Experiments on cooperative behaviour between two mobile manipulators*

João Sequeira ** **and M. Isabel Ribeiro** ***

Instituto Superior Técnico / Instituto de Sistemas e Robótica
IST - Torre Norte, Av. Rovisco Pais 1, P 1096 Lisboa CODEX, Portugal

Abstract. This paper describes an experiment on cooperation between two mobile manipulators using a *behaviour*-based approach. The problem considered is a variation of the transportation of a rigid bar between two locations in the workspace with the two mobile manipulators holding the bar while avoiding any physical contact between each other. One grasping point is fixed at a predefined extremity of the bar while the other is allowed to slide along the bar. Both manipulators are assumed to grasp the bar using a compliant connection.
The control methodology proposed in this paper, for each single robot, is supported on the definition of equivalence classes in the space of feasible trajectories. An iterative algorithm "shapes" the *a priori* defined equivalence classes to cope with the diversity of the environment, while executing a mission. The resulting equivalence classes constitute the *behaviours* of the robot. Simulation results are presented.

1 Basic control methodology for single robots

The *behaviour*-based control paradigm considered in this paper for the control of every single robot, follows the information pipeline *Environment* → *Sensors* → *Behaviour generator* → *Actuators* → *Robot* (see Table 1). The *behaviour* generator assumes that a robot can be controlled by switching, at spaced time

S	The sensing information, e.g., battery level, ultrasound data, etc,
Q	The robot configuration space, or C-space, $(q \in Q)$,
A	The set of every possible action, $(a \in A)$,
$A^p \subset A$	The set of the user defined actions,
$T = \{a(q)\}$	The state space.

Table 1.: Relevant sets for the information pipeline.

instants, among *a priori* defined references established using the *a priori* knowledge about the robot and the environment where it operates. These references represent the action of the group $\mathbb{R}$ (the time axis) on Q, the robot C-space, (see [Hungerford, 1996]), thus being named as *actions* in this paper. Hereafter, the word *agent* will be used to designate a single robot system behaving according to the aforementioned pipeline.

The basic building blocks of a *behaviour*-based architecture are supported in the three following concepts (see [Sequeira and Ribeiro, 1997] for details):

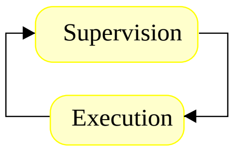
State space T: Any element of the state space is composed by an action, a, and a configuration, q, in the neighbourhood of which the action starts to be applied, to generate a trajectory belonging to an equivalence class of $a(q)$.

Equality relation in T: The equivalence relation defining a condition of similarity between states.

State composition operation: An operation expressing the ability to create motion in T. A trajectory resulting from the composition of any two states is equal, in the sense of the equality relation defined in T, to the trajectory generated by the present state, the one of the future state and, if necessary, a link path between the two of them.

* Work sponsored by the programme PRAXIS XXI, project COOPERA - 2/2.1/TPAR/2087/95.
** e-mail: jseq@isr.ist.utl.pt
*** e-mail: mir@isr.ist.utl.pt

The state space T, defined under the aforementioned principles, has a group structure represented in the execution block of the architecture of Figure 1, [Sequeira and Ribeiro, 1997]. The $\ominus$blocks represent transformations preserving the equality relation in T while $\otimes$ stands for the state composition operation. A supervision block divides each assigned task (which are themselves elements of T) into a sequence of elements of T, each of which acts as a reference to be followed by the execution block. The feedback path returns to the supervision block the necessary information on the current task status.



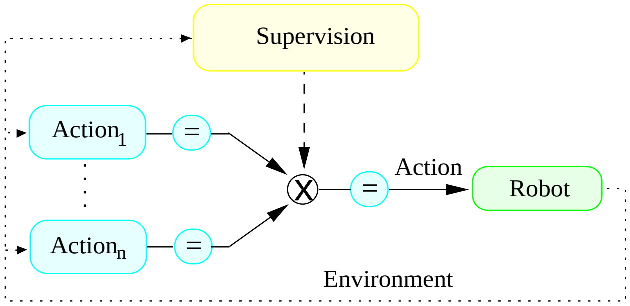


Fig. 1.: A *behaviour* generator architecture.

In general, the trajectory generated by an agent state $a(q)$, depends on the interactions with the environment. At different occasions and due to the environmental uncertainties and modifications, the same $a(q)$ will, in general terms, generate different trajectories that still lead the agent to accomplish its task. Thus, if this set of trajectories is enclosed by some subset of the Q, represented by $B_a(q)$, an equivalence class is established in the sense of the aforementioned equality relation in T. Complex interactions between the agent and the environment may require state bounding sets which are difficult to visualize *a priori*. Therefore, an iterative procedure for the construction of the $B_a(q)$ is defined as follows:

1. Assume that a covering $\{B^i(q^i),\ i = 1, \ldots\}$ for the set $B_a(q)$ can be defined. For instance, if an agent is in state $a(q)$, let $B^1(q^1), B^2(q^2), \ldots$ be disjoint bounded sets such that

$$q^p \in B_a(q)$$
$$\cup_p B_a^p(q^p) = B_a(q).$$

2. Each time the agent configuration goes outside an element $B_a^p(q^p)$ into a region not yet covered, a new element $B_a^{p+1}(q^{p+1})$ is automatically joined to the covering. Whenever a change of state is imposed by the supervisor, the construction of a new covering for the new state starts.

3. For each p, register the action in execution, defining a state $a^p(q^p)$. This step allows future executions of the assigned task using the same trajectory in T, this procedure being a learning type methodology. However, to reinforce this learning and instead of relying exclusively on the supervisor choices for the actions, the performance of the navigation algorithm can be improved by a local search in the actions space. This search is defined by:

3.1. Define an action modifier mapping $M : A \mapsto A$, to introduce slight changes in the actions so that alternative trajectories can be generated and tested.

3.2. Define a metric in $S \times T$ to compute a distance between states.

3.3. Using the metric defined in step 3.2, apply M to each of the k-nearest neighbours of $a^p(q^p)$, for each p, and choose the best.

The resulting trajectory is generated by the composed state $a^{p_i}(q^{p_i}) \circ a^{p_i-1}(q^{p_i-1}) \circ \ldots a^1(q^1) \circ a^0(q^0)$, which is contained inside the covering set $\cup_{l=0}^{p_i} B_l(q^l)$. To the best of the authors knowledge, a version of the algorithm first appeared in [Millan, 1996] under the name of *Incremental Preserving Topology Map* (ITPM), without being included in any particular framework.

2 Case study

Figure 2 shows a 3D view of the simulation environment. The mobile manipulators are similar to the one described in [Sequeira et al., 1997]. No obstacles in the range of the mobile platforms were considered. However, since one of the grasp points is allowed to slide along the bar (3m long), each mobile platform has detection and collision avoidance capabilities. The system has a total of six agents, three for each mobile manipulator: mobile platform, arm (the first three joints of the manipulator) and gripper (the remaining three joints), each running according the architecture of Figure 1. All the supervision blocks were implemented using Finite State Machines. The information exchanged between the two mobile manipulators (MM0 and MM1) is limited to the position of the bar grasp points.

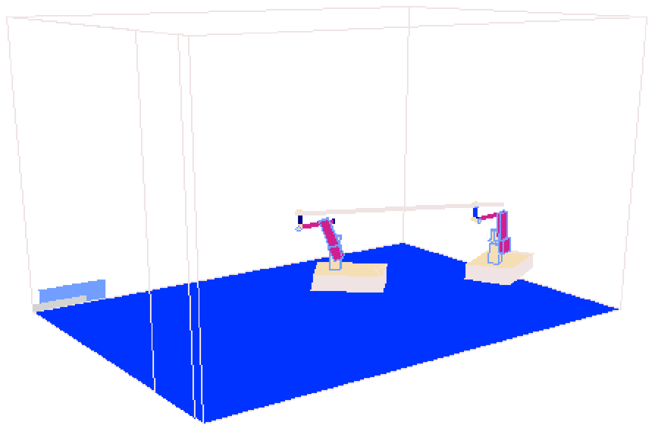


Fig. 2.: A 3D perspective of the simulation environment.

The mobile platform is equipped with a set of basic actions able to: stop, go straight to goal and avoid an obstacle. The arm basic actions define the strategies to: stop, go straight to goal and get out of the workspace regions that are close to the singularities. The gripper has a unique action: go straight to goal.

The mission considered in this paper is composed by four tasks, representing the approach, the positioning and the final phases of a typical transportation of a bar to a shelf in a warehouse. The x, y, z gripper assigned locations to fullfil the tasks are specified in Table 2 (units are meters), with the gripper at the orientation $\alpha = \beta = \gamma = 0$. Whenever the distance between the grasping points exceeds a pre-specified value, MM0 and waits for MM1 to move aiming at decreasing this distance. Note that the distance between the locations of the fourth task assigned to each mobile manipulator is made bigger than the lenght of the bar for the purpose of testing the cooperation between the two devices. The mobile platforms start at configurations $(3.93m, 2.07m, 0rad)$ for MM0 and $(6m, 2.9m, \pi/2rad)$ for MM1, with both manipulators in the upward position.

MM	Start	Task 1	Task 2	Task 3	Task 4
0	$(3.9, 2.1, 0.9)$	$(3, 1, 0.4)$	$(3, 1, 0.5)$	$(2, 1, 0.4)$	$(1, 2, 0.4)$
1	$(6, 2.9, 0.9)$	$(1, 3, 0.5)$	$(1, 3.5, 0.3)$	$(-1, 2, 0.3)$	$(-3, 2, 0.3)$

Table 2.: Mission assignments.

Figure 3 shows the superimposed results of five consecutive runs of the mission. The plots in the x-y plane show the trajectories of the mobile platforms (point lines) and of the grippers (full lines), for both mobile manipulators. The plots in the x-z plane and in the 3D workspace show the trajectories for the two grippers (point line for gripper 1 and full line for gripper 0). The tasks for MM0 and MM1 are marked, respectively, with $*$ and $\circ$. Since the distance between the MM0 starting configuration and its tasks is smaller than the

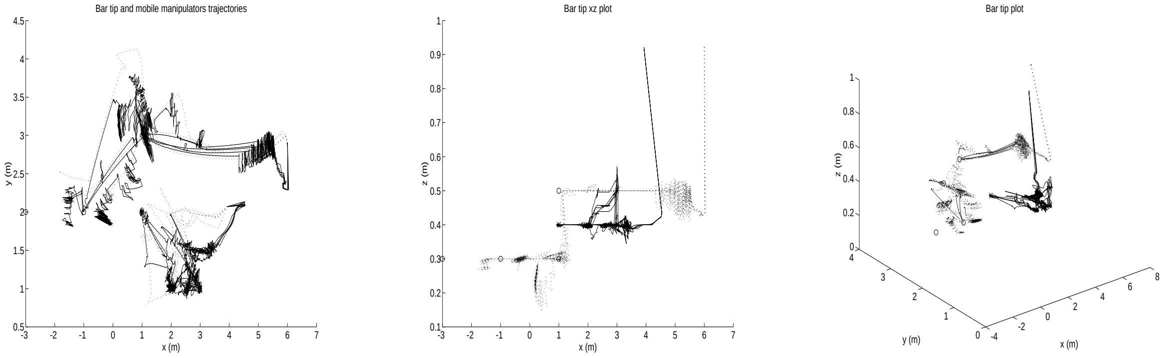


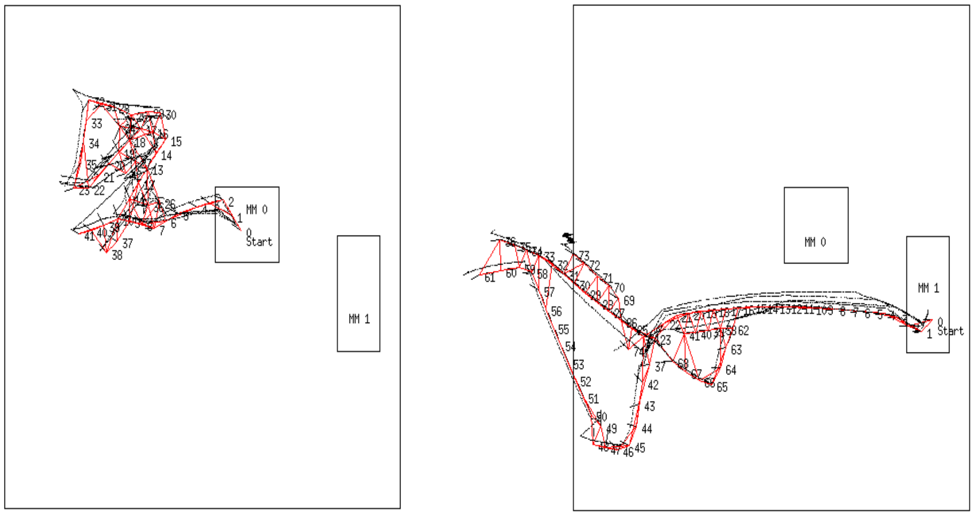
Fig. 3.: Mission plots.

distance for MM1, the system tends primarily to execute the tasks assigned to MM0, with MM1 cooperating to hold the bar.

Figure 3 clearly shows the aggregates of trajectories which represent the *behaviours* obtained for the mobile manipulators. Figure 4 shows the corresponding *behaviours* obtained for the mobile platforms. The mobile platforms are represented, at their starting configurations, by its borders with the world frame origin being located in the upper left corner of the border line shown.

Each of the diagrams in Figure 4 shows the position, in the xy plane, of the x^p, y^p coordinates of the center of the neighbourhoods $B^p(q^p)$ (cylinders in Q, with symmetry axis θ and radius 0.2m) used to bound the states $a^p(q^p)$. The numbers indicate the value of the p index. Superimposed on (x^p, y^p) is a small line segment that indicates the direction for the action taken inside that area. The solid lines connecting the (x^p, y^p) points represent the 2-nearest neighbours considered in this example. The trajectories made by the mobile platforms in the 5 runs of the mission are represented as pointed lines.

In the diagram corresponding to the mobile platform 1 two changes of *behaviour* are visible, which were due to collision avoidance action with the mobile platform 0. These are not so clearly visible in the diagram of mobile platform 0 due to the fact that MM0 is primarily concerned with the execution of its mission, while MM1 only executes its own mission if it is grasping the bar away from the grasp point of MM0.



Mobile platform 0. Mobile platform 1.

Fig. 4.: Typical *behaviours* of the mobile platforms after 5 runs of the mission.

3 Conclusions

This extended abstract presented a *behaviour*-based architecture for robot control. An iterative algorithm adapts the user defined actions to the diversity of environment conditions. This algorithm is supported on the definition of an equivalence relation in the state space used to describe the problem.

The architecture defined for single agents proved adequate in the bar transportation problem, as the mobile manipulators were able to cooperate in the execution of the assigned mission.

The full paper extensivelly describes the framework underlaying the developed cooperation architecture and presents extensive simulation results.

References

[Hungerford, 1996] Hungerford, T. (1996). *Algebra.* Number 73 in Graduate Texts in Mathematics. Springer, 8th edition.

[Millan, 1996] Millan, J. (1996). Incremental Acquisition of Local Networks for the Control of Autonomous Robots. Private communication.

[Sequeira et al., 1997] Sequeira, J., Millan, J., Ribeiro, M. I., and Gonçalves, J. M. G. (1997). Embedding Learning in Behaviour-Based Architectures: A Conceptual Approach. In Monostori, L., editor, *Proceedings of The Second World Congress on Intelligent Manufacturing Processes & Systems.* Springer. Budapest, Hungary.

[Sequeira and Ribeiro, 1997] Sequeira, J. and Ribeiro, M. I. (1997). A Behaviour-Based Kernel Architecture For Robot Control. In *Proceedings of 5th IFAC Symposium on Robot Control, Syroco'97.* Cité Des Congres - Nantes, France.

This article was processed using the TEX macro package with SIRS98 style