A Fuzzy Behavior-Based Control System for Manipulation

Zbigniew Wasik and Alessandro Saffiotti

Center for Applied Autonomous Sensor Systems
Dept. of Technology, Örebro University
S-70182 Örebro, Sweden
zbych@aass.oru.se, asaffio@aass.oru.se

Abstract

Behavior-based approaches to robot control are extremely popular in mobile robotics, but still rarely used for manipulation. We propose a behavior-based system that performs manipulation tasks using visual feedback. The distinctive points of our proposal are: (i) visual behaviors are implemented using a new camera-based approach; (ii) reactive fuzzy rules are used to arbitrate behaviors; and (iii) the outputs of concurrent behaviors are fused using fuzzy logic. We show experiments on a real arm performing a pick-and-place task that illustrate our approach and demonstrate its advantages over current approaches.

1 Introduction

The long term objective of the research reported here is to build a system for mobile manipulation, where the mobile platform works concurrently with an arm mounted on it. Our approach is to use behavior-based control both for the mobile base and for the robot arm, and then integrate them to obtain an overall coordinated motion. This paper presents a first step in this direction: the definition of a behavior-based control system for the manipulator. Since we want our system to operate in unstructured environments, where the position of objects is not known *a-priori*, we consider the use of visual feedback.

A few behavior-based approaches to manipulation have already been proposed in the literature. Several of these systems are based on Brooks' subsumption architecture [1], including Connell's early can retrieving robot [3], as well as systems based on fuzzy logic control [5], on force-based control [18], or controlling artificial muscles in a humanoid arm [19].

Other behavior-based approaches use finite state machines to sequence behaviors. For instance, Pettinaro [13] mostly uses sequential activations of generic arm behaviors to perform assembly tasks. He also uses a few concurrent behaviors, but these usually inhibit each-other. De Giuseppe [6] uses sequential activations of fuzzy behaviors to perform a pick-up task. Pettersson [12] and Nagatani [10] implement mobile manipulation tasks by sequencing of behaviors.

All the above approaches have two points in common. First, they implement vision-based behaviors using either image-based or position-based image servoing (see Section 3 below). Second, they perform a crisp arbitration between behaviors, meaning that only one manipulation behavior is in the control of the manipulator at any given time. A few exceptions where behaviors can be concurrently executed are [8, 2, 9]. [8] uses fuzzy behaviors for mining excavator tasks, and allows several behaviors to concurrently control the arm: arbitration is performed by a ANN, and command fusion is performed by weighted linear combination in the work space of the arm. However, their behaviors are not vision-based. [2] and [9] present a system for mobile manipulation where obstacle avoidance is activated in parallel with other behaviors. However, in the experiment reported in those papers obstacle avoidance only generates controls to the vehicle joints and not to the arm's.

In the above context, the main new contributions of this paper are:

- a suitable set of basic, unitary behaviors for a pick & place task;
- a new approach for visual behaviors, called *camera-based*, based on the control of the camera's motion in camera space; and
- a *hierarchical* approach to compose manipulation behaviors in which several behaviors can run concurrently, and their outputs are combined using fuzzy command fusion.

Experiments performed on a real robot arm show that our system produces a smoother trajectory than the one obtained by using crisp arbitration of behaviors, and it is more reactive to unexpected events than systems based on sequential activation.

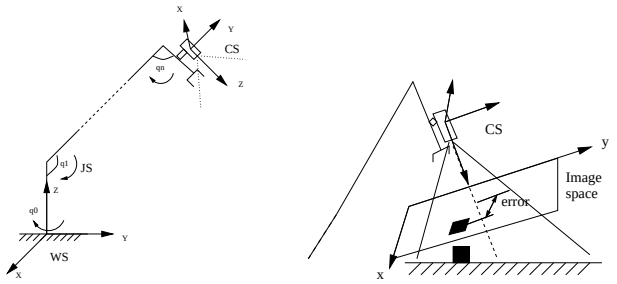

Figure 1: *The arm and camera configuration.*

2 A suite of behaviors for manipulation

A robot arm is a complex plant with usually more than 3 degrees of freedom, and it is typically controlled by giving desired joint angles (in joint space, JS) or by giving the desired external coordinates of the gripper (in work space, WS) [17]. Most industrial manipulators are controlled off-line by specifying a fixed trajectory for the gripper. If the environment is not totally predictable, however, visual feedback may be used to determine the needed parameters on-line. In this case, an additional space, the image space (IS), must be considered: the IS is the space in which the object's parameters can be measured. The relation between the IS and the WS (or the JS) depends on the arm-camera configuration. Fig. 1 illustrates this relation for the "eye-in-hand" configuration considered in this paper.

The first problem encountered when designing a behavior-based control system for manipulation is to define an adequate decomposition of manipulation tasks into a set of basic behaviors. By "adequate" we mean here that: (i) each behavior in the set implements a *simple* control strategy; and (ii) a large variety of tasks can be performed by *combining* these behaviors in different ways. While there are *de-facto* standard sets of behaviors for mobile robots (corridor following, go-to-point, door crossing, etc.), the literature offers few examples of behaviors for manipulation.

A possible way to arrive at a set of basic behaviors for manipulation is to analyze the typical manipulation tasks that we want to perform, and then decompose these tasks into basic manipulation primitives [13]. These primitives can then suggest a set of behaviors needed to implement them, for a given arm and sensor configuration. Typical primitives include point-to-point motion, object relative motion, contour following, grasping, pulling, and pushing, as well as primitives that restrict the set of possible configurations, like collision avoidance or avoiding to occlude the object from the camera view.

In this paper, we consider pick-and-place tasks and use a camera mounted on the gripper as the main sensor. Accordingly, we have identified the following set of basic behaviors.

Center Move the gripper in order to keep the object in the center of the image.

Zoom Move the gripper forward or backward in order to keep the size of the object to a given set-point.

Surround Adjust the position of the gripper so that the object is between the fingers.

Catch Grasp the object and move up a little.

Search Visually explore the workspace by moving the gripper until an object appears in the image.

FindFreeSpace Move the gripper toward a free position on the pallet.

PutDown Release the object on the pallet.

GoToPos Move the gripper to a predefined position.

GoHome Bring the arm to its home configuration.

Interestingly, these behaviors are defined from the point of view of the sensors and actuators available to the robot and not from the point of view of the external observer.

More complex behaviors can be obtained by behavior composition: for instance, we show below how to implement an "Approach" behavior by composing the Center and the Zoom behaviors. The above behaviors can be used for different tasks: for example, the Search and Center behaviors can be used to perform visual tracking, while the Approach, Surround and Catch behaviors can be combined to perform a pick-up operation. In Section 5, we show a few experiments related to the latter task.

3 Behavior design

Some of the above behaviors are defined in terms of the relative position between the gripper and a given object, and thus require input from the camera: in our case, the size and position of the target object in the image space (Fig. 1). Other behaviors only consider the position of the gripper and can be implemented by directly generating controls to the joints. In this section, we focus on the former class of behaviors, since the latter one is much simpler.

Vision-based behaviors take input in the image space (IS) and must eventually control the robot joints in joint space (JS). This problem is known as the visual servoing problem [4]. In most approaches to visual servoing, the controller either maps the IS directly into the JS (image-based visual servoing), or it maps it to the WS which is then mapped into the JS (position-based visual servoing) — see Fig. 2. Image-based approaches must compute the inverse

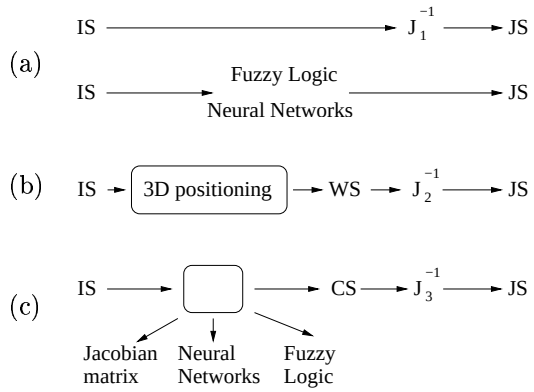

Figure 2: *Approaches to visual servoing. (a) Image-based. (b) Position-based. (c) Our approach.*

of the Jacobian transformation J from joint angles in JS to features in IS. Usually this Jacobian is a large non-square matrix that is difficult to invert and has singularities. Some people proposed alternative techniques to compute the transformation from IS to JS, but the complexity remains: for instance, [7] uses fuzzy control rules but still needs more than 120 rules for a grasping task. Moreover, the IS $\rightarrow$ JS mapping is task dependent, and a different one must be defined for each different task.

Position-based approaches, on the other hand, need an accurate estimate of the 3D position of the target object in external coordinates. This is difficult to obtain, especially in eye-in-hand configuration due to poor depth estimation.

In this paper, we propose a different approach to implement vision-based control using an intermediate space, which we call the camera space (CS). Correspondingly, we call our approach *camera-based approach*. The camera space is a 3D Cartesian space with its origin at the center of the camera and its z axis along the optical axis of the camera, as shown in Fig. 1. The controller (behavior) maps input from the IS to control values defined in the CS. These are then transformed to joint controls by a Jacobian transformation. The Jacobian in this case is well-known, since it only depends on the arm kinematics. Moreover, this Jacobian is the same for all vision-based behaviors: the only part that changes from one behavior to the other is the IS $\rightarrow$ CS map. This map is simple, since the transformation between the IS and CS spaces preserves the orthogonality between x and y: for instance, we can center an object in the image by moving the camera vertically and horizontally in CS depending on the individual x and y offsets of the object in IS.

Each mapping from IS to CS defines a specific visual behavior. These mappings can be defined in several ways (Fig. 2): in our approach, we use fuzzy rule-

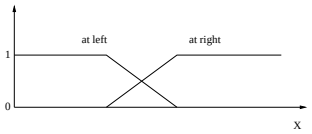

Figure 3: *Membership functions for the Left and Right conditions used in the Center behavior.*

based control systems [11, 14, 7]. The main motivation for this is that, once we introduce the notion of camera space, it becomes very easy for a designer to write the control rules that implement basic vision-based motions like "center" or "zoom." No analytical model of the system is needed to do so. For example, the following rules implement the Center behavior.

IF object at left in the image **THEN** GO(LEFT)
IF object at right in the image **THEN** GO(RIGHT)
IF object is upper in the image **THEN** GO(DOWN)
IF object is lower in the image **THEN** GO(UP)

The rule conditions are evaluated from the features in IS, and produce a truth value between 0 (fully false) and 1 (fully true). Fig. 3 shows the truth value, or *membership function*, of the first two conditions as a function of the x position of the object in the image. These functions are defined by the designer, and depend on how the object should be centered in the image. The consequent of the rules indicate which control variable should be affected, and how: the first two rules involve movements along the x axis in CS, the other two along y. Each rule affects the corresponding control variable by an amount that depends on the truth value of its condition: as a result, smaller or larger adjustments to the camera position will be generated depending on how much the object is close to the center of the image.

4 Complex behaviors

Complex behaviors are built by *combining* simpler behaviors together. As we have seen in the Introduction, most current behavior-based approaches to manipulation use a crisp behavior selection schema: in each situation, exactly one behavior is selected and is given complete control of the effectors. Selection is often based on a pre-defined sequence. In our approach, by contrast, we select behaviors in a purely reactive way using situation-action rules, and we fuse the output of concurrent behaviors by using mechanisms from fuzzy logic. Our approach is inspired by the one suggested in [16].

We write complex behaviors using fuzzy meta-rules of the form

IF <condition> **THEN** USE(<behavior>)

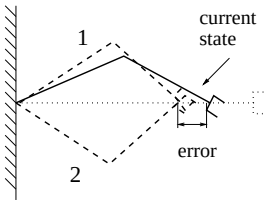

Figure 4: *The fusion problem in Joint space.*

For each such rule, the controller evaluates the truth value, in range [0,1], of <condition> and activates <behavior> at a level that corresponds to this value. Several behaviors may be active at the same time: in this case, their outputs are fused by a weighted combination according to the respective activation levels. Fusion is performed on each control variable independently. For instance, the following meta-rules implement the Approach behavior as a combination of the basic behaviors Zoom and Center.

IF object is not close to the border
THEN USE(ZOOM)
IF object is not centered
THEN USE(CENTER)

As in the case of the rules for basic behaviors, the rule preconditions are evaluated on the features in the IS. The result of executing the Approach behavior is a smooth motion of the camera/gripper that simultaneously keeps the object in the center of the image and makes its size grow.

There are two key characteristics of our behaviors that allow us to combine them in this way:

1. behaviors are fuzzy, and there are well established techniques to perform fuzzy fusion of the output of fuzzy behaviors;[16, 15]

2. the control values are defined in the camera space, which is a linear Cartesian space, as opposed to the joint space.

To understand the impact of the last point, consider the example shown in Fig. 4. Two behaviors concurrently control a two degrees of freedom arm. The first behavior has the goal to move the gripper closer to the base. Using the inverse kinematics, this behavior choses configuration 1 (the solution closest to the current state) and generates the corresponding joint angles in joint space. The second behavior has the goal to point the camera/gripper to a given direction, and thus generates the joint angles corresponding to configuration 2. Fusing the outputs from these two behaviors in JS by, say, linear combination would result in joint angles that bring the gripper to the farthest position from the arm base: a configuration

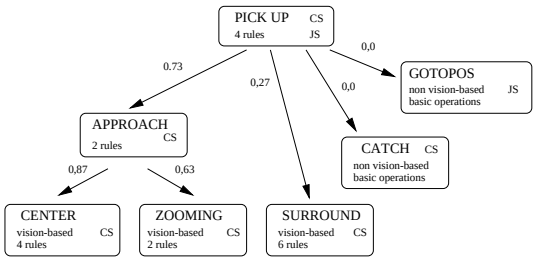

Figure 5: *The hierarchical composition of used behaviors*

that does not satisfy either behavior. The problem is that the control values in joint space are dependent on each other and cannot be combined individually. By contrast, control values in CS are orthogonal.

We can use complex behaviors to form even more complex ones, thus creating a *behavior hierarchy.* For instance, the above Approach behavior can be composed with other behaviors into a PickUp behavior using the following meta-rules:

IF object is far from the gripper
THEN USE(APPROACH)
IF object is close to the gripper
THEN USE(SURROUND)
IF object is between fingers of the gripper
THEN USE(CATCH)
IF object is catched
THEN USE(GOTOPOS)

Fig. 5 shows the corresponding hierarchy of behaviors in a graphical form. The values on the links indicate the activation values of each behavior, derived from the condition of the corresponding meta-rule, at one point of execution. Note that these values do not need to add up to one. Also note that the conditions in the meta-rules used to decide between JS-behaviors (e.g., GoToPos) are crisp. This means that only one JS-behavior at the time will be active, and therefore we never need to perform command fusion of JS-behaviors which, as mentioned above, is problematic. Finally, the figure lists the number of fuzzy rules used in each behavior: the total number of rules for the pick-up task is 18, thus indicating the simplicity of our approach.

5 Experiments

We have implemented the above PickUp behavior on a 5DOF Scortec ER1 arm, which is shown in Fig. 6, and we have tested this behavior in tens of experiments where the task was to pick up a selected block from a pallet beside the arm. Blocks were placed at a random configuration in each experiment. Images were taken from a web camera mounted on the top

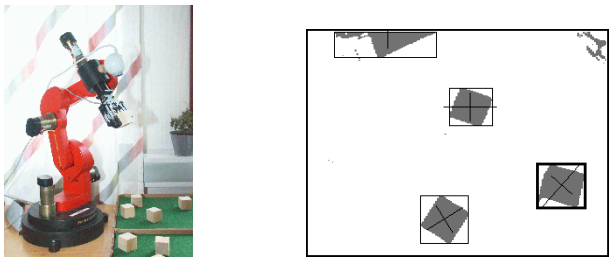

Figure 6: *Left: The used arm. Right: The image space with 3 boxes in it.*

of gripper. The extracted feature from the image are size, centroid and minor/major axis of all blocks. The target block is selected by clicking on it in the image: here, it is the rightmost one in Fig. 6 (right).

Fig. 7 shows the result of a typical run. The top picture shows a plot of the activation values for each behavior over time. The bottom picture shows the trajectory of the gripper in the 3D space, together with its projections on the xy, xz and yz planes. At the beginning only the Approach behavior is activated, which calls its two sub-behaviors Center and Zoom. Since the precondition for the Zoom behavior requires that the object be far from the edge of the image, this behavior is initially not active, and it becomes gradually more active as the Center behavior brings the block closer to the center.[1] Later (time 4 in the plot), Zoom and Center co-operate to bring the gripper close to the block. The activation level of Center depends on the distance of the block from the center of the image. When the gripper is close to the block (time 20 in the plot) the Approach smoothly starts to be de-activated in favor of the Surround behavior. When the gripper is placed around the block, the Catch behavior is used to grasp the block and then the GoToPos behavior brings the gripper back to a safe holding position. The gripper trajectory shows the smoothness of the resulting motion, mainly due to the use of fuzzy command fusion to interpolate between concurrent, partially active behaviors.

The next two experiments have been designed to test the advantages of fuzzy fusion of concurrent behaviors, and of the use of reactive arbitration rules.

Fig. 8 shows an execution of the previous task using crisp arbitration instead of fuzzy fusion. This run was obtained by replacing concurrent activation of behaviors by a winner-take-all strategy where only the most active behavior is in control of the effectors at any given time. This is visible, for instance, in the alternance of activations between the Center and Zoom behaviors.The resulting trajectory is

[1]The Search behavior may be used in case the block goes out of the image

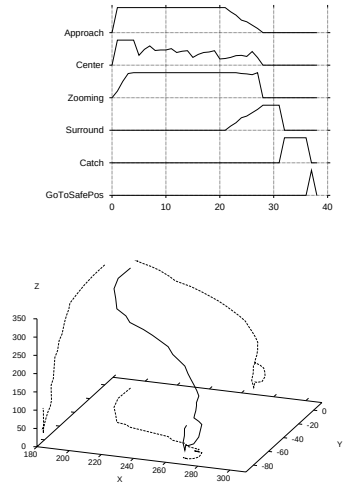

Figure 7: *Sample execution using fuzzy concurrent behavior.*

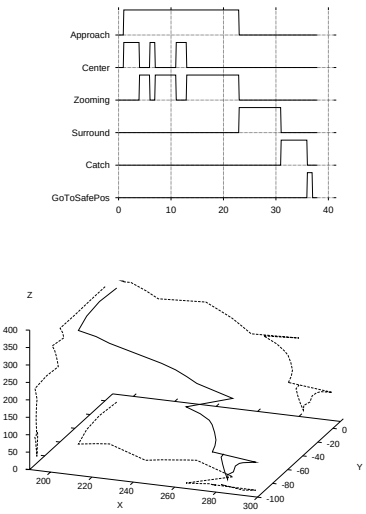

Figure 8: *Using crisp arbitration.*

rather discontinuous, especially during the Approach maneuver.

In order to test the advantages of the reactive rule-based arbitration as opposed to a purely sequential arbitration, we have set up an experiment in which an unexpected event occurs. The target block was put on a black object and, once the Surround behavior was started, it was manually tipped down to the pallet. Fig. 9 shows the result of this experiment. At time 19 the block was tipped down. Since the block now became far from the gripper, the Surround behavior became de-activated and the Approach was re-activated, thus bringing the gripper to the new position of the block. When the block was reached, the Surround, Catch and GoToPos behaviors were re-activated as in the first experiment. Notice that this situation would be problematic if the behaviors were organized, say, in a finite state machine [10, 13, 12]:

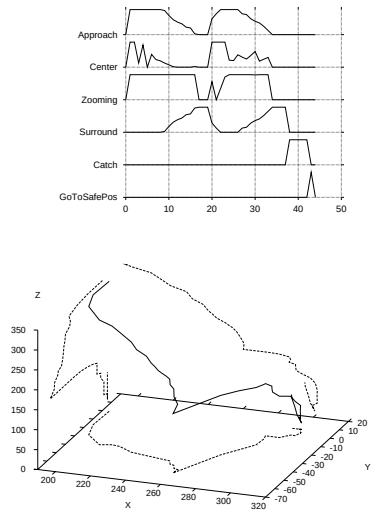

Figure 9: *Recovery from an unexpected event*

in this case, an appropriate state transition should have been designed into the FSM to account for this possible event.

6 Conclusions

We have defined a basic set of generic manipulation behaviors that provide an ontology for pick-and-place tasks, and proposed a new camera-based approach to implement these behaviors. This approach greatly simplify behavior design: we only need 18 fuzzy rules in our PickUp task, compared with the over 120 used in [7] and the over 100 used in [6] (who has a similar set of behaviors). This approach is also expected to make behaviors more portable across different platforms since the IS $\rightarrow$ CS controllers do not depend on the configuration of the arm.

We have also defined a way to hierarchically compose basic behaviors into more complex behaviors using fuzzy rules, and shown that it results in smooth trajectories and in improved robustness with respect to unexpected sequences of events.

Future work will mostly focus on the extension of this behavior-based approach to different tasks, and its integration in a system for mobile manipulation.

References

[1] R. Brooks. A layered intelligent control system for a mobile robot. *IEEE J. Robotics Automat.*, RA-2:14–23, 1986.

[2] J.M. Cameron, D.C. MacKenzie, K.R. Ward, R.C. Arkin, and W.J. Book. Reactive control for mobile manipulation. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 228–235, 1993.

[3] J.H. Connell. A behavior-based arm controller. *IEEE Transactions on Robotics and Automation*, 5(6):784–791, 1989.

[4] P. Corke. Visual control of robot manipulators – a review. *Robotic and Autonomous Systems*, 7:1–31, 1993.

[5] P. Dassanayake, K. Watanabe, and K. Izumi. Fuzzy behavior-based control for a task of three-link manipulator. In *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics*, pages 776–781, 1999.

[6] R. De Giuseppe, F. Taurisano, C. Distante, and A. Anglani. Visual servoing of a robotic manipulator based on fuzzy logic control. In *Proc. IEEE Int. Conf. on Robotics and Automation*, p. 1487–1494, 1999.

[7] C.S. Kim, W.H. Seo, S.H. Han, and O. Khatib. Fuzzy logic control of a robot manipulator based on visual servoing. In *Proc. IEEE Int. Symp. on Industrial Electronics*, pages 1597–1602, 2001.

[8] P.J.A. Lever, Fei-Yue Wang, Xiaobo Shi, and Deqian Chen. A fuzzy-behavior-based approach for controlling mining excavator bucket/rock interactions. *Conf. Record of the 1994 IEEE Industry Applications Society Annual Meeting*, 3:2126–2133, 1994.

[9] D.C. MacKenzie and R.C. Arkin. Behavior-based mobile manipulation for drum sampling. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2389–2395, 1996.

[10] K. Nagatani and S.I. Yuta. An experiment on opening-door-behavior by an autonomous mobile robot with a manipulator. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 45–50, 1995.

[11] K.M. Passino and S. Yurkovich. *Fuzzy Control.* Addison-Wesley Longman, Inc., 1998.

[12] L. Petersson, D. Austin, and D. Kragic. High-level control of a mobile manipulator for door opening. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2333–2338, Takamatsu, Kagawa, Japan, 2000.

[13] G.C. Pettinaro. *Basic Set of Behaviours for Programming Assembly Robots.* PhD thesis, University of Edinburgh, 1996.

[14] Palm Rainer. Control of a redundant manipulator using fuzzy rules. *Fuzzy Sets and Systems 45*, pages 279–298, 1992. North-Holland.

[15] A. Saffiotti. The uses of fuzzy logic in autonomous robot navigation. *Soft Computing*, 1(4):180–197, 1997. Online at http://www.aass.oru.se/~asaffio/.

[16] A. Saffiotti, E. H. Ruspini, and K. Konolige. Blending reactivity and goal-directedness in a fuzzy controller. In *Proc. of the 2nd IEEE Int. Conf. on Fuzzy Systems*, pages 134–139, San Francisco, California, 1993.

[17] L. Sciavicco and B. Siciliano. *Modelling and Control of Robot Manipulators.* Springer, 1959.

[18] T.G. Williams and Hardy N. Behavioural modules for force control of robot manipulators. In *Proc. IEEE Int. Symp. on Robot Control*, 2000.

[19] M.M. Williamson. Postural primitives: Interactive behavior for a humanoid robot arm. In *Proc. of SAB*, Cape Cod, MA, USA, 1996.