Technical Report On Joint Actions For An Aibo Team

Areej Mahdi [amahdi@science.uva.nl]
Mark de Greef [mgreef@science.uva.nl]
Dave van Soest [dsoest@science.uva.nl]
Isaac Esteban [iesteban@science.uva.nl]

February 2^{end}, 2006

Supervisor: Arnoud Visser

Index

Abstract
In this report we describe new behaviours developed to perform joint actions in an Aibo soccer game. In particular, we will focus on passing the ball between Sony Aibo robots using collaborative behaviour. This is actually one of the challenges in the four-legged RoboCup 2006 [1]. We will describe our approach, which includes varying the joint actions behaviour by using two methods: expectation and message based. A number of experiments have been carried out to test and evaluate these joint behaviours.

1 Introduction

We have been assigned to develop joint actions and collaborative behaviour for Aibo robots in a soccer game. This is presented by the ball-passing challenge where three Aibo robots pass the ball to each other. This challenge is part of the Sony Four Legged Robocup 2006 [1]. The Robocup league is organized every year and each year new rules and challenges are introduced to evolve towards a more natural soccer play. The goal set is to let a robot soccer team compete with a human team in the year 2050. In our implementation of the ball passing challange we will use the Dutch Aibo team code 2005 [2][3][4]. The current implementation of playing soccer uses some messaging and decision-making. Each robot in the soccer team, which consists of four robots, is assigned a specific roll in the game. The different rolls are: goalkeeper, striker, left- and right supporter. The ball location is communicated between the robots if the ball is not seen by one of the robots. The robot then depends on the information sent by other teammates to find the ball again. Another form of joint decision-making during the soccer match is when robots negotiate about which roll to take (striker or supporter) depending on their approximated time to reach the ball.

To implement ball-passing behaviour we need to define some new behaviour and add basic behaviours in the XABSL code and the C++ code [Appendix].

The framework we will be using is described in the following section. Then in section 2 we will describe the passing challenge in more detail as well as the changes that are required to perform this challenge. Section 3 focuses on the implementation of the ball passing and the different approaches we have taken. The performed experiments can be found in section 4 and finally the conclusions and discussion in section 5.

2 Framework

To implement the ball-passing challenge we will use the Dutch team code 2005 [2], which is based on the code of the German team 2004 [5]. The code is written in XABSL (eXtensible Agent Behaviour Language) and C++[ref]. XABSL is an XML based language used to develop complex behaviour. There is also the C++ layer where all the basic behaviours and symbols are written and defined. When a new function or a new option needs to be added, it has to be implemented in the C++ files and included in the XML files.

2.1 Code Modules

The code we will be using is divided into a substructure of four different levels. Each of these levels is defined according to the task that needs to be performed in order to play soccer. The levels are: Perception, Object Modelling, Behaviour Module and Motion Module [5].

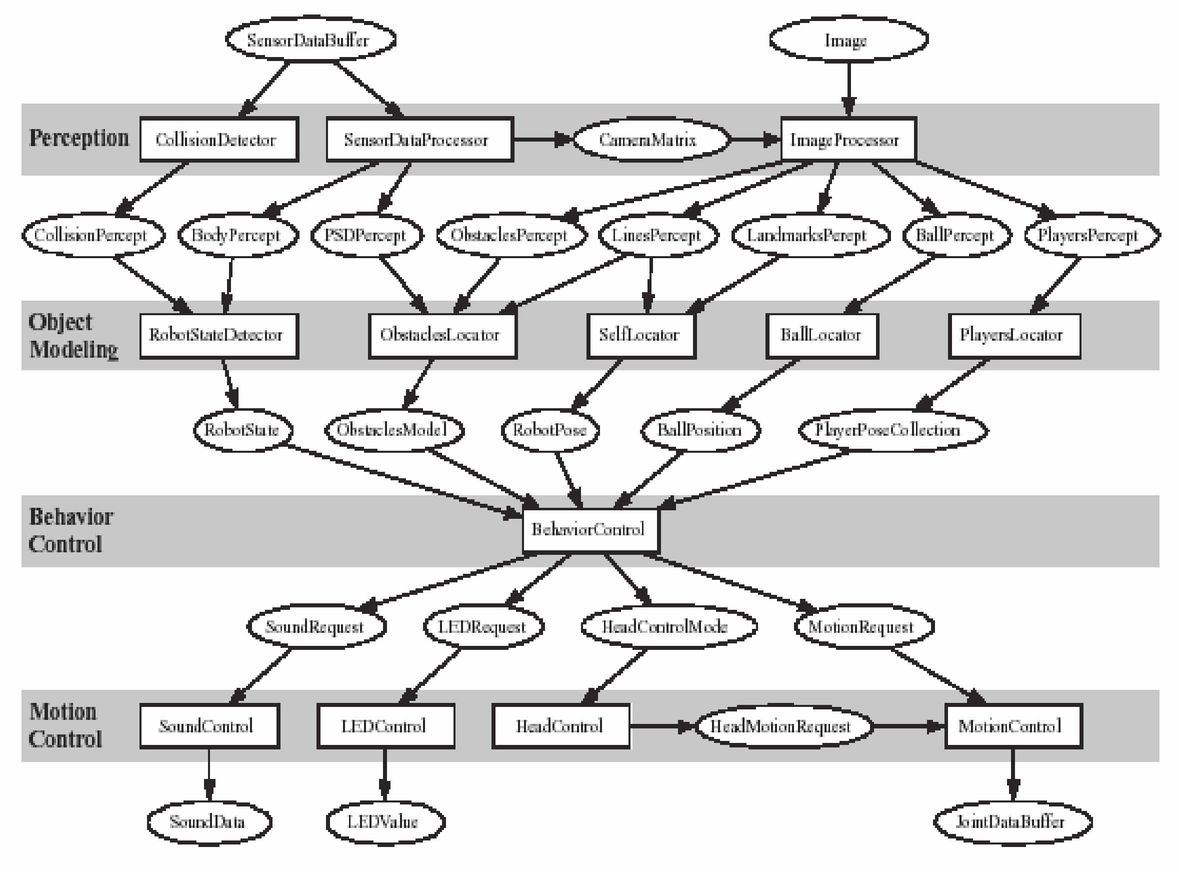


Figure 1: tasks identified by the german team 2004 for playing soccer

Our assignment is based on modifying the behaviour of the robots towards a more cooperative team play behaviour. So we will mainly be working on the BehaviourControlModule. Inaccuracies or shortcomings from other modules will not be dealt with, but will be considered. An example of that is the inaccuracy of the self-localization of the robots. We will try to work around these inaccuracies as much as possible.

2.2 Robot Control

A very useful tool that we will be using is the *RobotControl* [3]. *RobotControl* is a debugging program for the Aibo software. It is designed to increase the speed and comfort of developing the software used in the Aibos. *RobotControl* represents and visualizes nearly all the internal state of the robots such as sensor data, joint data, images, messages between modules and the world state. This is very helpful when developing new software, because this allows us keep track of what is happing in the field according to the robot. Further more, the internal representation and the behaviour modules of the robots can also be changed or replaced by other behaviour modules. Exchanging information between *RobotControl* and the Aibo robots is done using wireless LAN or the memory stick of the robots.
It is also possible to use *RobotControl* to record and store log files of the robot's data if the software used in the robot is in debug mode. These log files can be used to test the code without having to use a physical robot on the field. [3]

3 'Ball-Passing' Challenge

One of the challenges of the four-legged RoboCup 2006 is the passing challenge [1]. It requires three Aibo robots to pass the ball to each other. Each of these robots is placed inside a predefined location on the field in which the distance between the robots may differ. The ‘fooling around’ challenge is an extension of the ‘ball passing’ challenge in which an opponent robot from another team is introduced to the ball passing to try to grab the ball. Also the constraint of the predefined robot locations is removed.
To be able to perform this kind of team play, each Aibo robot has to be aware of the presence of its other teammates on the field and locate them. Another requirement for ball passing is to be able to catch and control the ball, then perform an accurate ball-kick to another teammate. To achieve this kind of behaviour some modifications in the code need to be issued concern the following aspects:

3.1 Recognizing and locating other teammates

To do this challenge, the Aibo robot must recognize and locate other teammates. The current implementation does not contain any behaviour option to search or look for other teammates in the field. The only options used are looking for landmarks and looking for the ball. However there is already some code written that can be used to search for teammates. So we will use ‘*PlayerPercept’* to detect a teammate in the field by means of vision.

In order to pass the ball to another teammate, the robot possessing the ball first needs to turn around to face one of its teammates. This means that it is not only necessary to detect whether a teammate is in the field but also to have somewhat of an accurate estimation of the other teammate's location. Locating other teammates by letting the robot’s head constantly move left and right to scan the environment is not a very good choice, because the robot can loose sight of the teammate. Instead once the teammate is seen, the robot turns with a steady head around the point that is standing on in the direction it saw the teammate. This is done until the robot comes to face the teammate. We implemented this method using a function that determines the turning angle to another teammate called ‘*DegreeToFriend’* and it is used in the behaviour ‘turn-to-friend’ [Appendix].

3.2 Ball handling

To pass the ball around, it is necessary to combining ball passing and catching skills. This begins with selecting an accurate kick to pass the ball to another teammate. When playing soccer, the kicks that the Aibo robots perform are chosen from a kick selection table. A kick is selected from this table based on the current ball position and the desired direction of the kick [3]. To pass the ball around we will use just one type of kick, which will be performed by all the robots when passing the ball. The reason behind this is that not all the kicks in the kick selection table are accurate. We also noticed that some of the kicks are too fast, so if performed it will be difficult for the teammate to control the ball and the ball will most probably exit the field. So after testing all the kicks we have chosen the ‘*hook-left kick*’. The kicks are named according to the direction in which the ball is kicked. This kick is actually performed by the right paw of the robot. It is quite accurate and the speed of the ball is good making it possible to block and control the ball by other robots. With this chosen kick, it is possible to kick the ball accurately to another teammate if the robot turns with the ball to a certain kicking-angle.
The kick can be extended later on to include also the ‘hook-right’ kick. But we have chosen the ‘hook-left’ to keep it simple.

After the ball is kicked towards a robot, that robot will try to estimate the ball’s speed and location and attempt to block it. The robot will also attempt to block the ball if the ball is within a distance range of 50 cm. In the soccer game, the distance range used by the goalie to perform a block is set at the safe distance of 85 cm. This relatively large distance range is obviously chosen because the balls kicked to the goalie

are not very accurate and have high speed. In the beginning of the Passing Challenge implementation we reduced this block distance range to 30 cm. This close distance range was initially chosen because in ball passing we use more accurate ball kicks with less ball speed. We finally set the distance to 50 cm because it results in better free ball passing play.
If the ball has low speed or is lying still, the robot will attempt to grab the ball if it is within 25 cm. This grabbing range is set to this close distance to let the robot block the ball more than it grabs the ball. If the grabbing distance is high the robot will attempt to grab the ball when it should block it.

3.3 Passing strategy

To pass the ball between the robots we will implement two approaches. The first approach is without communication or joint decision making between the robots. So every robot decides separately to approach the ball, grab it or try to block it. The main passing ball strategy in this approach is that every robot waits for the ball until the ball comes within the distance range of which the robot can grab or block it. The robots do not use communication so it is possible that a robot attempts to block or grab the ball while it is already possessed by another robot. This happens if the ball rolls away from a robot and becomes very close to another robot. Then both robots will go for the ball and attempt to grab it and perform a kick.

The second approach permits communication on the decisions between the robots. The robots communicate with each other using messages to perform a more cooperative team play. Information about the ball state (‘ball is free’ or ‘ball is possessed’) will be exchanged. The robot that is going for the ball sends out a team-message to its other teammates to let them know that the ball is possessed. When the teammates receive this message none of them will try to approach the ball. They will all wait for the ball until it is released. After the robot performs a kick, it will send another team-message to inform the other teammates that they can go for the ball. We expect this to reduce the confusion and uncertainty for the robots that might occur in some situations and allow them to perform better ball passing.

3.4 Localization

As mentioned earlier one of the essential issues in ball passing is being able to accurately locate the ball and the other teammates in the field. We have noticed that the robot is not very accurate in estimating its own absolute position on the field. Having an incorrect estimation of the robots own self-position will also effect the robots estimation of the global position of the ball and other teammates on the field. To work around this localization problem we will not use the absolute position of the robot or global distances. Removing self-positioning will allow the robot to pass the ball everywhere and not be restricted to the football field defined by the Robocup. Further more we will only be using the relative distance to estimate the ball location. We will not try to determine the distance to other teammates when we want to pass the ball. As will be seen in the next section, each team member is in charge of communicating its distance from the ball in the free play. The only variable that we will be interested in is the turning angle to face a teammate. By doing this we will not depend much on the incorrect distance estimation when passing the ball. However the perception inaccuracies will remain.

4 Implementation

To implement the passing challenge, we have designed a new behaviour control. We tried to design each behaviour and state machine to be independent. Our design consists of three main layers: initial layer, middle layer and the final layer. The first layer is the initial state in which the robot either waits for other players to join him, this is the case if he is the only robot on the field, or joins an already existing team. In the middle layer a number of state machines are defined, this is where the ball passing actually takes place. The final layer is the finish state; it is the state the robot reaches at the end of the passing game.

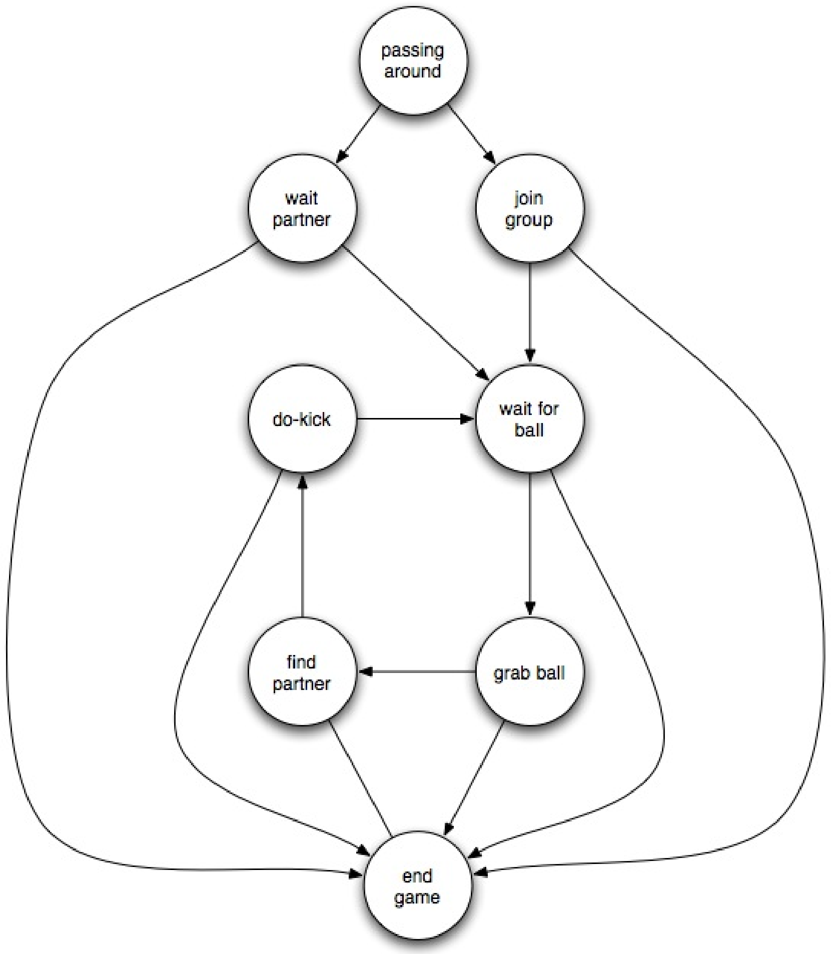


Figure 2: Behaviour design to play passing ball

We will now focus on the state machines of the middle layer and explain them in more detail. After exiting the initial layer, the robots will enter the ‘wait for the ball’ state. In this state the robots will look for the ball by performing a head scan and once the ball is found the robot will keep track of the ball. However if the ball is not found, the robot will turn around on its spot to try to find the ball.
The robots will not attempt to go after the ball if the ball is not kicked to them or is within a certain distance from them. When the robot detects that the ball is coming towards it or is within 50 cm from it, it enters the following state ’grab ball‘.

If the ball is coming towards the robot, it will make an estimation of the speed of the ball and its location. Based on this the robot may try to block the ball. The blocking methods we use are the blocks used by the goalie in a soccer match [3]. If the ball is close enough the robot will grab the ball between its paws and head to try to control it.

The next state is ‘find partner’ in which the robot that grabbed the ball chooses a teammate to pass the ball to. The robot first scans the environment for another teammate. Once a teammate is found, the robot will

turn around with the ball towards the direction of the teammate. While the robot is turning, the head of the robot is kept steady. The first teammate that the robot sees is chosen to be the teammate it passes the ball to. A more complex way to choose the teammate can be implemented for instance for the fooling around challenge, but we have noticed that our simple approach works well. After turning to the teammate the robot grabs the ball again to be sure that the ball has not rolled away while the robot was turning.
To be able to kick the ball accurately to the chosen teammate, the robot will first make a turn to compensate for the direction in which the ball is kicked. The kick is performed in the ‘do kick’ state. After that the robot returns to the ‘wait for ball’ state and waits till the ball is kicked to it again.

We have implemented the passing ball in two approaches: one with using communication and the other without any. The passing-ball behaviour that is described up till now does not use any communication. The communication that will be added involves passing information about the ball state. The player who sees the ball coming towards him sends out the team message ‘preparing kick’. The other robots that receive this message do not go after the ball and stay in the ‘wait for ball’ state until the kicking robot sends the ‘kick performed’ message. Both these messages were already implemented in the Dutch-team code 2005 based on the German code 2003. Using team messaging prevents more than one robot to go after the ball and perform a kick.

The difference in the implementation can be seen in the ‘wait-ball’ state. In figure 3, the state machine of ‘wait-ball’ is shown in the no-communication approach. The implementation of this state is previously explained. In the figure we see that the robot determines independently whether to grab the ball or not without any cooperation with the other teammates.
In the communication approach however, the robot blocks, grabs or approaches the ball only if the ball is in ‘free’ state, see figure 4. This ball status is given when no other teammate has sent the message that it is going for the ball.

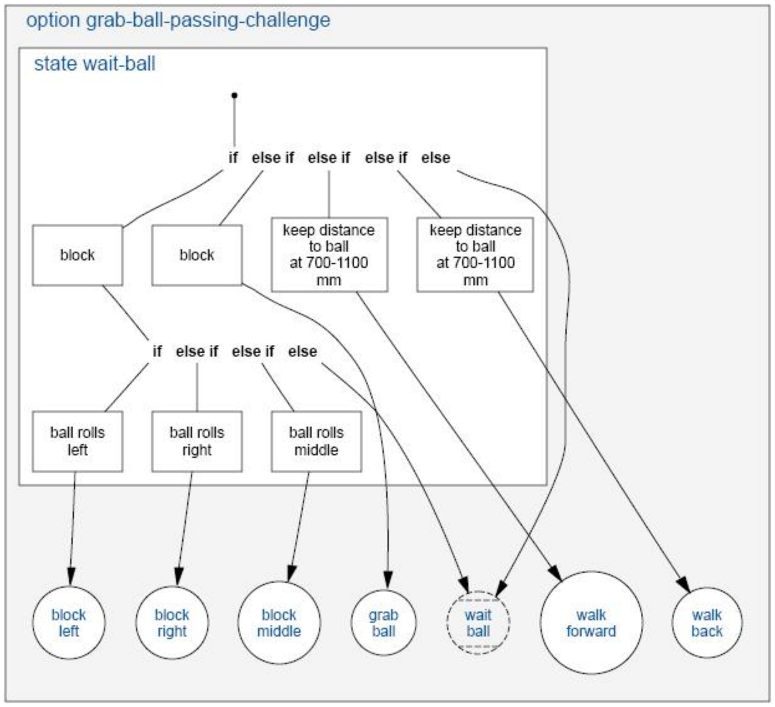


Figure 3: implementation of the’ wait-ball’ state machine in the no communication approach

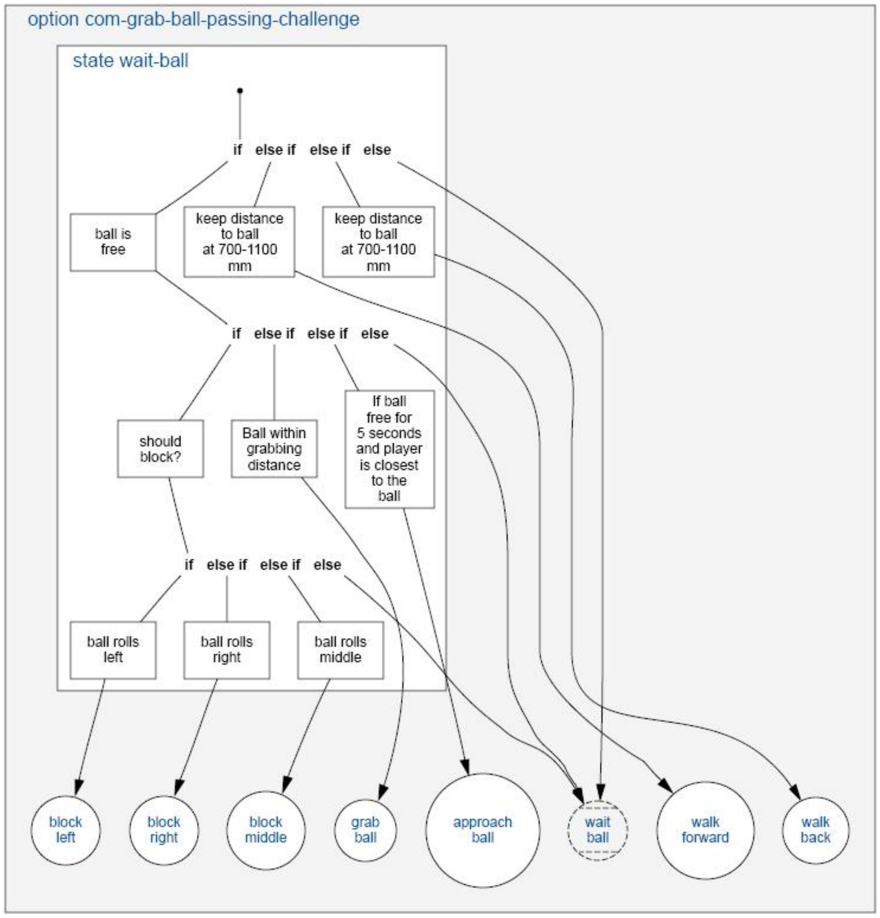


Figure 3: implementation of the' wait-ball' state machine in the message-based approach

The whole ball-passing behaviour can be seen in Appendix B.

4.1 Behaviours in free play

After we implemented the two passing ball approaches we introduced some additional behaviour to improve the overall ball passing in free play. Free play is one of the experiments we will perform and in it we will let three robots pass the ball to each other. The robots will be using the message-based approach.

At the start of the free play one of the robots has to approach the ball. In our implementation the robots will only approach the ball if it is with a certain distance range. To avoid this unnecessary waiting for the ball we have added the behaviour that the closest robot to the ball will approach the ball. So we have implemented a function that calculates which robot is closest to the ball. Every robot that sees the ball estimates its relative distance to the ball and sends that out to the other teammates. Based on the robot's own ball localization and the message it receives from other teammates, the robot can determine if it is the closest to the ball or not. So every robot knows its distance to the ball and knows what other robot think there distance to the ball is. The robot closest to the ball is the only one that will approach the ball.

One of the added behaviours is grabbing the ball and keeping it close to the robot's chest before it performs a kick and after it locates its teammate. It happens regularly that the ball rolls away after the robot has turned to face one of the teammates. By letting the robot grab the ball again before performing the kick ensures that the robot will kick the ball and not just thin air. This allows the robot to perform the kick correctly.

Further more, we added the behaviour that the robots keep at a certain distance from the ball if it is possessed. This is done to avoid the situation where the robot grabbing the ball cannot perform a kick because it doesn’t detect other team players. The robots that do not possess the ball will try to stay at a distance range of 70 cm to 110 cm. The maximum threshold of 110 cm is determined by the distance that the playerpercept and colour collaboration can still perceive and recognize other teammates. By determining the minimum threshold of the distance range, it was important to choose a distance not too close of to the maximum threshold. The distance of 70 cm is chosen roughly but it allowed us to precisely observe the behaviours of the robots.

Also another behaviour is added to avoid situations where no robot goes for the ball and the robot that claims to possess the ball can’t see the ball. In this case no robot will go for the ball even if the other robots can see the ball and one of the other teammates is actually the closest to the ball. To prevent such a situation we added a constraint that if the robot that possesses the ball looses sight of it and cannot find it for 5 seconds the ball will be released again and the closest robot is allowed to go for the ball.
In the case that the ball is possessed but one or more robots cannot see the ball, the robots wait for 3 seconds before looking for the ball. This is done because the robot possessing the ball may turn around the ball to face one of the teammates. As the robot turns it may occlude the ball with its body preventing a teammate from seeing the ball. The 3-second wait is added to avoid unnecessary search for the ball.

5 Experiments

Some experiments have been performed to test the performance of the ball passing. In these experiments we wanted to get a clear idea about the behaviour of the robots when passing the ball to one another. We started with a number of established constraints**:**

- *Ball passing independent*: ruling out kick inaccuracies by letting a human pass the ball
- *Localization independent*: eliminating localization problems by giving the robots fixed locations
- *Environment independent*: reducing effects of light and environment conditions by choosing different location angles

After each performed experiment we removed one or more of the initial constraints to scale it up to the Challenge level.

5.1 Kick angle accuracy

One of the experiments we performed to tune the passing behaviour is finding a good kicking angle for our chosen kick. The experiment setup consists of placing two robots in the field, facing each other. One of the robots is placed on the field with its paws spread out sideways. This is actually one of the blocks a robot can perform and it is called ‘middle block’. The other robot is given the ball to kick it to the other robot. By using a different angle each time and recording the number of correct kicks we are able to determine the angle in which the kicks performed were the most accurate. The kicks are considered correct if the ball is kicked to the other robot between its paws. For each angle the experiment was performed ten times. The results of this experiment can be seen below:

Turn angle in degrees	Number of correct kicks	Number of Misses
45	3	7

55	4	8
60	5	5
65	3	7

We have chosen the 60-degree angle turn before performing the kick, due to its 50% accuracy. It produces a higher accuracy percentage then the other angles. Performing more accurate kicks will defiantly help improve the ball passing between the robots.

5.2 Manual ball passing

This experiment is performed to determine if a robot correctly detects when the ball is coming towards it. We will measure that by the number of robots that approach the ball at the same time to block or grab it. So in this experiment we placed two robots standing next to each other with a distance of 53 cm between them. Both robots facing the direction, in which the ball will be kicked, see figure 5. The ball is rolled manually towards one of the two robots. We switch the robots position each time. The robots are also placed in different angles to determine if the light in the field affected the ball passing play. This experiment was performed three times for each approach. With communication, only one robot approached the ball each time. So it is obvious that in this case the confusion about to which robot the ball is heading is very limited, only 3,3%. With on message passing, we notice that the level of confusion is higher. The number of times two robots approached the ball at the same time is about 40% of the time.

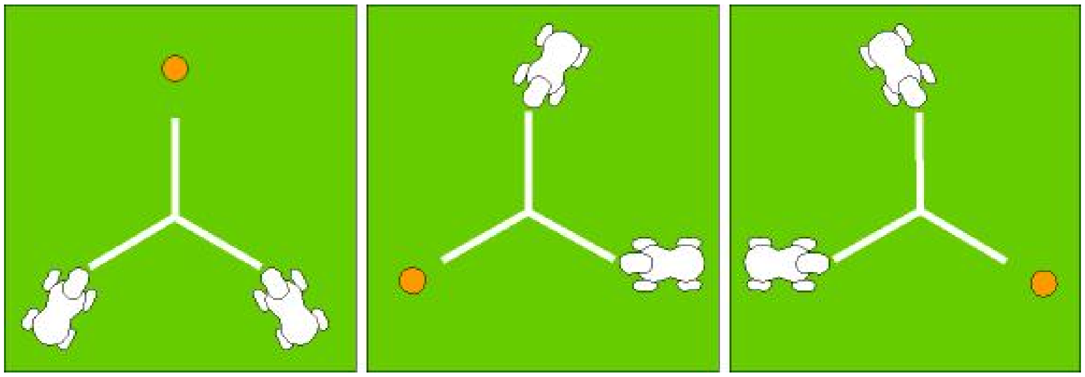


Figure 5: setup of the manual ball passing experiment

5.3 Robot ball passing

In this next experiment we performed exactly the same experiment as the previous one, except this time the ball is kicked by a robot, see figure 6. We wanted to observe the effect of the inaccuracy of kicks on catching the ball and the confusion that occurs sometimes when the ball is kicked between the two robots. Passing the ball with the no-communication approach results in 30% confusion that happens when two robots approach the ball at the same time. The results produced are actually better then the results of the previous experiment with 10%. Communicating the ball status in this experiment shows again that the confusion remains constant at 3.3%.

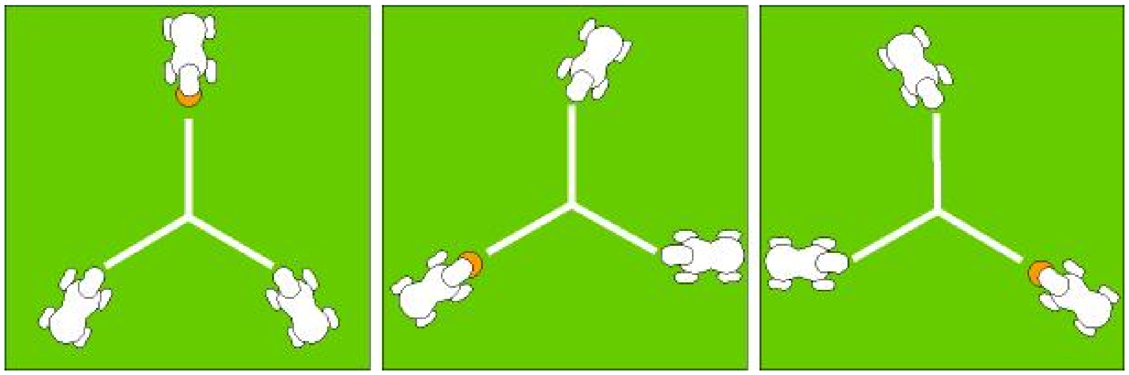


Figure 6: setup of the robot ball passing experiment

5.4 Free Ball Passing

The last experiment we performed is letting three Aibo robots pass the ball to each other with only the constraint that the robots stay within an enclosed area. The only interference with the ball passing occurs to limit the playing space of the robots. This experiment is performed using the communication approach of the ball passing. The robots passed the ball successfully to each other with no or little confusion about who will grab the ball and perform the kick. The robots not possessing the ball do not grab the ball even if the ball gets really close to it. They also nicely stay at the set distance range to the ball, which led them to show an emerging behaviour of forming a triangle each time [6].

6 Future work and discussions

We were able to implement collaborative ball passing behaviour for the Aibo robots. The experiments performed prove the joint actions are successfully implemented. However there are still some improvements that can be done. One of these improvements would be performing a full ball-block without pushing the ball away. Another issue that will help improve ball passing is having a better localization of the ball and the other teammates on the field. Furthermore, the kick performed in passing can be extended to not only one type of kick.

7 References

[1] RoboCup Technical Committee: 'Sony Four Legged Football League Rule Book' (2006 rules, as of November 8, 2005), http://www.tzi.de/4legged/pub/Website/Downloads/Rules2006.pdf

[2] J. Sturm, A. Visser and N. Wijngaards: 'Dutch Aibo Team: Technical Report RoboCup 2005', October 2005, http://www.dutchaiboteam.nl/research/publications/DAT2005TechReport.pdf

[3] Dutch Aibo team http://www.dutchaiboteam.nl/

[4] svn://info.science.uva.nl/scratch/svn/DAT/trunk/DT2005/

[5] Thomas Röfer et al.: 'German Team - RoboCup 2004', http://www.germanteam.org/GT2004.pdf

[6] I.Esteban, M.Greef,de, D. Soest, van, A. Mahdi, Cooperative Robots: Expectation and Message Driven Behaviour January 2006

Appendix A

A.1 Changes in C++ Code

The table below contains the files we have modified and the new functions that we added to those files. The table also includes a description of the new functions or the code lines that are added inside existing functions.

File	Function	Description
BallSymbols.cpp BallSymbols.h	getBallIsTaken	A Boolean function that returns true if the ball is possessed by a robot
	getLastTimeBallBecameFree	A function that returns the last time since the ball became free
	Update	A function that updates variables, we added the 'getLastTimeBallBecameFree'
Cognition.cpp Cognition.h	PlayersPercept	Looks for other players used in StrategySymbols, this is added to the BehaviourControlInterface
GT2004StragegySymbols.cpp GT2004StragegySymbols.h	getPlayerIsClosestToBall	A function that returns true if the robot is the closest to ball
	getDegreeToFriend	Retrieves the turn angle to another player
BehaviourControl.h	PlayersPercept	Looks for other players
BallModel.cpp	ballJustSeen	A Boolean that is added in the SeenBallPosition function
GT2004BallLocator.cpp	DetermineNumberOfImages With_withoutBall	We added the line 'BallModel.seen.ballJustSeen =.ballPercept.ballWasSeen' to determine which player is closest to ball

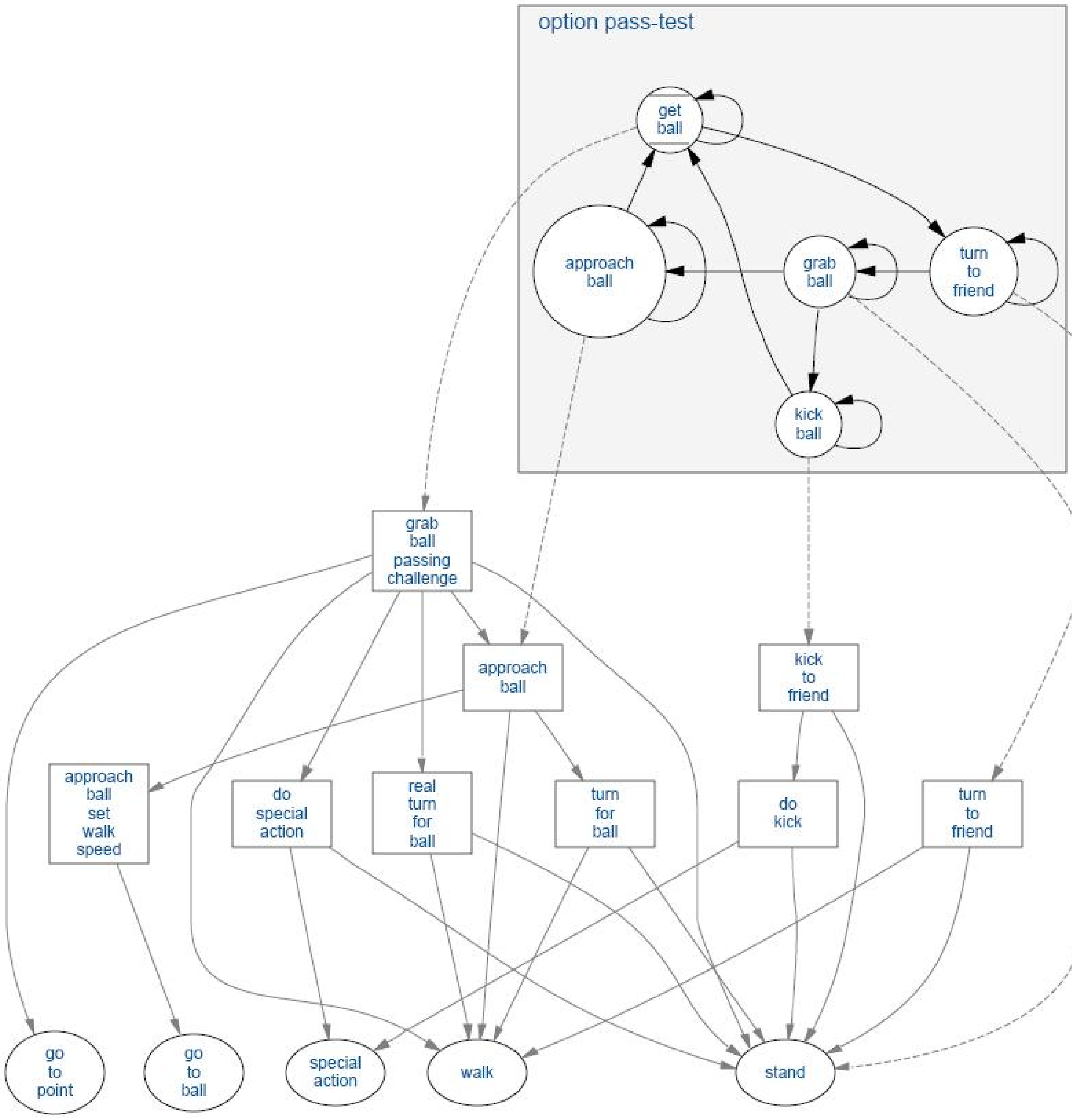
A.2 Changes in the XABSL Code

Here follows a table containing the new behaviours that we have added to the XML files and the description of each of those new behaviours.

File	Behaviour	Description
Options.xml and Agent.xml	Wait-for-ball	Wait till the ball is the closest to this player
	Real-turn-for-ball	Turn to ball
	Kick-to-friend	Kick the ball to friend
	Pass-test	A test for the ball passing challenge
	Grab-ball-passing-challenge	Catching the ball for the ball passing challenge
	Turn-to-friend	Turning towards a teammate to perform a pass, using the degree-to-friend
	Turn-with-ball	Turn around the ball without moving the ball.
	Com-grab-ball-passing-challenge	The catching of the ball passing challenge with communication
	Com-kick-to-friend	Kick ball to friend with communication
	Com-pass-test	Test for the pass with communication
	play-ball-passing	Selects game state according to game manager messages and buttons
Ball-symbols.xml	Ball-is-taken	The current state of the ball
	Ball.get-time-since-ball-became-free	Time in seconds
Strategy-Symbols.xml	Player-is-closest-to-ball	Gets the player that is closest to the ball
	Degree-to-friend	Gets the degree of the turning angle to face friend/teammate
	Team-message	Added team-message.preparing-a-kick and team-message.just-performed-a-kick

Appendix B

Here follows the general ball passing behaviour graph. The difference in implementation of the two approaches: the no communication approach and the message driven approach occurs in the ‘grab-ball-passing-challenge’ and the ‘kick-to-friend’. In the second approach these behaviours also include passing messages to other teammates about the ball status.


option pass-test
get ball
approach ball
grab ball
turn to friend
kick ball
grab ball passing challenge
approach ball
kick to friend
approach ball set walk speed
do special action
real turn for ball
turn for ball
do kick
turn to friend
go to point
go to ball
special action
walk
stand