Human – Robot Interaction

for RoboCup MSL

Tech United Eindhoven – Wiki report

Control Systems Technology
Department of Mechanical Engineering
Eindhoven University of Technology
CST 2014.019

Student: Marjon van ’t Klooster
Studentnummer: 0819200
Cursus part: Premaster final project
Date: 28-01-2014

Supervisor: dr. ir. René v.d. Molengraft
Coaches: Robin Soetens
ir. Rob Hoogendijk
Summary

Since 2014 high level human coaching is allowed through the RoboCup committee. Several rules are attached to this regulation. This is in order to keep the development of the robot soccer in progress, so the goal of the RoboCup committee could eventually be achieved.

“By 2050, a team of fully autonomous humanoid robot soccer players shall win a soccer game, complying with the official FIFA rules, against the winner of the most recent World Cup of Human Soccer.”

The project is set up for developing a system to communicate on a human way with the Turtles on the field during a game. The intention of this system is to coach the Turtles and to make strategic moves, when a refbox task should be performed through Tech United Eindhoven.

In the first chapter the background information about Tech United Eindhoven and the mission with the goals are described. In the following chapter there has been studied on two possible coach moments. The system is divided in parts with the information of the coach moments and each part is analyzed and solutions are conceived. These solutions are converted to code and implemented in the system. The code is clarified in chapter 4.

Some experiments are done with the coach software. The maximum possible distance between the coach and Turtle is recorded. Further, there has been tested how often the right QR code was scanned. When the QR code was properly scanned the strategy was performed in the simulator to verify whether it was correct performed.

After the experiments some recommendations are suggested. These recommendations have been done on short terms and long terms. The short terms are before the next competition. The long term recommendations are for the development of the coach software.
## Contents

Summary .......................................................................................................................... 2

1. Introduction .................................................................................................................. 4
   1.1 Tech United Eindhoven ......................................................................................... 4
   1.2 Mission .................................................................................................................. 6

2. Background .................................................................................................................. 7
   2.1 Related work ......................................................................................................... 7
   2.2 Rules and Regulations of Middle Size Robot League ........................................... 10

3. Analysis ...................................................................................................................... 11
   3.1 Application .......................................................................................................... 11
   3.2 Communication contents .................................................................................... 14
   3.3 Communication method ....................................................................................... 17
   3.4 Moment of coaching ............................................................................................ 18
   3.5 Concept description ............................................................................................. 19

4. Implementation .......................................................................................................... 20
   4.1 Moment of coaching ............................................................................................ 20
   4.2 Communication method ....................................................................................... 22
   4.3 Communication contents .................................................................................... 25

5. Experiments ............................................................................................................... 26
   5.1 Maximum QR scan distance .............................................................................. 26
   5.2 Possible coach moments ..................................................................................... 26

6. Results ....................................................................................................................... 27
   6.1 Maximum QR scan distance .............................................................................. 27
   6.2 Possible coach moments ..................................................................................... 29

7. Conclusion .................................................................................................................. 32

8. Recommendation ....................................................................................................... 33

References ..................................................................................................................... 34

Appendix ....................................................................................................................... 35
1. Introduction

This document is written for Tech United Eindhoven. The education that it concerns is Mechanical Engineering of the department Mechanical Engineering at Eindhoven University of Technology.

This project is for completing the Pre-Master year of Mechanical Engineering. The period of the project will be 252 hours which is equal to 9 credit points. The project is started on 4 September 2013. The project comes from the Robotics field and is a development project for coaching the Turtles during a game.

I would like to thank Tech United Eindhoven for the opportunity to perform the Premaster final project. Especially I thank Robin Soetens for supervising.

1.1 Tech United Eindhoven

This paragraph contains a description of Tech United. The first paragraph is about the organization. The last two subparagraphs contain contact information and the appointments.

1.1.1 Organization

General
Robocup is established in 1993. The first Dutch team was officially founded in 2000 under the name Clockworks Orange and consisted of a team of several Universities of the Netherlands. When Clockworks Orange stopped, Tech United continued with the development of the soccer team. Tech United Eindhoven was founded in 2006 as a restart and is the result of Tech United which was formerly formed by the Universities of Delft and Eindhoven.\(^1\)

The first world championship was in 1997 in Nagoya. Tech United Eindhoven has participated in 2006 for the first time. Since 2008 they are at the world’s top ranking at the second place. In 2012 they have become world champion.\(^2\)

Structure
Tech United Eindhoven can be divided in three research groups: Soccer, Humanoid and Care. Each group has their own robot, see figure [2.1].

- The Turtle and the goal keeper are members of a team with six players. These robots will be explained in more detail later.
- TULip is made for the Adult size competition. This robot plays one-to-one games by taking penalties and goalkeeping.
- AMIGO is the care robot, which learns from his own events but also of other robots by exchanging information on a platform.\(^3\)

![Figure 2.1](image)

\(^{1}\)Tech United Eindhoven website, http://www.techunited.nl/nl/ontstaan

\(^{2}\)Tech United Eindhoven website, http://www.techunited.nl/nl/resultaten

\(^{3}\)Tech United Eindhoven website, http://www.techunited.nl/nl/amigo/
1.1.2 The soccer team

The soccer team consists of five TURTLEs and one goal keeper, see figure [1a] and [1b]. TURTLE is an abbreviation of Tech United Robocup Team Limited Edition. The team plays in the Middle Size League of the Robocup. The robots have changed a lot over time since the start-up of Tech United. At this moment the focus of development lays on shooting and receiving lob balls, on dealing with scrum moments, self-learning ability and coach software. Furthermore, the 5K TURTLE is developed for starting teams, so they can make the step easier to start into the league.

A detailed description of the TURTLES can be found on the site of Tech United Eindhoven.\(^4\)\(^5\) The most important parts of hardware and software of the TURTLES are described. It is important to have a certain previous knowledge before starting with reading this document.

1.1.3 Contact information

**Organization:**
Name: Tech United Eindhoven office
Address: De wielen 1 (on the TU/e terrain)
5612 Eindhoven
E-mail: techunited@tue.nl
Telephone: +31(0)40 247 22 48
Site: www.techunited.nl

**Coach:**
Name: Robin Soetens
Function: Team leader of the soccer team from Tech United Eindhoven
E-mail: robinsoetens@gmail.com
Telephone: +31(0)6 41 60 64 31

**Student:**
Name: Marjon van ‘t Klooster
Address: Hohorstlaan 32
3833 BC Leusden
E-mail: m.e.v.t.klooster@student.tue.nl
Telephone: +31(0)6 12 73 24 94

1.1.4 Appointments

A meeting will take place once a week between the supervisor and the student. The progress and plan will be discussed. When the student has questions she can reach the supervisor, but also the other team members of Tech United. Further a weekly team meeting is on Wednesday morning from 09.15 till 10.00. In this meeting the process of each of the members will be discussed.

\(^4\) Tech United Eindhoven website, http://www.techunited.nl/nl/turtle
1.2 Mission

1.2.1 Problem definition

The project is about the communication with the Turtles during a game. The main reason for this project is described below.

“Middle-size league RoboCup soccer matches are extremely exciting. Scores are often tight and involvement of crowd and team members is huge [1]. In such a tense environment it could be frustrating that team members are not allowed to pass any information to their robots during a game. A tunable parameter that is slightly off, or a strategic mistake that happens over and over again, could screw up an entire game. In order to meet RoboCup regulations, in-game coaching by simply connecting to the robots through WiFi, Bluetooth, or any other electromagnetic signal will never be allowed. However, for 2014 season the executive committee for the middle-size league will propose a rule change allowing robot coaching in ways that are natural to human beings. Goal of this Pre-Master End Project is to figure out what would be the best way to do so, and to implement this in software.” [2]

The first explanation of the problem definition is about what is meant by ‘ways that are natural to human beings’. The human has many different senses to detect things of his surroundings. They can be described in five main senses, namely olfaction, sound, sight, taste and touch. [6] These sentences can be used to communicate with each other.

1.2.2 Mission

A main question is described on the base of the problem definition. This question is divided in sub goals and questions, in order to control the project better.

How could communication on a human way be possible with the Turtles on the field during a match so strategic decisions can be made?

1.2.3 Product description

Three goals are already set in the project description of Tech United [1]:

1. Experiment with multiple methods of human-robot interaction (e.g. gesture recognition, voice commands, recognition of colored surfaces, symbols or QR code on a whiteboard.)
2. Identify existing tunable parameters/strategic decisions that could benefit from in game coaching.
3. Develop software that allows in-game coaching of Tech United's Middle-Size League robots.

Several sub questions can be asked by the goals:

1. Which methods are there for communicating between robots and human?
2. Which way is possible to communicate with a Turtles and a human?
3. How and when to make the first contact with the Turtles during a game?
4. What has to be communicated with the Turtles?

---

2. Background

This paragraph gives an overview of work from other researchers that are related to this project. The main subject of the research is communication between human and robots on a human way. First the possible ways of communication between humans are put in a row. A research is done to related work on these communications forms between a human and a robot. The background of the found work will be described and then this information will be discussed.

2.1 Related work

As mentioned in the mission description the human has five senses. Olfaction, taste and touch are not convenient to use during a RoboCup game, where the Turtles are not nearby the coach. Therefore the other two senses will be further explored on which way they could be used in the communication between the Turtles and the coach. In the list below some examples are given of applications which use these communication mediums.

Sound
1. Voice recognition: Nowadays we can talk to devices, which understand us and can then execute a command. Like Siri, the personal assistant, on the Apple devices or the voice command control of the Renault cars for navigation or telephone. Sometimes the environment is too noisy, there for a research is done on voice recognition in combination with face recognition in a car. [3]

2. Morse code (example: whistling)

Sight
1. Color code: The CMUcam5, known as the Pixy camera, can search the surrounding for colors and recognize objects with the users preset colors. [7]

2. Bar code; is used for a long time and can be found on almost every product in the supermarket or other shop.

3. Showing numbers: The ‘Traffic Sign Recognition’-system recognize the traffic signs and could be used for limit the speed. [4]

4. Quick response code; is begin increasingly used for example through companies, but also in hospital environment it is applied for example in Meander Medisch Centrum Amersfoort.

5. Morse code (example: light)

6. Gesture: The Kinect camera which is used on the Turtles is official developed for computer games. It is used for recognition of the movement of a person. Currently the Kinect is used for other goals, but also other games are developed. Like a memory game for the humanoid robot. The robot needs to remember the posture needs to copy and remember the posture which the person occurs. [5]

7. Semaphore signals

2.1.1 Advantages and disadvantages

The project can be delineated if a direction is chosen. The direction can be selected by an overview of the advantages and disadvantages.

Sound
- All Turtles have the same information on the same time
- The Turtles can play through without directing to the coach
  - Many other sounds at stadium can cause distortion

Sight
- Better to shield off for opponents
- Already exist several simple programs which can be applied
  - First contact should be transmitted by another method, or automatically at certain moments.

Image encoding is chosen as communication form for the first version of the coach software. In particular Quick Response code encoding, nowadays Quick Response codes could be provided on many different sites with different contents. Also several open source programs are available to use for decoding the Quick Response codes. This is a good first step with a low threshold which is possible to extend.

2.1.2 Nowadays application of the QR code

As mentioned earlier QR codes are used a lot around us. As example for advertising, in restaurant it is used to make an order, providing travel information and more applications are possible.

QR code in Hospital environment
QR code is tested in hospital environment by coding and accessing medical data of the patient. Data like vital signs after measurements of a patient can be registered. Only certified members can upload and use the information. This information is stored in a central database, which will be located on the server in the hospital. Quickly exchange of digital information is now possible in a hospital. [6]

QR codes can also be used in medical bracelets, when the bracelet is scanned the medical staff has immediately an overview of medical data of the patient, such as emergency contact, blood type, allergies, etc. [8]

Tracking a person with QR code
Tracking a human through a robot can be needed, for example by moving heavy equipment, like luggage on an airport or medical equipment in a hospital [7]. A tracking system is developed by Okinawa National College of Technology and The Institute of Scientific and Industrial Research, Osaka University which detect the location of a person through a QR code. The tracking system is not yet consistently, when the QR code is lost in the field of view the rediscovering of the QR code is difficult. [8]

A company called Demcon, has created a program that scans the area for the interested object. This can be an improvement on the tracking system described above.

---

2.1.3 Connection between previous work and this project

The main goal of this project is to communicate with the Turtles in the field. QR code is chosen as the communication form between the coach and the Turtles. During a coach situation a Turtle will scan the area where the QR code possible will be. When it is found, the Turtle will have to move to the QR code until it can scan properly. The program of Demcon can be used for the focus on the QR code in a certain area and the tracking program can be used for the movement to the QR code.

One of the Turtles will scan the QR code when it has found in the area and is close enough to scan. The QR code contains information that makes a decision which preset tactic will be performed. This is similar with the preset information of the medical bracelet.

The one Turtle that has scanned the QR code will communicate the information with the other Turtles through a central system. It is comparable to the database of the hospital, where the information is saved. Other Turtles can use the information out of the database. The Turtles can only be coached with the right QR codes. Tech United can communicate with the Turtles by using preset tactics. Like the certificated medical staff of the hospital, only they can use the information of the central database.
2.2 Rules and Regulations of Middle Size Robot League

The Rules and Regulations for 2014 are recently published. [9] Coaching is now included in the regulations for upcoming matches. The rules for coaching can be found in the table below.

Table [3.1] The Rules and Regulations about the high level human coaching

<table>
<thead>
<tr>
<th>RC-Decision 2.1:</th>
<th>High level human coaching is allowed. For that purpose each team may use a set of paper boards with QR Codes to be directly interpreted by the robots. High level coaching has to comply the following rules:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Boards can not be larger than 30 x 30 cm;</td>
</tr>
<tr>
<td></td>
<td>• Only robots that are in the field can be coached;</td>
</tr>
<tr>
<td></td>
<td>• Each team will designate a member to perform coaching;</td>
</tr>
<tr>
<td></td>
<td>• Coaching is only allowed from the team leader position, in front of the teams’ base station pc;</td>
</tr>
<tr>
<td></td>
<td>• The robot(s) that is/are being coached should remain within the field of play, and cannot be touched by human team members;</td>
</tr>
<tr>
<td></td>
<td>• The human that is coaching should stay always outside of the field of play;</td>
</tr>
<tr>
<td></td>
<td>• No electronic device, other than electronic devices that are mounted on the robot itself, can be used to transfer coaching instructions to the robot;</td>
</tr>
<tr>
<td></td>
<td>• Coaching can take place only during ‘dead time’ (i.e., the 10 seconds between a stop and a start by the assistant referee).</td>
</tr>
<tr>
<td>QR Codes can only be of the version 1 or 2. Any level of Error Correction can be used and any type of data can be encoded. In the figure below a summary of version 1 and 2 of QR Codes can be seen. Use of other higher versions of the QR Codes as well as Micro QR Code, iQr Code, SQR Code or LogoQ is strictly forbidden.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version</th>
<th>Modules (mixed)</th>
<th>ECC Level</th>
<th>Data bites (numeric)</th>
<th>Alphanumeric</th>
<th>Binary</th>
<th>Kanji</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>21x21</td>
<td>L</td>
<td>152</td>
<td>41</td>
<td>25</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M</td>
<td>128</td>
<td>34</td>
<td>20</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q</td>
<td>104</td>
<td>27</td>
<td>16</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H</td>
<td>72</td>
<td>17</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>25x25</td>
<td>L</td>
<td>272</td>
<td>77</td>
<td>47</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M</td>
<td>224</td>
<td>63</td>
<td>38</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q</td>
<td>176</td>
<td>48</td>
<td>29</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H</td>
<td>128</td>
<td>34</td>
<td>20</td>
<td>14</td>
</tr>
</tbody>
</table>

NOTE: Complementary information regarding high level coaching may be released until February 28, 2014.

Extra rules might be added regarding constant coaching during a match. Constant coaching might be possible by making use of a microphone and/or a webcam. It is important that these communication mediums have a delay. The delay has to prevent that the coach software will be used as a remote control.

The other rules in the table are also to prevent that the coach software will be used as a remote control. For example “The robot(s) that is/are being coached should remain within the field of play, and cannot be touched by human team members.” Otherwise the coach could use a robot out of the field which is not taking part of the game and communicate with the team through this robot. The intention of the coach software is that it only will be used for high-level commands, like to communicate certain strategies.
3. Analysis

As mentioned in the project description it would be easy to communicate with the Turtles, so tunable parameters can be changed and strategies, which go wrong, can be avoided. In this chapter the existing system will be analyzed on which strategies and parameters should be changed and where the coach software can be implemented. First, two scenarios are given where the coach system could be used.

3.1 Application

In the problem definition, two situations where coaching is preferred are described. Two other possible scenarios are outlined below, in which the coach software is applied. On the basis of these scenarios the coach system can be divided into smaller parts. These parts can be analyzed and different solutions can be conceived.

*Scenario 1:*

*Step 1:* The last minutes of the match are entered and the team is behind with the score. The referee blows his whistle for a free kick for Tech United.

*Step 2:* One of the Turtles moves to the sideline where the coach stands. On this moment the coach gives the Turtles the mission for replay as less as possible and redials distance as large as possible during a pass. This is for saving time and to score as soon as possible.

*Step 3:* Turtle 2 pass the ball to Turtle 4. Turtle 4 turns around and shoot at the goal. The Turtles perform the refbox task and scores!
**Scenario 2:**

**Step 1:** Of course, it can also be the other way around. Tech United has one point more than the opponent and the last minutes have started. The referee blows his whistle for a free kick for Tech United.

**Step 2:** It is now important to stretch time. This can be achieved just by replay more between the Turtles. The coach communicates with one of the Turtles during the game. He gives the Turtle the mission to replay more in many small passes.

**Step 3:** Turtle 2 pass the ball to Turtle 4. Turtle 4 pass the ball back to Turtle 2. Turtle 2 goes around Turtle 4 and dribbles the ball towards the goal. Turtle 2 shoots at the goal, when he is near the goal. The possible result: Tech United wins the match!
A constant line can be found from the above scenarios namely; On a certain moment the coach wants to change the strategy of the Turtles through human communication. These data can be divided in three parts:

- The moment of coaching: The question is when contact is needed with the Turtles for communicating the strategy, but at first the strategy should be known.
- Communication method: A certain communication medium is needed, when the coach wants to communicate.
- Communication contents: In the example a single pass or a double pass is used and the long or short pass distance is changed. The first falls under strategy, the other two falls under tunable parameters.

The tunable parameters and the strategies will be analyzed first, with the possible solutions and where it should be implanted in the current system. If these are clear the moment of coaching will be described in subparagraph 3.4.
3.2 Communication contents

In this paragraph the communication contents is described. This refers to the message which will be communicated between the coach and the Turtle. In the first subparagraph a number of possible tunable parameters are explained which could be changed during the match. In the next subparagraph the strategies are described which can be coached during a match. The focus is especially on the strategies in this report.

3.2.1 Tunable parameters

The turtles have a number of parameters that can be changed. These parameters must be properly set before a game, because during a match it is not possible to change the settings. It would be possible to change the parameters during the game, when the coach software is included in the system.

In the table below the parameters which would be useful to be modified are listed with a description of the parameter and an explanation why it is useful to change the particular parameter.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GF Kicking Region</td>
<td>Kicking region: The line from where the ball will be shot at the goal.</td>
<td>The kicking region line should be moved, when the distance is too large to shoot the ball and score from the line without interception from the opponent.</td>
</tr>
<tr>
<td>PASSTO-OPPENTHALF</td>
<td>Pass to opponent half: The long distance shooting, so the ball will go to the opponent’s half as fast as possible.</td>
<td>When the long distance passes are easily intercepted, it might be better to make short passes between the Turtles and turn of the long distance parameter.</td>
</tr>
<tr>
<td>BA</td>
<td>Backwards attack: Driving backwards with the ball near the goal.</td>
<td>If a scrum situation occurs often through moving backwards, than this parameter can better be turned off.</td>
</tr>
<tr>
<td>SF_v_max_move</td>
<td>Velocity movement: The velocity of driving without the ball</td>
<td>The velocity range can be set higher or lower, when the Turtles can’t keep up with the opponent or when the Turtles can’t accurately pass because of the speed.</td>
</tr>
<tr>
<td>SF_v_max_dribble</td>
<td>Velocity dribbling: The velocity of driving with the ball</td>
<td>The description above of the velocity movement also belongs here.</td>
</tr>
<tr>
<td>SF_v_max_aim</td>
<td>Velocity near aim: The velocity of driving near the aim</td>
<td>The description above of the velocity movement also belongs here.</td>
</tr>
<tr>
<td>SF_a_max_move</td>
<td>Acceleration movement: The acceleration of driving without the ball</td>
<td>The acceleration value should be set lower, when the Turtle spins often because of the high acceleration. The acceleration should be set higher when the maximum velocity value not can be reached.</td>
</tr>
<tr>
<td>SF_a_max_dribble</td>
<td>Acceleration dribbling: The acceleration of driving with the ball</td>
<td>The description above of the acceleration movement also belongs here.</td>
</tr>
<tr>
<td>SF_a_max_aim</td>
<td>Acceleration near aim: The acceleration of driving near the aim</td>
<td>The description above of the acceleration movement also belongs here.</td>
</tr>
</tbody>
</table>
3.2.2 Strategy

A referee supervises the RoboCup Middle Size League, this is comparable with the human soccer. The same kinds of rules are applied during the game. A Kickoff, Freekick, Throw in, Corner, Penalty and Goalkick are also known in the RoboCup games. These decisions of the referee are called refbox tasks. These refbox tasks can be performed with different tactics. The decision of the referee, with a so-called death game moment, shall be communicated to the turtles through a central system, the refbox. Who should take, for example, the Freekick is hereby also declared.

Tech United Eindhoven describes the active and passive refbox tasks. When the ball is for the opponent, Tech United Eindhoven uses the passive refbox task and takes the defensive position. The active refbox task is used, when Tech United takes the ball out. There are several tactics possible in the active refbox task. This is a decision that could be coached.

In the next subparagraph an example is given of an active refbox task, the kickoff.

**Explanation of an Active refbox task**

Depending on which refbox task is chosen by the referee, a case in a switch is selected in the program of the turtle. As first a kickoff is taken with a game, therefore this one will be explained.

```plaintext
case COMM_KICKOFF_ACTIVE:
    set_targetpoint(FIELDWIDTH*0.5, FIELDLENGTH*0.02, pS_RefP);
    angle = 0.5*M_PI+getangle(pS_RefP->Target_point,Ball_Pos);

Explanation: The point where the ball should go is calculated using the dimensions of the field.
```

```plaintext
if (psfgd->RefboxStruct.StartBallPosition_xy[0]>=0)
    set_startpos(0.0, 0.75, y_dist_startpos_role_2*sin(angle), y_dist_startpos_role_2*cos(angle), pS_RefP);

Explanation: A statement is described between two parentheses, behind the 'if' word. This statement holds, in this case, that when the ball is located above the horizontal centerline, the following part will be performed. This part describes the start position of the Turtle when the kickoff will be taken by Tech United Eindhoven.
```

```plaintext
else
    set_startpos(-0.0, 0.75, -y_dist_startpos_role_2*sin(angle), y_dist_startpos_role_2*cos(angle), pS_RefP);
    pS_RefP->obstacle_type = obstacle_none;

Explanation: This describes the start position of the Turtle when the kickoff will be taken in another part of the field, in this case beneath the horizontal centerline. The field is divided into standard parts. The distribution of the field can be found in appendix [A].
```
N_variants = 4;
    set_percentages(percentages_cum,N_variants,0,50,50,0);
    set_pass_action_variant(rndr,percentages_cum, pS_RefP, S, N_variants,
    pass_double+dribble+ BA,pass_single+dribble, pass_single+dribble+ BA,pass_single+kick);

Explanation: As mentioned earlier, there are a number of tactics which can be chosen. There are six
possibilities to perform the several refbox tasks:

- pass_single+kick
- pass_single+dribble
- pass_single+dribble+BA
- pass_double+kick
- pass_double+dribble
- pass_double+dribble+BA

pass_double: Two times of passing the ball before shooting on the goal.
pass_single: Only play once the ball to another turtle before shooting on the goal.
kick: Shooting on the goal
dribble: Moving with the ball, with interruption of holding the ball.
BA: Stands for backwards attack, driving backwards with the ball before shooting.

The choice which tactic will be performed is based on a percentage of how many times a tactic is
executed. In the example above, four tactics can be chosen. The most likely is that pass_single+dribble
or pass_single+dribble+BA will be chosen, this because the percentage is set to 0,50,50,0.

pS_out->pSTB->RefboxTaskExtended = TaskRefboxKickoff;
pS_out->pSTB->RefboxTaskExtended += 1;
break;

Explanation: All of the tactics has a combination of numbers. The sum of these numbers is a unique
number, which will be set to the output. This unique number is known in the rest of the system.

Tactics

Several pass opportunities are possible during the kickoff. The ball can be passed one single time to
another Turtle and then be shot to the goal. Another example is more replays before shooting on goal.
The pass opportunities which can be chosen in the code for the kickoff are the following:

COMM_KICKOFF_ACTIVE
    Variants:
    pass_single+kick 0
    pass_single+dribble 50
    pass_single+dribble+BA 50
    pass_double+dribble+BA 0

If the team is behind during the match in the final minutes, it would be nice when the team can focus on
quick score by overplay as little as possible. Or when the team is well ahead, it could be practical to buy
time during the match by more overplay. This could be controlled by changing the ratios during the
game. In chapter 4 the implantation of the different ratios in the refbox handler code can be found.
3.3 Communication method

In paragraph [2.1.1] is chosen for QR code as communication medium between the coach and the Turtles. Some rules are also attached to the type of QR codes through the MSL committee. In this paragraph a piece of background information.

For a long time barcodes are used in stores for scanning information of the item. The barcode has a limit on characters, because it contains information in one direction. Therefore the quick response code is developed. This form of code contains two directions information and can therefore provide more information. ⁹

Quick response code known as QR code is released in 1994 and is developed by DENSO WAVE for the auto industry. Nowadays the QR code is used for different purposes and can easily be generated by everyone on several websites.

3.3.1 Structure of a QR code

There are several versions of the QR Code. The first version contains 21 modules x 21 modules, the versions increases with steps of 4 modules on each side. The largest version, version 40, contains 177 modules x 177 modules. [10]

**Position Detection Patterns**

Every QR code contains three position detection pattern with a size of 7 modules x 7 modules. These are located in the corners of the QR code. These detection patterns indicate the location and orientation of the QR code in the field of view.

**Timing Pattern**

The timing pattern shows the density and the version number of the QR code. It is also used for the coordination of the modules. The timing pattern starts and ends always with a black module, between these two modules white and black modules are alternated.

**Alignment Patterns**

An alignment block is added from version 2. These are required for the definition of the coordinates by larger QR codes.

**Encoding region**

This region contains the format information, version information and the data. The data contains characters and an error correction. The error correction is added to prevent loss of data caused by damage.

---

3.4  Moment of coaching

It is now known what to communicate with the Turtles, but the question is on which moment communication can be done. It is good to look at the possible moments for coaching, even though this is already determined in the regulations.

With human soccer the coach gives advice from the sideline during the game, but also during time outs strategies are discussed. This is possible because the coach communicates through sound and gestures. During a game the coach uses sound to make contact with the captain of the team and then use mainly gesture for communicating the advice.

There is a limitation on the moment of coaching with the Turtles, because only vision will be used in the first version of the coach software. The result is that the first contact can’t be made via sound, so it must therefore be predefined when there is contact. The following predefined moments are probably possible to communicate during a soccer game with the Turtles.

1. During a stationary match moment, before a refbox task is taken.
2. When the ball is over the center line by the opponent during the game.
3. During the entire game.

Which one of the predefined moments will be chosen depends on which camera will be used for scanning. When the already available Kinect camera of the Turtle is used, then option number 1 and 2 are only possible. This is because during the game the Turtles should be focused on the ball.

Option 1: There is time to have contact with the coach, when the turtles will be ready for the refbox task. At least two Turtles must be ready for playing the refbox task, the other Turtles could have contact with the coach.

Option 2: In case of option 2 the goalkeeper could have contact with the coach when the ball is far away enough, like over the center line at the field of the opponent. The goalkeeper doesn’t have to lay off much distance, so he can stand still for a moment to have contact with the coach.

Option 3: The coach could be scanned the entire game, when a second camera is installed. The camera should always be directed to the coach. Only high demands are asked on the camera, because the camera must be able to scan a code at a long distance and probably also while it is in motion.

Option 1 is chosen for the first version of the coach software, since only the strategy for a refbox task and tunable parameters will be affected. Recently it is also included in the regulations that only contact between the coach and the Turtles may take place during a stationary match moment. [9] See also paragraph [3.2] for the rules about the high level human coaching.
3.5 Concept description

A good description of the system can be made now the solutions of the various parts are conceived. The description will be described on the basis of a situation of paragraph [3.1]. This situation will now be described in detail with the coach system, so it is clear what needs to be implemented in the current system.

The last minutes of the match are entered and Tech United has one point more than the opponent. It is now important to stretch time. The referee gives Tech United a free kick on the opponent’s field in the corner region. The Turtles get a role number assigned, immediately when the free kick is given through the refbox cabinet. The Turtle with Role 3 comes directly to the sideline at the level of the teams’ base station. The coach is standing in front of the base station with certain QR code.

The coach has three strategy choices which the Turtles can perform at this place in the field. The strategy which cost the most time will be performed to stretch time.

- pass_single+kick
- pass_single+dribble
- pass_single+dribble+BA

In this case it is a single pass with a dribble and a backwards attack. The coach holds this particular QR code for the camera of the Turtle. The Turtle will scan the QR code and communicate the tactic with the other team members in the field. The Turtles can select the right tactic in the program.

The Turtle goes to his position in the field, when the QR code is scanned. When the Turtle is too late with the scan of the QR and the referee has started the game, in that case the Turtles will play a random strategy of the list.
4. Implementation

The implementation of the coach software is discussed in the following chapter. In the first paragraph the moment of coaching is described. In the second paragraph the communication medium is explained, which QR scan program is integrated and how it is communicated among the Turtles. In the last paragraph the message which will be communicated from the coach to the Turtle is clarified.

4.1 Moment of coaching

Not only the moment of coaching is important, but also the place. In the regulations is written that the coach only may communicate when it is on the team leader position. This prescribed position is in front of the teams’ base station pc. In the subparagraphs the moment en place will be described.

4.1.1 Triggering

First contact is made with the Turtles when there is a stationer match moment, this because only the strategy for a refbox task will be affected. These strategies don’t have to be changed during the game, but only until a refbox task will be performed. The coach software is therefore triggered at certain prescribed events;

- When an active refbox task is given.  pTRCB->refboxTask == COMM_KICKOFF_ACTIVE
- When there are at least three Turtles present.  pSTB->refbox_role == ROLE_REFBOX_3
- When the ball is on the opponent half of the field.

The base rate of the motion scheme runs too fast for the QR code scanner\textsuperscript{10}. When the coach software is triggered the scanner will be started on a lower sample rate, this is done with the help of a Task Block. The Task Block is made through Tech United Eindhoven to create a subtask which runs not at the base rate and can be triggered by three different kinds of signals.

- It can be clock triggered, thus on the base of time.
- Event triggered, when it meets certain requirements it gives a signal.
- Or self-triggered, when the task is over it start itself again.

The QR code scanner has to be put in an enabled subsystem and will be enabled by the Task Block. All inputs of the subtask have to be led through the Task Block in order to set all date on the same sample rate. In figure [4.1] the Simulink model of this particular part is shown. The QR code scanner part can be found in Coach_Trigger.c.

It is important to know that every task block should be unique. This means that every block has a unique number. Otherwise the task blocks get the same triggering and the same sample rate. This can be checked by editing the mask of the block. Click the right mouse button, go to Edit Mask, select the tab of Initialization and change the value of the period in a unique number.

More information about the task block can be found on the Tech United Eindhoven wikipage.\textsuperscript{11}

\textsuperscript{10} More information about the motion scheme can be found in Appendix [C].

\textsuperscript{11} More information of the Task Block on http://www.techunited.nl/wiki/index.php?title=MultiTasking_Target_for_Linux
4.1.2 Place

The coach may only communicate from front of the teams’ base station. The Turtle should focus on this place, so the coach can communicate the strategy via QR code. In the first version of the coach software the Turtle with Role 3 moves to the coach on the sideline. When the correct QR code is scanned or the refbox task is started the Turtle with Role 3 goes to the right position in the field.

```c
int Coach;
{
    Target_ROLE_REFBOX_3[0] = -FIELDLENGTH/3.0;
    Target_ROLE_REFBOX_3[1] = -6.0;
    Target_ROLE_REFBOX_3[2] = 1.5;
}
else{
    if (psfgd->RefboxStruct.StartBallPosition_xy[1]>0.1667*FIELDLENGTH){
        // find most forward opponent on own half
        for(i=0;i<MAX_OPPONENTS;++i){
/* The rest of this positioning part is left out, because it does not provide more information for this section*/
    }
}
```

Explanation: When a refbox task is chosen through the referee, the Turtle with Role 3 will move to the sideline. Role 3 is chosen to make sure that there are enough Turtles which can perform the refbox task. The Turtles with Role 1 and 2 can already in their positions, while Turtle with Role 3 is going to scan the QR code. The Turtle stays on this position as long as not the correct QR code has been scanned. The position of the Turtle which is specified in the code can be found in figure [4.2].

- The first value is of the longitudinal
- The second value is of the width.
- The third value indicates the direction of the front of the Turtle

The Turtle moves to the right position in the team line-up, when the right QR code is scanned, or the start signal is given. Above, the existing code from positioning is given in the else statement.

This part of the program is included in task_refbox_active_PASS.c.

![Figure 4.2](image.png) The location of the Turtle with Role 3 is showed, when the Turtle is still scanning the QR code or the start signal isn’t given yet.
4.2 Communication method

4.2.1 QR Code detection

After the Task Block is triggered, the coach software is running. This means that the QR code scanner starts. Zbar is chosen as program for QR scanning. Zbar is an open source software. It can detect and read multiple types of codes out of various sources.

*Code type:* EAN-13/UPC-A, UPC-E, EAN-8, Code 128, Code 39, Interleaved 2 of 5 and QR Code. These types of code can be seen in figure [4.3]

*File type:* Video streams, image files and raw intensity sensors can be used as sources to detect the codes.

In the coach software a video stream will be used to record a QR code. A picture is captured from the video stream when the QR code is in a good position. With this picture the QR code is analyzed.

Zbar uses a library that anyone can use after installing the package, in appendix [B] the installation of Zbar is described. The words from this library list are connected to functions from the Zbar program. The functions of these words are explained on a webpage. Zbar has written codes to scan an image on a particular location at the computer, but also a code to scan images from a video stream. The code for scanning a video stream can be used within the coach software. Only it needs a little modification. On the next page the code is explained on the basis of the library list.

```c
const char *device = "/dev/video0";
zbar_processor_t *proc = zbar_processor_create(1);
zbar_processor_set_config(proc, 0, ZBAR_CFG_ENABLE, 1);
```

*Explanation:* At first a video device is defined. In this case it is the webcam of the laptop. Now a processor is created, which will be configured with a '0' for success and another value for failure.

```c
zbar_processor_init(proc, device, 1);
zbar_processor_set_data_handler(proc, my_handler, NULL);
```

*Explanation:* Now a video input opens and a display output is prepared. A certain function is called by the processor when there are new results from the scan. This function is explained in the last code block. When there is no data a NULL value will be send out.

```c
int timeout = -1;
zbar_process_one (proc, timeout);
```

---

**Explanation:** If timeout has a value of -1, the process streams until a result is available. Another option is to set a value in milliseconds for the timeout, after it expire the streaming stops.

```c
zbar_processor_destroy(proc);
```

**Explanation:** All the information which the processor saved will be cleared.

```c
void my_handler (zbar_image_t *image, const void *userdata)
{
    const zbar_symbol_t *symbol = zbar_image_first_symbol(image);
    for(; symbol; symbol = zbar_symbol_next(symbol)) {
        zbar_symbol_type_t typ = zbar_symbol_get_type(symbol);
        const char *data = zbar_symbol_get_data(symbol);
        printf("decoded %s symbol \"%s\"\n", zbar_get_symbol_name(typ), data);
    }
}
```

**Explanation:** In this function the data, the text of the QR code, is processed of the scanner. The kind of type of the code is defined. In our case QR code will be released as type. Not only the code type but also the data is retrieved. In the terminal a text will be printed with the information. For example: “Decoded QR code symbol PSK”.

### 4.2.2 QR Code processing

When the QR code is scanned the data will be compared with the prescribed information. A certain number will set to the output, when the comparison between the two data matches. The time of the moment when the number is set to the output will be attached.

```c
const char str[][3] = { "PSK", "PSD", "PSB"};
int r;
puts ("Comparing QR results");
for (r=0 ; r<3 ; r++)
    if (strncmp (str[r], data, 3) == 0) {
        if (str[r] == str[0]) {
            retval = 2;
        }
        if (str[r] == str[1]) {
            retval = 3;
        }
        if (str[r] == str[2]) {
            retval = 4;
        }
    }
if ((strncmp (str[0], data, 5) != 0) && (strncmp (str[1], data, 5) != 0) &&
    (strncmp (str[2], data, 5) != 0)) {
    retval = 1;
}
```
4.2.3 Communication between the Turtles

The QR code is scanned through the Turtle with Role 3. Only this Turtle now has the knowledge on which strategy should be performed. This data has to be shared with the rest of the team. A piece of background information about the system needs to be explained before the code can be explained.

The software of a Turtle is made up of three parts; Mission, Vision and Worldmodel, see figure [4.4] section [A]. The coach software is part of Motion and is communicated via the code of `mccomm_motion.c`.

Each Turtle has a certain storage, which is divided in two parts. One is the Turtle's own memory, the other part is a shared memory, see figure [4.4] section [B]. This shared memory is again divided in six parts. Each part is managed by another Turtle, hence the numbers 1 to 6, see figure [4.4] section [C]. A Turtle can place information in another Turtle's shared memory. This information can then be used by the other Turtle.

![Diagram of communication system between Turtles](image)

*Figure [4.4] An overview of the communication system between the Turtles.*

The Turtle with Role 3 has scanned the QR code. This QR code is converted in a number. This is done in the S-function of the coach software, which governs by the Strategy block. This number with the time of that moment will be placed in the strategybus. Busses are used between the S-function for communication exchanges as explained in appendix [C]. The strategybus is linked to comm output + vision input-function, this is the S-function where the communication motion data is put, this is the part of figure [4.4] section C. All of the Turtles can use the information of this S-function.

The information, the number and the time of scanning of the certain QR code, in `mccomm_motion.c` file will now be requested in the `mcomm_teamstatus.c` file. In this part of code all of the information of the five Turtles will be compared on the most recent information. This can be done with the time which every Turtle has set to the output which is attached to a certain QR code number. The information of the most recently scanned QR code will be placed then in the teambus. The S-functions which needs the coach information can now retrieve the data from the teambus. In this case the `functions_refboxhandler` needs the information of the coach software for selecting the right strategy. The selection of the strategy will be discussed in the next paragraph.
4.3 Communication contents

After the QR code is scanned and the specific number is communicated to the Turtles, the right tactic can be performed. Each number stands for a certain strategy:

<table>
<thead>
<tr>
<th>Number</th>
<th>QR code</th>
<th>Tactic</th>
<th>Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Wrong QR code</td>
<td>Default</td>
<td>Case 1</td>
</tr>
<tr>
<td>2</td>
<td>PSK</td>
<td>pass_single+kick</td>
<td>Case 2</td>
</tr>
<tr>
<td>3</td>
<td>PSD</td>
<td>pass_single+dribble</td>
<td>Case 3</td>
</tr>
<tr>
<td>4</td>
<td>PSB</td>
<td>pass_single+dribble+BA</td>
<td>Case 4</td>
</tr>
<tr>
<td>5</td>
<td>PDK</td>
<td>pass_double+kick</td>
<td>Case 5</td>
</tr>
<tr>
<td>6</td>
<td>PDD</td>
<td>pass_double+dribble</td>
<td>Case 6</td>
</tr>
<tr>
<td>7</td>
<td>PDB</td>
<td>pass_double+dribble+BA</td>
<td>Case 7</td>
</tr>
<tr>
<td>8</td>
<td>DEF</td>
<td>Random</td>
<td>Case 8</td>
</tr>
</tbody>
</table>

The right tactic is selected in a switch by using the unique number. The switch is built in the refbox handler. Below a part of the switch is shown of the Kickoff with four cases. Each case in the switch contains another ratio between the tactics. After selecting the right case, the values of a,b,c,d will be completed in the percentage distribution.

```c
int a,b,c,d;
switch(Tactic){
    case 7: /* pass_double+dribble+BA */
        a = b = c = 0;
        d = 100;
        break;
    default:
        a = d = 0;
        b = c = 50;
        break;
}
N_variants = 4;
set_percentages(percentage_cum,N_variants,a,b,c,d);
set_pass_action_variant(rndr,percentages_cum, pS_RefP, S, N_variants,
pass_single+kick,pass_single+dribble,pass_single+dribble+BA,pass_double+dribble+BA);
```

For example, The QR code of ‘pass_double+dribble+BA’ is scanned, this QR code is linked to the number 7. Case number 7 will be selected in the switch above. This case contains a ratio of a = b = c = 0 and d = 100. These values will be entered in ‘set_percentages’. a,b,c,d are connected to the strategy variants in ‘set_pass_action_variant’. This means ‘d’ has a percentage of 100, so there is no other option than perform ‘d’ and that is ‘pass_double+dribble+BA’.

The strategies of the other refbox tasks are constructed the same as the kickoff and can be found in the script of functions_refboxhandler.c
5. Experiments

In this chapter the procedure of the experiments are described. First the maximum distance is recorded. This is done by changing the distance between the Turtle and the QR code. This is repeated several times so the best location could be recorded. Also a match of 2013 is analyzed so the numbers of possible coach moments are known, so there is an idea how often there can be coached.

5.1 Maximum QR scan distance

There are six different strategies which can be chosen in a match. The strategies are related to a QR code. All of these QR codes plus a QR code which performs a random strategy are tested in this experiment on the maximal distance. In the regulation of Robocup a maximal size of 30x30 [cm] of a QR code is allowed. Not only this size of a QR code is tested, but also the size of 20x20 [cm] and 10x10 [cm].

The QR codes are shown randomly at a large distance outside the range of the scanner. Every QR code has tested five times by moving the QR code towards the Turtle. Once the code was scanned the distance could be read off the measuring tape and also the result of the scan could be recorded.

The system is tested with this experiment on if it scans correctly and what the maximum distance between the coach and the Turtle can be.

5.2 Possible coach moments

A match is analyzed in order to get an idea of how often the opportunity there is to coach the team. The semifinal between Water (China) and Tech United Eindhoven in 2013 is reviewed.

The possible coach moments are listed if it met the following requirements;
- A refbox task for Tech United is selected
- Minimal 3 Turtles are on the field
- The refbox task is played on the opponent’s field

Of the selected coach moments the following data is recorded;
- The position of the Turtle with Role 3 when the referee gives a stop sign
- The time when the referee gives a refbox task
- The time when the referee gives the start signal after a refbox task is given

With this data there can be calculate if the QR code will be scanned on time before the refbox task will be taken place.
6. Results

Two experiments have been done. In the first experiment the maximal distance is defined. This is done by repeating the scan procedure of a QR code through the simulator. The results are described in paragraph [7.1]. In paragraph [7.2] the data is processed which is recorded of the semifinal of 2013 between Water (China) and Tech United Eindhoven. The possible coach moments are here discussed.

6.1 Maximum QR scan distance

The maximum distance is recorded through moving a QR code towards a Turtle. The measurements of this experiment can be found in appendix [D]. The measurements have been incorporated in the figure below.

As can be seen in the figure on the next page, there are three areas where the measurements are recorded. These areas depend on the size of the QR code.

- Measurements of the QR code size 30x30 [cm] are around the 500 cm
- Measurements of the QR code size 20x20 [cm] are around the 300 cm
- Measurements of the QR code size 10x10 [cm] are just above the 100 cm

All of the measurements were correct scanned, about 5 times there was a certain barcode scanned in the background. Only these scanned barcodes weren’t there. but because there is programmed that the scanning should be continue until a correct QR code is scanned or the 10 seconds are expired.

About 5 times there was a certain UPC-E barcode scanned in the background. Only these scanned barcodes weren’t there. All of the measurements were correct scanned, because there is programmed that the scanning should be continued until a correct QR code is scanned.

The strategies which were performed in the simulator, after a correct scan was done, were consistent with the expectations. They were performed correctly only two small notes are important.

- The strategies with the backwards attack were difficult to monitor, this because the backwards attack is not performed clear in the simulator. The pass_single+dribble+BA, for example, looked more like a pass_single+dribble with a strange movement during it.
- The QR code with the content DEF is used when the choice of the strategy doesn’t matter. In that case the strategy will be performed like in the current system, with a degree of randomly. The system chose every time for pass_single+dribble to perform in the simulator, when this QR code was scanned. This may have been a coincidence, or the ratio is high on this strategy or there is something wrong with the coach software on this particular QR code.
Figure [6.1] Measurements of the maximum distance, between the coach and the Turtle, with different sizes of QR codes.
6.2 Possible coach moments

In table [6.1] the number of refbox task moments is listed of team Water and Tech United Eindhoven. In the fifth column the possible coach moments which satisfy the requirements are listed. All of these moments are analyzed, whether the Turtle with Role 3 is on time for scanning the QR code.

One refbox task will be analyzed in this section. The method of the analysis is thereby illustrated. A throw in of the first half of the match is chosen as refbox task.

Step 1: A picture is taken of each refbox task which satisfies the requirements. In figure [6.2] the picture of the throw in is shown. In this picture the Turtle with role 3 is selected. It can be recognized with the red mark on it. A grid is added to the picture so the position of the Turtle in relation to the coach can be registered. In table [6.1] the values are listed of the distance. The distance between the coach and the Turtle can be calculated with formula (1). This is the shortest distance without obstacles.

\[ d_3 = \sqrt{x^2 + y^2} \]  
(1)

Step 2: The maximal velocity and maximal acceleration of the Turtles are listed in table [6.1]. With these data the time can be calculated which is needed to move to the coach in this particular case. The velocity curve of the Turtle is sketched in figure [6.3]. It contains an acceleration part, a maximal velocity part and a deceleration part. The acceleration part and deceleration part are similar with the time and distance.

First, the time is calculated which is needed to accelerate to the maximal velocity of the Turtle in formula (2).

\[ t_m = \frac{v_m}{a_m} \]  
(2)

The distance which is covered during the acceleration or deceleration can be calculated with the time of formula (2) in formula (3).

\[ d_a = \frac{1}{2} * a_m * t_m^2 \]  
(3)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_3$</td>
<td>Distance on the x-as from Role 3 to the coach</td>
<td>[m]</td>
<td>1</td>
</tr>
<tr>
<td>$y_3$</td>
<td>Distance on the y-as from Role 3 to the coach</td>
<td>[m]</td>
<td>8,5</td>
</tr>
<tr>
<td>$v_m$</td>
<td>Maximal velocity</td>
<td>[m/s]</td>
<td>4</td>
</tr>
<tr>
<td>$a_m$</td>
<td>Maximal acceleration</td>
<td>[m/s]</td>
<td>4,5</td>
</tr>
<tr>
<td>$t_3$</td>
<td>Time between the refbox task and the start</td>
<td>[s]</td>
<td>9</td>
</tr>
<tr>
<td>$t_{QR}$</td>
<td>Time needed for scanning</td>
<td>[s]</td>
<td>4</td>
</tr>
</tbody>
</table>
The remaining distance can be calculated with formula (4). In this formula the acceleration and deceleration distance is subtracted of the total distance which is calculated in formula (2).

\[ d_{aa} = d_3 - 2 \times d_a \]  
\( (4) \)

The time of the maximal velocity part can be calculated, when the distance of the maximal velocity part is calculated in previous formula. The time can be calculated through formula (5).

\[ t_{mv} = \frac{d_{aa}}{v_m} \]  
\( (5) \)

The time of all of the parts are now known. Through summation of these parts in formula (6) the total time which is needed to move to the coach is calculated.

\[ t_{tt} = 2 \times t_m + t_{mv} \]  
\( (6) \)

**Step 3:** Now it can be determined whether the Turtle is on time to scan the QR code before the start signal is given. The time between the refbox task selection and the start signal through the referee is listed in table [6.1]. The time to move to the coach and the time of the QR code scanning are subtracted of this time.

\[ t_{st} = t_3 - t_{tt} - t_{QR} \]  
\( (7) \)

The QR code is scanned when the result is positive. In that case the coach moment is registered in column 6 of table [6.3]. These numbers are the possible positive coach moments of this particular match. All of the data of this match can be found in appendix [E].

In table [6.2] the values of the throw in of the first half of the match are given. These are calculated with the formulas which are described above. As can be seen is the time after subtraction of the movement and QR code scanning positive. This means that this coach moment is listed in the positive possible coach moments.

**Table [6.2] Calculated values of a throw in in the first half of a match**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Value</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>( d_3 )</td>
<td>Distance between Role 3 and the Team Base Station</td>
<td>[m]</td>
<td>8,56</td>
<td>(1)</td>
</tr>
<tr>
<td>( t_m )</td>
<td>Time during acceleration to maximal velocity</td>
<td>[s]</td>
<td>0,89</td>
<td>(2)</td>
</tr>
<tr>
<td>( d_a )</td>
<td>Maximal distance during acceleration</td>
<td>[m]</td>
<td>1,78</td>
<td>(3)</td>
</tr>
<tr>
<td>( d_{aa} )</td>
<td>Distance after the acceleration and deceleration is subtracted of the total distance</td>
<td>[m]</td>
<td>5,00</td>
<td>(4)</td>
</tr>
<tr>
<td>( t_{mv} )</td>
<td>Time to cover the remaining distance at the maximal velocity</td>
<td>[s]</td>
<td>1,25</td>
<td>(5)</td>
</tr>
<tr>
<td>( t_{tt} )</td>
<td>Total time needed to cover the whole distance of Role 3</td>
<td>[s]</td>
<td>3,03</td>
<td>(6)</td>
</tr>
<tr>
<td>( t_{st} )</td>
<td>The spare time after scanning</td>
<td>[s]</td>
<td>1,97</td>
<td>(7)</td>
</tr>
</tbody>
</table>
Table [6.3] Match of 2013 between team Water from China and Tech United Eindhoven

<table>
<thead>
<tr>
<th>Refbox task</th>
<th>Water</th>
<th>Tech United Eindhoven</th>
<th>Possible coach moment</th>
<th>Scanned in time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>First half</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kick off</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Free kick</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Corner</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Throw in</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Goal kick</td>
<td>2</td>
<td>2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Penalty</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Second half</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kick off</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Free kick</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Corner</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Throw in</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>Goal kick</td>
<td>2</td>
<td>4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Penalty</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Extra time</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kick off</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Free kick</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Corner</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Throw in</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Goal kick</td>
<td>2</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Penalty</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Total Refbox tasks</strong></td>
<td><strong>26</strong></td>
<td><strong>32</strong></td>
<td><strong>21</strong></td>
<td><strong>17</strong></td>
</tr>
</tbody>
</table>
7. Conclusion

The project was set up for developing a system to communicate on a human way with the Turtles in the field during a game. The intention of this system is to coach the Turtles and make strategic moves, when a refbox task should be performed through Tech United Eindhoven.

A background research is done to several communications systems, such as voice recognition, gesture recognition and communication via symbols. Symbol recognition is eventually chosen as communication medium for the first version of the coach software, because this was a low threshold to start with.

Situations were sketched of two coach moments, on the base of that the system was divided in several parts which were analyzed and solutions are conceived. These solutions are converted into code and implemented in the motion part of the Turtles.

A coach system is developed that provides the possibility to influence the strategy, which should be performed during an active refbox task. A coach can show a certain QR code to the Turtle during a ‘dead’ game moment. The Turtle scans this QR code and processes the information. This information will be shared with the other Turtles on the field and the right strategy can be performed by the team.

The system is tested on correct QR scanning and the maximum distance between the coach and the Turtle. The QR codes were scanned correct most of time. A barcode was scanned about 5 times in the background, when it wasn’t there. This doesn’t matter because the scanning will be continued until a correct QR code is scanned or the time of 10 seconds is expired.

Other experiments should be done to record the real accuracy of the system. Now, it has been tested in a simulation environment, it would be better if it was tested with more Turtles on the field. A more realistic view can then be made of the performance of the system.

There can be concluded, after watching a game of 2013 between Water (China) and Tech United Eindhoven, that a lot of coach moments are possible during a match. This means about 54% of the active refbox tasks for Tech United Eindhoven could be coached.

The base is set for the interaction between the Turtles and the coach during a match. Still more tests should be done but this base can be improved and extended in the future. In next chapter the recommendation can be found for the following project on this subject.
8. Recommendation

In this chapter the next steps in the coach software are described for the future and the things which should be done before the next competition are listed. Further, a number of options are given where the coach software also could be used.

**The next steps in the coach software**

First, the coach software should be tested with the Turtles on the field and during a competition between several Turtles. Then it could be considered whether it works properly and if it should be added to the software system. For now it looks promising, but the coach software could be improved and extended.

One of the improvements which are possible is to increase the distance between the coach and the Turtle. The distance could be increased when the QR code could be earlier detected in the view. This can be done with the program from Demcon. This program searches to a field of interest in an image and follows it until is sharp enough, on that moment Zbar could scan the QR code properly.

The next step could be made in gesture recognition. The program of Demcon may ensure to recognize a field of interest in an image. This may be able to be used to recognize a person on the sideline. This could be combined with another gesture recognition program.

**The things which should be done on the coach software before a competition**

A several things should be done before it can be used during a competition.
- Program the software so it only detects QR codes and no other barcodes.
- Program the software so that Zbar not starts before the Turtle is on the right place.
- A better research to the libraries which are really needed for Zbar, so the installation can be included in the buildlibs_once script and the Turtles can be updated offline.
- Debug the startup of the simulator. Sometimes it starts Zbar while it is not the intention.
- Combine the Kinect software and the coach software.

**The combination with the coach software**

As mentioned in the project description it would be easy to communicate with the Turtles, so tunable parameters can be changed and strategies which go wrong can be avoided. These are not the only moments there could be communication with the Turtles. In order to ease the use of the Turtles, it could also be used to perform the calibration. Another example is during the demonstrations for companies, only a QR code needs to be shown and the Turtle performs a refbox task, like a penalty. Also the online learning project could be combined with the coach software, with the recorded data of the online learning platform, the right strategies could be chosen.
References


Appendix

A. Field distribution

- isInCornerRegion
- isInOwnCornerRegion
- isCloseToBorderAtOpponentHalf
- Ball_Pos[1] >= -1.9
- isInKickingRegionNear
- Ball_Pos[1] <= 0.0
B. Zbar installation

In this appendix the procedure of installing Zbar is described.

Step 1: Starting the internet via the terminal.
→ `sudo start network-manager`

Step 2: Zbar needs QT SDK for start the application software on the computer. QT software stands for QT Development Frameworks.
→ `sudo apt-get install qt-sdk`

Step 3: The library libv4l needs to be installed. This library makes it possible to use devices, like the webcam or Kinect camera.
→ `sudo apt-get install libv4l-dev`
   `cd/usr/include/linux`
   `sudo ln -s../libv4l1-videodev.h videodev.h`

Step 4: This is the step where Zbar is installed.
(There are more options which can be turned off, but for now this is the best option to install Zbar)
→ `cd svn/trunk/src/Turtle/Libs/zbar-0.10`
   `./configure --without-imagemagick`
   `sudo make`
   `sudo make install`

Step 5: Zbar is installed, with codes the program can already be used. If you want to use the already programmed codes of Zbar you need to install Zbar tools. It is not necessary, but it is easy to check whether the installation of Zbar is succeeded.
→ `sudo apt-get install zbar tools`

Step 6: Zbar can be tested through start zbarcam in the terminal. The installation is gone well, if the webcam is started on the computer screen with using this commando.
→ `zbarcam`

Note: Including Zbar library is needed for using the API documentation in the code. On the top of the code the library can be included with `#include zbar.h`
C. Introduction Tech United Eindhoven Software

The Robot Soccer of Tech United Eindhoven is an open source project. All the CAD models of the hardware and all the codes of the software can be found on the robotic open platform.\(^{14}\)

The software of the Turtles are divided in three parts; Vision, Motion and World Model. The focus on this project lies on a section of Motion. The software is kept organized through using Simulink, with this program clear schemes can be made. The Motion scheme can be found in figure [C.1].

As can be seen the scheme is build out of blocks, these block can consist several layers. The communication between these block goes through, so called, busses. These busses contain parameters which can be shared through the links between the blocks. See figure [C.1].

The lowest layer of the scheme has blocks which are called S-functions. These blocks contain scripts with codes. The parameters in the busses can be used in the S-functions. The early stages of the Coach Software is taken as an example for making an S-function and is located in Strategy section of the Motion part and will be explained on the next pages.

![Figure C.1. Represent the Motion part of the Simulink scheme. This is the second layer of the Motion scheme. The black and green wires are the links between the blocks.](http://www.roboticopenplatform.org/wiki/TURTLE)
C.2. Add a S-function in the Simulink Scheme

As described before the Motion part of the Turtle will be used as example for the explanation for adding an S-function in the Simulink Scheme. In this description the Interaction S-function will be add in the Strategy block and will be linked to the Task Executer S-function.

How to add an S-function block to the scheme?

- Open in the Current Folder the Motion folder.
  home → robocup → svn → trunk → src → Turtle2 → Motion
- Open the motion_turtle.mdl file, click twice on strategy & control (the blue block) and then click twice on strategy (orange block). See figure [C.2].
- As mentioned before, the S-function for the Interaction part will be added in this section. Open the Library browser under the heading View.
- Go in the Libraries to Simulink → User-Defined Functions. In the right part of the library browser you can find the S-Function block. See figure [C.3].
- The S-Function block can be dragged to the Simulink scheme.

Make a new script in the S-Function block.

- Open de S-Function block by clicking twice on it.
- Click on Edit and choose for Open Editor, now can be started with writing a c.-file. (How the c.-file should be written is described in the following paragraphs.)
- Save the file as following, go to File → Save As. Save the script as a .c-file like: Coach_Software.c
- The Coach_Software.c script should be included in the make_motion.m script in order to make and build the script. Otherwise the S-function will not work.
- Go for the make_motion.m function.
  home → robocup → svn → trunk → src → Turtle2 → Motion.
- Place the file name under the heading %control.
C.3. Build a script in the S-function

C.3.1. General

The script of an S-function can be divided in three sections:
- Defines and includes
- Callback method implementations
- Simulink product interfaces

The script will now be explained on the basis of these parts.

C.3.2. Defines and includes

Describe S-Function

The script begins with naming the S-Function as following:

```
#define S_FUNCTION_NAME Coach_Software
#define S_FUNCTION_LEVEL 2 /*Level 2 is related to the version of Simulink*/
```

Libraries

Matlab has standard libraries which are necessary to write in C-language and to be able to make calculations etc.

Data of other S-function can be used in another S-function, but only when the file is included in the top of the function file.

Several standard libraries of Matlab

```
#include "stdio.h"
#include "stdlib.h"
#include "math.h"
#include "simstruc.h"
```

Established libraries

```
#include "coaching_functions.h"
#include "bus.h"
```

The libraries can be written in two ways:
- include "example.h" is for files which are located in the same folder.
- include <example.h> is for files which are located somewhere else.

A remark: Not only .h-files can be included, also .c-files can be used if it is needed.
Define ports

Data can come in and leave a S-function, but it can only when ports are described. The description of a port is written below.

/*******************************
* Input Ports definitions *
****************-------------*/
#define NINPUTS 1 /* Number of input ports (0...). In this case is there 1 input port, the input ports starts counting at 0*/
#define NINPUTS0 TRCBUS_SIZE /* Tunable parameters motion bus */
/*(bus.h is included at the libraries, so the TRCBUS_SIZE can be used in this file.*/)*/
static double NINPUTS_TE[NINPUTS] = {NINPUTS0};

/*******************************
* Output Ports definitions *
****************-------------*/
#define NOUTPUTS 3 /* Number of output ports (0...)*/
#define NOUTPUTS0 1 /* An output port with one variable */
#define NOUTPUTS1 3 /* An output port with three variables */
#define NOUTPUTS2 3 /* An output port with three variables */
static double NOUTPUTSCOACH[NOUTPUTS] = {NOUTPUTS0,NOUTPUTS1,NOUTPUTS2};

C.3.2. Callback method implementations

The callback method implementations part can also be divided in several parts.
- mdlInitialize Sizes
- mdlInitialize Sample Times
- mdlOutputs
- mdlTerminate

Start of simulation

Initialization

- mdlInitializeSizes
- mdlInitializeSampleTimes

Simulation loop

- mdlOutputs

mdlTerminate
**mdlInitialize Sizes**

The parameter definitions part should be placed between the Input and Output definitions and the Input and Output properties. In this section the S-function parameters field is cleared, so no parameter mismatches will occur.

```c
#define NPARAMS 0
static void mdlInitializeSizes(SimStruct *S) {
    int_T Rworksize, il;
    ssSetNumSFcnParams(S, NPARAMS);
    if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount(S)) {
        return;
    }
}
```

Every port which is defined previously has several properties which are described below. In some cases the properties are described with a loop, because they have the same properties.

```c
/***************************
* Input Ports properties *
***************************/
int i;
if (!ssSetNumInputPorts(S,NINPUTS)) return; /*Specify the number of input ports that a block has */
for(i=0;i<NINPUTS;i++) /*Describes a loop for the input ports from 0 to the last one*/
{
    /* input port i */
    ssSetInputPortWidth(S,i,NINPUTS_TE[i]); /*Specify the width of an input port*/
    ssSetInputPortDirectFeedThrough(S,i,1); /*Specify the direct feedthrough status of a block's ports*/
    ssSetInputPortRequiredContiguous(S,i,1); /*Specify that the signal elements entering a port must be contiguous */
    ssSetInputPortDataType(S,i,SS_INT8); /*Set the data type of an input port */
}
/***************************
* Output Ports properties *
***************************/
if (!ssSetNumOutputPorts(S,NOUTPUTS)) return; /*Specify the number of output ports that a block has */
for(i1=0;i1<NOUTPUTS;++i1)
{
    ssSetOutputPortWidth(S,i1,NOUTPUTSCOACH[i1]); /*Specify the width of an output port */
}
ssSetOutputPortDataType(S,0,SS_INT8); /*Set the data type of an output port This port has type int8*/
ssSetOutputPortDataType(S,1,SS_DOUBLE); /*This port has type double and sends 3 variables to the task executer*/
ssSetOutputPortDataType(S,2,SS_INT8); /*This port has type int8 and sends 3 variables to the task executer*/

/********************
* Definition of States *
***********************/
ssSetNumContStates(S, 0);
ssSetNumDiscStates(S, 0);

/********************
* Default Definitions *
***********************/
ssSetNumSampleTimes(S, 1);
/* compute necessary amount of real_T workspace */
Rworksize = ( sizeof(sfun_global_data)/sizeof(real_T) + 1 );
ssSetNumRWork(S, Rworksize);
ssSetNumIWork(S, 0);
ssSetNumPWork(S, 0);
ssSetNumModes(S, 0);
In this section the definition of the sample time is described on which the S-function operates.

```c
static void mdlInitializeSampleTimes(SimStruct *S)
{
    ssSetSampleTime(S, 0, 1.0/MOTION_SAMPLE_RATE);
    ssSetOffsetTime(S, 0, 0.0);
}
```

```c
#define MDL_INITIALIZE_CONDITIONS
#if defined(MDL_INITIALIZE_CONDITIONS)
static void mdlInitializeConditions(SimStruct *S)
{
    /* get pointers to global data */
    real_T* ptrRwrk = ssGetRWork(S);
    psfun_global_data psfgd;
    psfgd = (psfun_global_data) ptrRwrk;
}
#endif /* INITIALIZE_CONDITIONS */
```
mdloOutputs

static void mdlOutputs(SimStruct *S, int_T tid) /* mdlOutputs compute the S-function's outputs at the current time step and store the results in the S-function's output signal arrays.*/
{
    psfun_global_data psfgd;                       /* Get pointers to global data */
    real_T* pRwrk = ssGetRWork(S);
    psfgd = (psfun_global_data) pRwrk;

    /* 'The pointers to output ports' and 'Output variable' parts, which are described in next paragraph (A.3.) should be written on this place. Make sure that the port definition and properties are correct! */
}

mdlTerminate

static void mdlTerminate(SimStruct *S) /* Perform any actions required at termination of the simulation */
{
}

C.3.3. Simulink product interfaces
This part should always be included in the script. Other ways the Simulink model cannot be compiled.

#ifdef MATLAB_MEX_FILE
/* Is this file being compiled as a MEX-file? */
#include "simulink.c" /* MEX-file interface mechanism */
#else
#include "cg_sfun.h" /* Code generation registration function */
#endif
C.4. Sending parameters between S-functions

C.4.1. One variable of the type int from the one S-block to another S-block

First block describing the variable (Can be found in: Coach_software.c file)

```c
/* Define Output Ports/
#define NOUTPUTS    1 ;               /* Number of output ports */
#define NOUTPUTS0   1 ;               /* The output port with one variable */
static double NOUTPUTS_TE[NOUTPUTS] =  {NOUTPUTS0};

/* Output Ports definitions */
ssSetOutputPortWidth(S,0,NOUTPUTS_TE[0]);
ssSetOutputPortDataType(S,0,SS_INT8);  /*Assigning data type to output port*/

/* pointers to output ports */
int* Tactic = (int*)ssGetOutputPortSignal(S,0);

/* Input variable */
*Tactic = 4;
```

Second block receiving the variable (Can be found in: task_excuter.c file)

```c
/* Define Input Ports */
#define NINPUTS    1 ;                /* Number of input ports */
#define NINPUTS0   1 ;                /* The input port with one variable */
static double NINPUTS_TE[NINPUTS] =  {NINPUTS0};

/* Input Ports definitions */
ssSetInputPortWidth(S,0,NINPUTS_TE[0]);
ssSetInputPortDirectFeedThrough(S,0,1);
ssSetInputPortRequiredContiguous(S,0,1);
ssSetInputPortDataType(S,0,SS_INT8);   /*Assigning data type to input port */

/* Input ports */
S_in.Interaction = (int*) ssGetInputPortRealSignalPtrs(S,10);

/* Showing the variable */
printf( "Print the variable in the terminal %d\n", *S_in.Interaction);
/*The variable '4' will be on the screen*/
```
C.4.2. A vector with variables of the type int from the one S-block to another S-block

First block describing the vector of type int (Can be found in: Coach_Software.c file)

```c
/* Define Output Ports */
#define NOUTPUTS    1;    /* Number of output ports */
#define NOUTPUTS0   6;    /*Output port with a vector with the length of 10*/
static double NOUTPUTS_TE[NOUTPUTS] = {NOUTPUTS0};

/* Output Ports definitions */
ssSetOutputPortWidth(S,01,NOUTPUTS_TE[01]);
ssSetOutputPortDataType(S,0,SS_INT8); /*Assigning data type to output port*/

/* pointers to output ports */
int* TacticIV = (int*)ssGetOutputPortSignal(S,0);

/* Input variable */
TacticIV[1] = (int) 2;    /* The variables which are described in the vector*/
TacticIV[2] = (int) 5;
...
TacticIV[6] = (int) 18;
```

Second block receiving the vector of type int (Can be found in: task_excuter.c file)

```c
/* Define Input Ports */
#define NINPUTS    1;      /* Number of input ports */
#define NINPUTS0   6;      /* Input port with a vector with the length of 6*/
static double NINPUTS_TE[NINPUTS] = {NINPUTS0};

/* Input Ports definitions */
ssSetInputPortWidth(S,0,NINPUTS_TE[0]);
ssSetInputPortDirectFeedThrough(S,0,1);
ssSetInputPortRequiredContiguous(S,0,1);
ssSetInputPortDataType(S,0,SS_INT8); /*Assigning data type to input port*/

/* Input ports */
S_in.Interaction = (int*) ssGetInputPortRealSignal(S,0);

/* Showing the variable */
printf("Send int vector with more variables to the terminator \%d\n", S_in.Interaction[2]); /* The variable '5' will be on the screen*/
```
C.4.3. A vector with variables of the type double from the one S-block to another S-block

First block describing the vector of type double (Can be found in: Coach_Software.c file)

```c
/* Define Output Ports */
#define NOUTPUTS    1;   /* Number of output ports */
#define NOUTPUTS0   10;   /*Output port with a vector with the length of 10*/
static double NOUTPUTS_TE[NOUTPUTS] = {NOUTPUTS0};

/* Output Ports definitions */
ssSetOutputPortWidth(S,01,NOUTPUTS_TE[01]);
ssSetOutputPortDataType(S,0,SS_DOUBLE); /*Assigning data type to output port*/

/* pointers to output ports */
real_T* TacticDV = (real_T*)ssGetOutputPortSignal(S,0);

/* Input variabele */
TacticDV[1] = (real_T) 1.0;
/* The variables which are described in the vector*/
TacticDV[2] = (real_T) 4.0;
...
TacticDV[10] = (real_T) 18.0;
```

Second block receiving the vector of type double (Can be found in: task_excuter.c file)

```c
/* Define Input Ports */
#define NINPUTS    1;    /* Number of input ports */
#define NINPUTS0   10;    /*Input port with a vector with the length of 10*/
static double NINPUTS_TE[NINPUTS] = {NINPUTS0};

/* Input Ports definitions */
ssSetInputPortWidth(S,0,NINPUTS_TE[0]);
ssSetInputPortDirectFeedThrough(S,0,1);
ssSetInputPortRequiredContiguous(S,0,1);
ssSetInputPortDataType(S,0,SS_DOUBLE); /*Assigning data type to input port*/

/* Input ports */
S_in.Interaction = (real_T*) ssGetInputPortRealSignal(S,0);

/* Showing the variable */
printf("Send double vector with more variables to the terminator %f\n",S_in.Interaction[2]);
/* The variable '4.0000' will be on the screen*/
```
D. Results of the maximum QR code distance experiment

<table>
<thead>
<tr>
<th>Size</th>
<th>QR code</th>
<th>Distance [cm]</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Measurement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30x30 [cm]</td>
<td>PSK</td>
<td>548 552 574 537 514</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSD</td>
<td>547 530 548 530 534</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSB</td>
<td>532 450 468 489 451</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDK</td>
<td>489 499 505 490 509</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDD</td>
<td>503 488 510 508 499</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDB</td>
<td>456 490 478 493 487</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DEF</td>
<td>474 442 451 480 467</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20x20 [cm]</td>
<td>PSK</td>
<td>315 292 289 298 291</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSD</td>
<td>300 302 286 299 308</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSB</td>
<td>272 275 288 270 295</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDK</td>
<td>297 294 288 299</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDD</td>
<td>290 315 306 274 287</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDB</td>
<td>278 275 291 282 287</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DEF</td>
<td>309 271 303 292 301</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10x10 [cm]</td>
<td>PSK</td>
<td>122 134 123 120 131</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSD</td>
<td>104 150 123 148 110</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSB</td>
<td>141 113 144 123 118</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDK</td>
<td>70  88  131 165 186</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDD</td>
<td>90  126  173 173 168</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDB</td>
<td>149 134 136 132 135</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DEF</td>
<td>71   99  166 174 176</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Size</th>
<th>QR code</th>
<th>Correct performance</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Measurement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30x30 [cm]</td>
<td>PSK</td>
<td>V V V V V</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSD</td>
<td>V V V V V</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSB</td>
<td>PSD PSD PSD PSD PSD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDK</td>
<td>V V V V V</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDD</td>
<td>V V V V V</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDB</td>
<td>PDD PDD PDD PDD PDD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DEF</td>
<td>PSD PSD PSD PSD PSD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
E. Results of the calculations for possible coach moments

<table>
<thead>
<tr>
<th></th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximal velocity [m/s]</td>
<td></td>
</tr>
<tr>
<td>Maximal acceleration [m/s]</td>
<td>4,5</td>
</tr>
<tr>
<td>Time to maximal velocity [s]</td>
<td>0,89</td>
</tr>
<tr>
<td>Maximal distance during acceleration [m]</td>
<td>1,78</td>
</tr>
<tr>
<td>Time needed for scanning [s]</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Refbox task</th>
<th>Time</th>
<th>x</th>
<th>y</th>
<th>Distance</th>
<th>Distance after maximal acceleration</th>
<th>Time to move at maximal velocity</th>
<th>Total time movement</th>
<th>Time before scanning</th>
<th>Not scanned in time</th>
</tr>
</thead>
<tbody>
<tr>
<td>First half</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 Kickoff</td>
<td>34</td>
<td>3,5</td>
<td>9</td>
<td>9,66</td>
<td>6,10</td>
<td>1,53</td>
<td>3,30</td>
<td>30,70</td>
<td>26,70</td>
</tr>
<tr>
<td>2 Kickoff</td>
<td>16</td>
<td>2,5</td>
<td>7</td>
<td>7,43</td>
<td>3,88</td>
<td>0,97</td>
<td>2,75</td>
<td>13,25</td>
<td>9,25</td>
</tr>
<tr>
<td>3 Throw in</td>
<td>8</td>
<td>0,5</td>
<td>7</td>
<td>7,02</td>
<td>3,46</td>
<td>0,87</td>
<td>2,64</td>
<td>5,36</td>
<td>1,36</td>
</tr>
<tr>
<td>4 Throw in</td>
<td>9</td>
<td>1</td>
<td>8,5</td>
<td>8,56</td>
<td>5,00</td>
<td>1,25</td>
<td>3,03</td>
<td>5,97</td>
<td>1,97</td>
</tr>
<tr>
<td>5 Free kick</td>
<td>9</td>
<td>1</td>
<td>6</td>
<td>6,08</td>
<td>2,53</td>
<td>0,63</td>
<td>2,41</td>
<td>6,59</td>
<td>2,59</td>
</tr>
<tr>
<td>6 Throw in</td>
<td>16</td>
<td>1</td>
<td>9</td>
<td>9,06</td>
<td>5,50</td>
<td>1,37</td>
<td>3,15</td>
<td>12,85</td>
<td>8,85</td>
</tr>
<tr>
<td>7 Throw in</td>
<td>10</td>
<td>1</td>
<td>7</td>
<td>7,07</td>
<td>3,52</td>
<td>0,88</td>
<td>2,66</td>
<td>7,34</td>
<td>3,34</td>
</tr>
<tr>
<td>8 Throw in</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>4,12</td>
<td>0,57</td>
<td>0,14</td>
<td>1,92</td>
<td>5,08</td>
<td>1,08</td>
</tr>
<tr>
<td>Second half</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 Free kick</td>
<td>7</td>
<td>2,5</td>
<td>6,5</td>
<td>6,96</td>
<td>3,41</td>
<td>0,85</td>
<td>2,63</td>
<td>4,37</td>
<td>0,37</td>
</tr>
<tr>
<td>10 Free kick</td>
<td>7</td>
<td>2</td>
<td>8</td>
<td>8,25</td>
<td>4,69</td>
<td>1,17</td>
<td>2,95</td>
<td>4,05</td>
<td>0,05</td>
</tr>
<tr>
<td>11 Kickoff</td>
<td>45</td>
<td>1,5</td>
<td>6,8</td>
<td>6,96</td>
<td>3,41</td>
<td>0,85</td>
<td>2,63</td>
<td>42,37</td>
<td>38,37</td>
</tr>
<tr>
<td>12 Free kick</td>
<td>11</td>
<td>2,5</td>
<td>8</td>
<td>8,38</td>
<td>4,83</td>
<td>1,21</td>
<td>2,98</td>
<td>8,02</td>
<td>4,02</td>
</tr>
<tr>
<td>13 Throw in</td>
<td>5</td>
<td>2,8</td>
<td>9</td>
<td>9,43</td>
<td>5,87</td>
<td>1,47</td>
<td>3,25</td>
<td>1,75</td>
<td>-2,25</td>
</tr>
<tr>
<td>14 Free kick</td>
<td>5</td>
<td>4</td>
<td>11</td>
<td>11,70</td>
<td>8,15</td>
<td>2,04</td>
<td>3,82</td>
<td>1,18</td>
<td>-2,82</td>
</tr>
<tr>
<td>15 Throw in</td>
<td>6</td>
<td>5,5</td>
<td>3,5</td>
<td>6,52</td>
<td>2,96</td>
<td>0,74</td>
<td>2,52</td>
<td>3,48</td>
<td>-0,52</td>
</tr>
<tr>
<td>16 Free kick</td>
<td>8</td>
<td>11</td>
<td>8,5</td>
<td>13,90</td>
<td>10,35</td>
<td>2,59</td>
<td>4,36</td>
<td>3,64</td>
<td>-0,36</td>
</tr>
<tr>
<td>Extra time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17 Free kick</td>
<td>10</td>
<td>0</td>
<td>7</td>
<td>7,00</td>
<td>3,44</td>
<td>0,86</td>
<td>2,64</td>
<td>7,36</td>
<td>3,36</td>
</tr>
<tr>
<td>18 Kickoff</td>
<td>11</td>
<td>3</td>
<td>4</td>
<td>5,00</td>
<td>1,44</td>
<td>0,36</td>
<td>2,14</td>
<td>8,86</td>
<td>4,86</td>
</tr>
<tr>
<td>19 Kickoff</td>
<td>9</td>
<td>7,5</td>
<td>3</td>
<td>8,08</td>
<td>4,52</td>
<td>1,13</td>
<td>2,91</td>
<td>6,09</td>
<td>2,09</td>
</tr>
<tr>
<td>20 Throw in</td>
<td>15</td>
<td>3,5</td>
<td>1</td>
<td>3,64</td>
<td>0,08</td>
<td>0,02</td>
<td>1,80</td>
<td>13,20</td>
<td>9,20</td>
</tr>
<tr>
<td>21 Throw in</td>
<td>20</td>
<td>6,5</td>
<td>3,5</td>
<td>7,38</td>
<td>3,83</td>
<td>0,96</td>
<td>2,73</td>
<td>17,27</td>
<td>13,27</td>
</tr>
</tbody>
</table>