Formulation and Implementation of Relational Behaviours for Multi-Robot Cooperative Systems

Bob van der Vecht[1,2] and Pedro Lima[2]

[1] Department of Artificial Intelligence, University of Groningen
Grote Kruisstraat 2/1, 9712 TS, Groningen, Netherlands
[2] Institute for Systems and Robotics, Instituto Superior Técnico,
Av. Rovisco Pais, 1 - 1049-001 Lisboa, Portugal
bvecht@isr.ist.utl.pt, pal@isr.ist.utl.pt

Abstract. This paper introduces a general formulation of relational behaviours for cooperative real robots and an example of its implementation using the pass between soccer robots of the Middle-Sized League of RoboCup. The formulation is based on the Joint Commitment Theory and the pass implementation is supported by past work on soccer robots navigation. Results of experiments with real robots under controlled situations (i.e., not during a game) are presented to illustrate the described concepts.

1 Introduction

Showing cooperation among robots from a team is probably one the top goals of RoboCup related research. Furthermore, it is desirable to formulate cooperative behaviours within a formal framework extendable to applications other than robotic soccer. Surprisingly, not many references to work on these two topics can be found in the RoboCup-related literature or in other publications referring cooperation among *real* robots.

An example of a tool that enables the development of applications based on the Joint Commitment Theory [1], including communications, has been thoroughly reported by Tambe, e.g., in [9]. Yokota et al. use explicit communication to achieve cooperation and synchronization in real robots. However, no explicit logical commitments are described [10]. Emergent cooperative behaviour among virtual agents is also described in [7], where a pass behaviour by implicit communication (observing the other robots behaviour) is implemented in a team of RoboCup Simulation League, and [6], where also a pass between RoboCup Simulation League agents is described such that conditions to pass are learned by a neural network. Again, there is no commitment between players, therefore the relational behaviour may be kept by one of the team mates even if the other has to withdraw its relational behaviour.

In this paper a general formulation of relational behaviours for cooperative real robots is introduced. Most of the paper describes an example of implementation of this formulation using the pass between soccer robots of the Middle-Sized

League (MSL) of RoboCup. The formulation is based on the Joint Commitment Theory [1], and the pass implementation is supported by past work of the ISocRob team on soccer robots navigation [3].

In the pass relational behaviour, two participants set up a long term commitment, in which several individual behaviours are executed. One of the robots is referred to as the *kicker*; he starts having the ball and will try to kick the ball in the direction of the other robot, the *receiver*, who has to intercept the ball. In order to accomplish a pass successfully, two components of the commitment should be working well: the synchronization of both players' actions and the execution of their individual skills. The synchronization has to be achieved by communication. In Fig. 1, an example of the pass commitment has been illustrated by two state machines.

In the following sections, the theory behind the pass commitment as shown in Fig. 1 will be described. First the individual decision making and the behaviour synchronization will be explained and then the individual primitive behaviours. Results of experiments with real robots under controlled situations (i.e., not during a game) are also presented to illustrate the key primitive behaviours described.

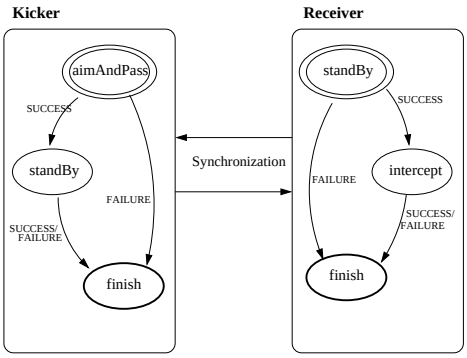


Fig. 1. Simple example of the pass commitment

2 Decision Making and Synchronization

In Fig. 1, several individual behaviours can be found within the commitment. At any time the participants have to select the correct primitive behaviour individually. The architecture that is used for behaviour coordination and decision making considers three types of behaviours: organizational, relational and individual [4], [5]. Organizational behaviours concern decisions involving the whole team, e.g., player role selection such as defender or attacker. The relational behaviours concern more than one player. Commitments among team mates are established here. Finally, at the individual level, primitive behaviours are selected, and motor commands are used to influence the environment in the desired way.

Behaviour selection is done in a module called the *logic machine*, explained in [8]. In this module, only the organizational and the individual levels were implemented. The work described here addresses the relational layer. The layers are processed sequentially. This means for the robot that he first chooses a role, next he selects a commitment, and the individual behaviour is selected after knowing the robot's role and commitment.

2.1 Relational Behaviour Selection

Joint Commitment Theory is used in this work to select relational behaviours [1]. Predefined logical conditions can establish a commitment between two agents. Once a robot is committed to a relational behaviour, he will pursue this task until one or more conditions become false, or until the goal has been accomplished. In the described project, the initiative for a relational behaviour is taken by one of the agents, who sets a *request* for a relational behaviour. A potential partner checks if the conditions to *accept* are valid. If so, the commitment is established. During the execution of the commitment the changing environment can lead to failure or success at any time. In that case the commitment will be ended.

In general, within a commitment three phases can be distinguished: *Setup*, *Loop* and *End*. During the setup and ending of a commitment, a robot is not executing a relational behaviour. The *logic machine* will not select any relational behaviour, so the commitment will be ignored during the primitive behaviour selection. Only in the *Loop* phase participants will select primitive behaviours concerning the commitment in order to achieve their joint goal.

2.2 Primitive Behaviour Selection

The selection of a primitive behaviour within the *Loop* phase of a commitment will be explained using the example of the pass commitment of Fig. 1. Three individual behaviours can be found there; **standBy** for both participants, **aimAndPass** for the kicker and **intercept** for the receiver.

The pass commitment has been split up in several states from the beginning until the end, referred to as *commitment states*.

- *request* and *accept* in the *Setup* phase.
- *prepare* and *intercept* in the *Loop* phase.
- *done* and *failed* in the *End* phase.

In general, the states in the *Setup* and *End* phase will be the same for all commitments, only the *Loop* changes. Splitting the *Loop* phase in states allows synchronized execution of the pass. Here, each commitment state is linked to (a set of) primitive behaviours for both robots, see Table 1. When the commitment runs as planned, the pass states will be run through sequentially, from *request* until *done*. An error at any time can lead to the state *failed*. Note that pass states in the *Setup* and *End* phase do not lead to a primitive behaviour from the relational pass. Table 1 describes the pass commitment of Fig. 1. New commitments

or other versions of commitments can be created under the same framework. One can change the individual behaviours that have been linked to the commitment states, or extend the *Loop* with more pass states and behaviours.

Table 1. Primitive behaviour selection by the *logic machine* in all pass states during the pass commitment.

	Setup		Loop		End		Default
Commitment States:	*request*	*accept*	*prepare*	*intercept*	*done*	*failed*	*none*
Kicker	—	—	aimAndPass	standBy	—	—	—
Receiver	—	—	standBy	Intercept	—	—	—

2.3 Synchronization

To synchronize the behaviours, the participants will use explicit communication. Four variables, containing the identities of the participants and their commitment states, are kept in the agent's memory. Each of these four variables will be sent to the other participant in the relational behaviour when it is changed.

Following setup conditions, one agent sets a *pass request*, and a partner can enter the *accept* state. When the commitment has been established, the following rules will be looped:

- **Synchronize** commitment state with the state of the parter, if partner has moved on to a next state.
- **Switch** to new state by yourself, if predefined *switch conditions* allow that.
- Select a basic behaviour using Table 1.

This happens until the *done* or *failed* state is reached. Then the commitment will be finished and all variables will be cleared.

Synchronization is achieved at each moment an agent switches to a new commitment state, since the partner will always follow. By synchronizing commitment states each iteration, the commitment states of both agents can only be one step away from each other, and this difference will be corrected in the next loop. It is possible to let agents execute a sequence of primitive behaviours in one state. They will run asynchronously.

3 The Individual Behaviours

The primitive behaviours are running in a *control* module. Here the world situation is evaluated continuously, and motor commands will be send to the robot in order to influence the environment. The world model of the robot can be seen as a map with all identified objects in a xy-coordinate system. In the **standBy** behaviour, the robot will stay at the same position and will try to keep its front towards the ball position. This behaviour has been implemented earlier in the ISocRob team [5]. The other two behaviours, **aimAndPass** and **intercept**, had to be developed from scratch.

3.1 The aimAndPass Behaviour

When the pass commitment is started, the *kicker* has the ball. He wants to rotate with the ball to a certain direction and then shoot. This has been implemented in the **aimAndPass** behaviour.

In earlier research a controller has been developed to dribble with a ball to a goal posture [3]. A navigation algorithm described in [3] is used which takes the robot to a goal posture while avoiding obstacles, using a modified potential fields method that takes into account the non-holonomic nature of the robot. The motor commands given by this navigation algorithm are passed to a dribble filter which adapts them to the extra constraints of keeping the ball close. Parameters of the navigation algorithm can be changed to return a very strong angular acceleration towards the desired direction. Since the dribble filter stays with the dribble constraints, a controller is achieved which rotates with ball to the desired direction at the maximum turn angle.

3.2 The intercept Behaviour

In order to intercept a moving ball, the **intercept** behaviour controller has been designed. The literature on visual servoing has solutions for similar problems [2], but only for a situation when the robot tracking the object is not moving within an environment cluttered with obstacles. Therefore, the solution described here is based on the previous mentioned navigation algorithm, [3], using a modified potential fields method. The intercept controller continuously estimates the interception point given the positions of ball and robot, and their predicted path. After a certain time t, position of ball and robot should be the same. The corresponding xy-coordinates indicate the interception point and they will be passed to the navigating algorithm. Interception point estimation is shown in Fig. 2.

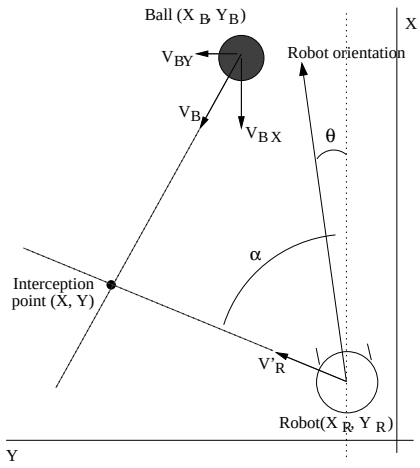


Fig. 2. Interception point calculation.

Figure 2 shows a ball path estimation with position (x_b, y_b) and velocity (v_{bx}, v_{by}), and a robot with position (x_r, y_r). For the estimation, a constant velocity is assumed.

The robot path is estimated with the assumptions that the robot will move with a constant velocity in a straight line towards the interception point. This implies that the orientation of the robot will not change during the movement to the interception point and thus the initial orientation is ignored. An assumed average forward speed v'_r is used. Now, the robot path can be given as function of time, using the robot's current position (x_r, y_r), its absolute speed v'_r (given in the x-direction) and the orientation towards the interception point α, as shown in Figure 2. Since the interception point is still to be calculated, α is still unknown.

Variables α and t can be eliminated from respectively the x and y-component from both path estimations. The calculated t represents the time it takes for the ball and the robot to get to the interception point. By replacing t in the ball path estimation, xy-coordinates for the interception point are achieved.

Note that the assumption that the robot moves in a straight line to the interception point, may lead to an error in the estimation of the interception point, since the robots are non-holonomic. However, the estimation is iteratively applied and becomes more accurate at each new iteration step, until when the robot faces the correct heading, where the straight line motion assumption is fully correct.

4 Implementation and Results

4.1 Implementation of the Commitment

The theory for joint commitments has been implemented in Nomadic Super Scout II robots of the Robocup Middle-Sized League team ISocRob. The decision making and the synchronization have been implemented successfully. The framework for joint commitments requires the definition of the following aspects of a relational behaviour:

- List of commitment states
- Setup conditions of the commitment
- Switch conditions to switch between commitment states
- Related individual behaviours within each commitment state

The framework takes care of all communication and the synchronized execution of the relational behaviour.

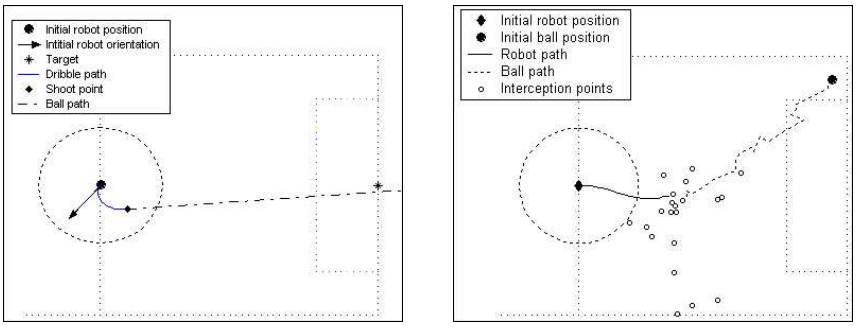
For the implemented version of the pass commitment, mentioned in section 1, the following pass situation has been defined: a defender has the ball on its own half of the field and he will pass the ball to an attacker, who is on the other half of the field. The division in states and the related primitive behaviours are shown in Table 1. When the *kicker* has successfuly executed the aimAndPass behaviour, he switches to the *intercept* state. The *receiver* moves to the same state, and starts the intercept behaviour.

Note that in this case the *kicker* switches the pass state to *intercept*, and by doing so he tells the *receiver* to start the intercept behaviour. More reactive approaches of the pass commitment can define conditions that allow the *receiver* to switch to the *intercept* state, following his own observations.

The results of the execution of the pass behaviour are strongly related to the results of the individual behaviours. AimAndPass and intercept will be analysed in the following sections.

4.2 AimAndPass

The **aimAndPass** behaviour has been implemented following the methods described in previous sections. The performance of the behaviour is as expected. Figure 3(a) shows the result of a test with a real robot on a MSL soccer field. The robot start position is at the middle of the field, and his target is the middle of the goal which is located diagonally behind him. The robot turns with the ball and shoots in the desired direction. However, since the algorithm works with coordinates, errors can occur when the robot is badly localized. A small difference in the kicker orientation leads to a significant spatial error when the distance to the target grows. Those errors caused by localization can be avoided if the camera is used directly to determine the target direction.



(a) (b)

Fig. 3. Individual behaviour results: a) aim-and-pass; b) intercept ball

4.3 Intercept

For the **intercept** behaviour, the assumed average robot speed, v'_r, has been set to 0.5 m/s. An example of the interception of a moving ball by a real robot in a MSL soccer field is shown in Fig. 3(b). Clearly it can be seen that the robot takes the ball velocity into account, and moves directly to the interception point. In this example, the ball rolls with a speed of 1.2 m/s in a straight line. The ball

path that is shown is the path as it is observed by the robot. Balls with a high velocity are likely to bump away after the interception.

5 Conclusions

In this paper the general formulation of relational behaviours among real robots, based on the Joint Commitment Theory, has been introduced through an illustrative example concerning a pass behaviour between RoboCup MSL robots. The formulation uses individual decision making and behaviour synchronization among intervening robots and has been tested successfully during laboratory games without an opponent. Results of the implementation of the key individual behaviours (**aimAndPass**, **intercept**) in real robots were presented.

Future work will concern the development of new relational behaviours under this framework, as well as the refining of the individual behaviours, particularly by using team mates visual recognition to eliminate the self-localization error and the required communication during the **aimAndPass** behaviour, and by providing the robots with force-controlled kicking ability, so as to enable ball interception by the receiver robot, by reducing the ball speed for pass kicks, as compared to goal kicks.

References

1. P. R. Cohen, H. J. Levesque, "Teamwork". *Nous*, Vol 35, pp. 487-512, (1991)
2. P. Corke, *Visual Control of Robots: High-performance Visual Servoing*, Research Studies Press LTD., (1996)
3. B. Damas, L. Custódio, P. Lima, "A Modified Potential Fields Method for Robot Navigation Applied to Dribbling in Robotic Soccer", *RoboCup 2002 Book*, Springer-Verlag, Berlin (2003)
4. A. Drogoul, A. Collinot, "Applying an Agent-Oriented Methodology to the Design of Artificial Organizations: A Case Study in Robotic Soccer", *Autonomous Agents and Multi-Agent Systems*, Vol 1, pp. 113-129, Kluwer Academic Publ., (1998)
5. P. Lima, L. M. Custódio, et al., "ISocRob 2003: Team Description Paper", Proceedings of the RoboCup 2003 Symposium, Padova, Italy (2003)
6. H. Matsubara, I. Noda and K. Hiraki, "Learning of Cooperative Actions in Multi-Agent Systems: a Case Study of Pass in Soccer.", *Adaptation, Coevolution and Learning in Multi-agent Systems: Papers from the AAAI-96 Spring Symposium*, SS-96-01, pp. 63-67, (1996).
7. E. Pagello A. D'Angelo, F. Montsello, F. Garelli, C. Ferrari, "Cooperative Behaviors in Multi-Robot Systems Through Implicit Communication", *Robotics and Autonomous Systems*, Vol 29, No. 1, pp. 65-77, (1999)
8. V. Pires, M. Arroz, L. Custódio, "Logic Based Hybrid Decision System for a Multi-robot Team", *8th Conference on Intelligent Autonomous Systems*, Amsterdam, The Netherlands, (2004)
9. M. Tambe, "Towards Flexible Teamwork", *Journal of Artificial Intelligence Research*, Vol 7, pp.83-124, (1997)
10. K.Yokota, K. Ozaki, N. Watanabe, A. Matsumoto, D. Koyama, T. Ishikawa, K. Kawabata, H. Kaetsu, and H. Asama, "Cooperative Team Play Based on Communication.", *RoboCup 1998 Book*, Springer-Verlag, Berlin (1999)