THE INFERENTIAL ENGINE

The robot control system is equipped with an inferential engine enabling the robot with different capabilities. This inferential engine is present in both modules, the reactive one and the deliberative one. It is based on symbolic processing being able to treat a data set of robot belief. Briefly, we give here some properties. It is based on two main characteristics: plans and beliefs, that can be regarded as a simplified Procedural Reasoning System [4].

Beliefs represent what the system states about its environment and the convictions established during its operations. They are constituted by first order logic ground predicates. The belief set dynamically changes updating itself by asserting or retracting new believes. This set can thus represent a simply, flexible and modifiable structure on which the system can get information in order to decide the best action to perform. Moreover, the inferential engine can operate on it adding new beliefs that represent hypotheses stated by the system and that can be validated in a future state of the robot. In this sense, it can be regarded as a learning mechanism.

Plans represent the system capability of operating on some information coming either from the sensors or from an inferential chain actually processing. They have the following form [5]:

$$+!e : b_1 /\backslash b_2 /\backslash \ldots /\backslash b_n \leftarrow h_1;h_2;\ldots;h_m$$

where: $+!e$ is the *triggering event* describing when a plan can be activated (the triggering event can be either a goal or an environment change); $b_1 /\backslash b_2 /\backslash \ldots /\backslash b_n$ is a beliefs conjunction that is the precondition to be held in order to execute a plan; $h_1;h_2;\ldots;h_m$ is the *body* of the plan: a sequence of actions to execute or subgoals to achieve that the agent has to perform once the plan has been selected.

The inferential engine works in this way. Some sensorial readings are events triggering an appropriate plan: if the preconditions of that plan are actually holding, the system performs the body of the plan. A single event can activate different plans, but the system performs the one that the preconditions match with the beliefs actually true. The body of a plan can be constituted by commands to the actuators, assertion or retracting of actual beliefs, other triggering events called "inner events" activating other sub-plans. In this way, the robot control is able to exhibit a flexible behaviour depending from different situations (discriminated by different preconditions) and a learning capability embodied in the belief set.

In the reactive module, the plan body is constituted by actions and/or sub plans calls that have to be processed only by the deliberative module. This is due to the strict time constraint of the system. The deliberative module, instead, can process plans more elaborate with the capability of asserting or retracting beliefs.

Example of system's base belief are the following:

- `safe_energy(k);`
- `computed_energy(T);`
- `location_sample(X,Y).`

The first states that the system "believes" that the value `k` is a safe energy for the robot structure, the second one means that the system has computed `T` as impact energy, the latter is a (rough) evaluation of the location of a rock sample.

Examples of "reactive plans" are the following:

- `+!escape() : computed_energy(X) > (safe_energy + tollerance) /\ free_pathway(west) <- !activate_run_away(west);`
- `+!protect() : (safe_energy(k) – tollerance) < computed_energy(X) < (safe_energy(k) + tollerance) <- stop_motors; turn(180); activate_evaluetor().`

The first plan is activated when the computed energy is greater than the safe energy and there is a free path in west direction. In this case, the robot has to escape in that direction. The second one, is enabled to protect the robot by disabling the motors and turns the shield if the impact energy included in a range of safe energy. Moreover, the reactive module activates an *evaluator* (a plan in the deliberative module) in order to update the safe energy parameter. Such a plan will have the following form:

- `+!activate_evaluator() : safe_energy(k) < impact_energy(k') /\ (not demaged(robot)) <- \safe_energy(k);/ safe_energy(k'); plan_new_trajectory()`

In this case, if the measured impact energy is greater then that previously computed (and the robot has not relevant damages), then this parameter is updated by retracting the old value of safe energy `k` and asserting the new one `k'`.

3. SIMULATION CASE STUDY

3.1 KINEMATICS MODELLING

The kinematic model of the robot proposed for simulations is one of the feasible combinations of a mobile base with a manipulator. We chose a holonomic platform (to perform future experiments with available robots) and a two-link planar robot arm. It is possible to use also a non-holonomic mobile base and a more complicated manipulator: from an inverse kinematics point of view, the proposed architecture is focused on the possibility of activating or not an obstacle avoidance task (using manipulators' techniques [6]) on the basis of inferences about the environment, based on the estimated potential danger of an impact. So, we considered a 2-wheel mobile base whose differential kinematics is described by a Jacobian matrix *J(q)* which maps the correspondence between the velocity of the end-effector $\dot{p}$ and $\dot{q}$, i.e., the vector including the velocity of the wheels and of the joints of the manipulator. Using manipulators' techniques, the inverse kinematics may be computed in a closed-loop inverse kinematics (CLIK) scheme. To solve the inverse kinematics problem, the vector *q* of the joint variables must be computed starting from the position vector *p.* An effective way to compute the inverse kinematics is that of resorting to the differential kinematics equation

$$\dot{p} = J(q)\dot{q} \qquad (1)$$

From a kinematic point of view, the robot may be considered redundant, if we are interested only to the positioning of the end effector; however, extra joints and wheels can be added to increment the degrees of mobility. The redundancy of the system is very important: we will see that it is possible to add secondary task which do not interfere with the primary one. For example, it is possible to follow a trajectory reconfiguring the manipulator to avoid obstacles or to keep a dexterous position. The differential mapping presented may be inverted using the pseudo-inverse (because the matrix is non-square) of the Jacobian matrix, i.e.,

$$\dot{q} = J^{+}(q)\dot{p}$$

where J^{+} corresponds to the minimization of the joint velocities in a least-squares sense. The redundancy of the system can be further exploited by using a task priority strategy corresponding to a solution of the form

$$\dot{q} = J^{+}(q)\dot{p} + (I_N - J^{+}(q)J(q))\dot{q}_a$$

where $I_{\mathbf{N}}$ is the (NxN) identity matrix (with N equal to the number of degrees of mobility available), $\dot{q}_a$ is an arbitrary joint velocity vector and the operator $(I_{\mathrm{N}} - J^{+}(q)\,J(q))$ projects a joint velocity vector in the null space of the Jacobian matrix. This solution generates an internal motion of the robotic system (secondary task) which does not affect the motion of the point p (primary task). The joint velocity vector $\dot{q}_a$ can be chosen to be aligned with the gradient of a scalar objective function to perform a secondary task. So we have

$$\dot{q}_a = -\,\alpha \frac{\partial G(q)}{\partial t} \qquad (2)$$

To avoid numerical drift due to discrete-time integration, a closed-loop inverse kinematics (CLIK) algorithm can be adopted, which computes q by integrating the vector:

$$\dot{q}_a = J^{+}(q)v + (I_{\mathrm{N}} - J^{+}(q)\,J(q))\dot{q}_a$$

with $v = p_{\mathbf{d}} + K(p_{\mathbf{d}} - p)$, where K is a (3 x 3) positive definite matrix gain to be chosen so as to ensure convergence to zero of the error $p_{\mathbf{d}} - p$. Notice that the subscript d in the expression of v denotes the components of the position and velocity vectors that are input to the CLIK algorithm (i.e., the desired trajectory, which can vary if the avoidance as a secondary task is not feasible, see section 2); the position components without subscript d are those computed from the joint position vector q (the output of the algorithm) via the direct kinematics equation, which can be obtained by integrating (1). The first two behaviours in the deliberative module can be implemented by varying the parameter α in equation (2). The secondary task of obstacle avoidance is activated on the basis of the value of that parameter. If inferences show that avoidance is impossible, a new trajectory will be planned. Also soft computing techniques can be used for the inference about α. So, in this example, the change of parameters in the CLIK scheme is used to implement the deliberative behaviours. Also potential field techniques can be used for the secondary task of obstacle avoidance.

3.2 SENSORS SUITE

The sensors needed for the estimation of the impact energy are mainly cameras combining information on colour, shape and dimension of the falling objects. A training phase is needed to select the features for the classification of the environment. Force sensors should be used in addition, for detection of the real energy at the impact.

3.3 SIMULATION

An example of application of the proposed robot is the exploration of an environment where obstacles are detected. The first obstacle is classified as avoidable and then avoided as a secondary task in the CLIK scheme; the second obstacle is falling very fast, but it is considered not dangerous and so the shield is used; the third obstacle is big and dangerous, so the robot runs away. The planned robot trajectory is shown in fig. 2.

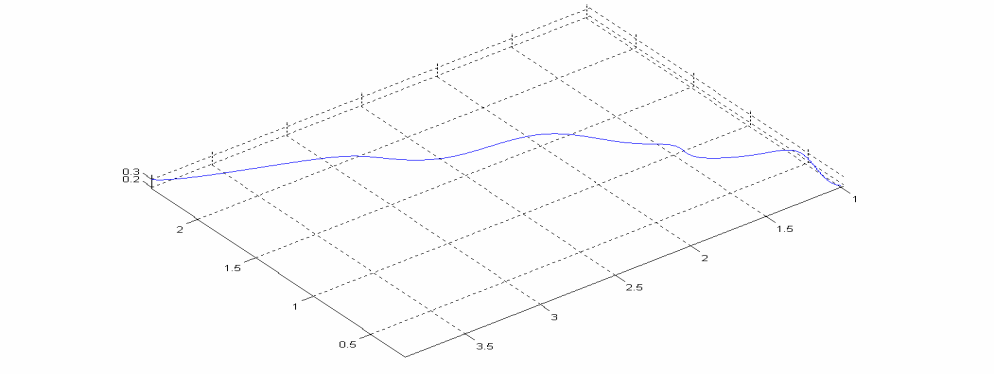

Figure 2 Planned robot trajectory

The obstacles are detected and the position of the estimated impact point is computed; in fig. 3 the complete resulting trajectory tracked is reported.

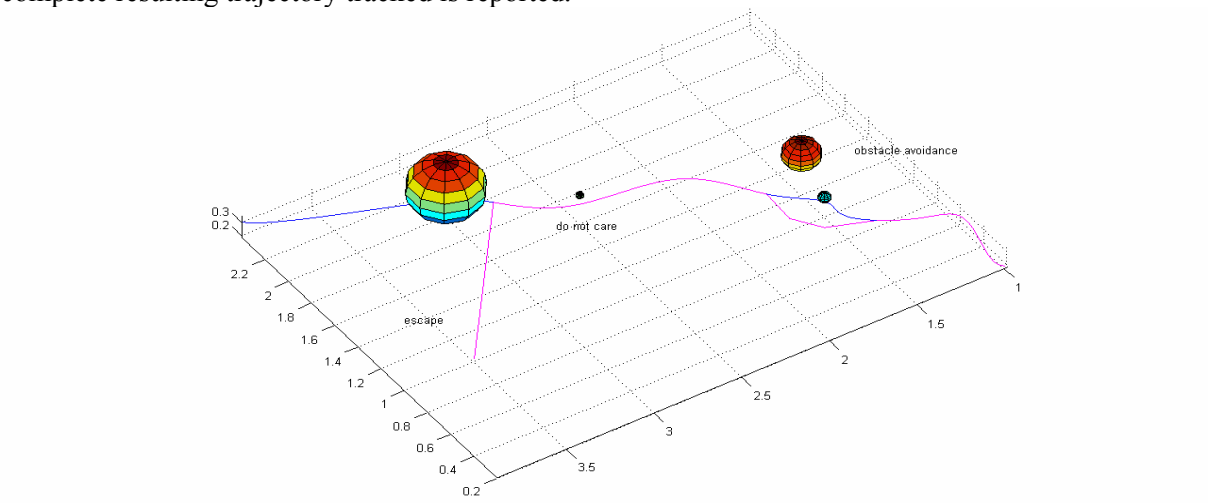

Figure 3 Resulting trajectory

4. FUTURE WORK

The proposed framework is organized in a very modular fashion: we plan to implement the modules (anticipated collision perception, kinematics, inferential engine) with different approaches. For planetary rovers, the features of potentially hitting obstacles and the possibility of avoiding digs has to be considered.

REFERENCES

[1] R. Brooks, "New approaches to robotics", *Science*, vol. 253, pp. 1227-1232, 1991.
[2] L. Steels, "When are robots intelligent autonomous Agents?", *Robotics and Autonomous Systems*, vol.15, pp. 3-9, 1995.
[3] R. Murphy, *Introduction to AI Robotics*, MIT Press, 2000.
[4] A. R. Rao, M. P. Georgeff, "BDI agents: From theory to practice", Technical Note 56, April 1995.
[5] E. Burattini, P. Coraggio, M. De Gregorio, "Agent wisard: A hybrid system for reconstructing and understanding two-dimensional geometrical figures", in *Design and Application of Hybrid Intelligent Systems*, Abraham et al. (Eds.), pp. 887-896, 2003.
[6] B. Siciliano, "Kinematic control of redundant robot manipulators: A Tutorial" *Journal of Intelligent Robotic Systems*, vol. 3, pp. 201-212, 1990.