Masters Thesis

The Reactive Virtual Trainer

Author:
Eike Dehling

Supervisors:
Dr. Job Zwiers
Dr. Dennis Reidsma
Ir. Herwin van Welbergen

Human Media Interaction
Department of Electrical Engineering,
Mathematics and Computer Science
University of Twente

March 16, 2011

Abstract

In this masters project, we set out to improve interaction of the virtual trainer with users. The goal was a trainer that supports a user with his workout goals like a real sports trainer: assist technically with an exercise and motivate the user to keep working hard. From this broad goal we have picked two main topics: we tried to make users perform exercises at a specified intensity and we have worked on motivating users by adjusting the feedback the trainer gives. We have developed and implemented a software architecture for the virtual trainer, and evaluated our work with the prototype.

Challenging a user with an exercise and motivating users during a workout are perhaps the most important tasks of a sports trainer. Working out at different intensities gives different training results: For example increased stamina or increased strength. Usually a human sports trainer looks at the users training goals and level of fittness to compile a workout that challenges the user and lets him achieve his goal. Many people sport not purely for fun, but to achieve some goal: For example lose weight or to be healthy. Those people sometimes rely on peers or the sports trainer for motivation.

To challenge a user, we started with two basic ideas: Performing an exercise faster is more intense and an exercise is most challenging when done at the optimal intensity. The trainer moves faster or slower, with the intention that the user picks up the change in speed. We measured the exercise speed using a wiimote and we measured the intensity using a heartrate sensor.

The evaluations showed that our prototype can influence the users speed in an intuitive way. To make users work out at a specific (optimal) intensity needs some more work, because the intensity of an exercise differs for each user.

Regarding motivation, we have applied aspects of several psychological models of motivation, politeness and mood to the virtual trainer. The trainer has a mood that is influenced by the users performance and the trainer chooses feedback of varying politeness, depending on that mood. We made videos of a baseline trainer and two trainer versions with motivational enhancements. We have showed these videos to users and asked them to compare the trainer versions on various aspects in a questionnaire.

Participants found the rude trainer and the friendly trainer more motivating than the baseline, the motivational factors competency and relatedness were rated higher. Participants found the rude trainer and the friendly trainer assisted them better. The rude trainer and the friendly trainer scored higher for positive character traits and the baseline trainer scored highest on negative character traits.

We have developed a software architecture for this project, looking at previous work in this area of software design. We have used a multi agent approach, where multiple agents generate behavior in parallel. The design has performed well, we did not encounter major issues, any changes needed during implementation fit in with the original design.

This project results in a system that intuitively influences the speed at which a user performs an exercise. Previous work only influenced users verbally. The other contribution is a trainer with aspects of several psychological models for generating feedback, which is new for virtual sports trainers. Finally, we have designed and implemented a software architecture for a sports trainer application.

Future work could develop a system to optimally challenge a user during a workout or build on our results to better motivate users.

Contents

Chapter 1

Introduction

This masters thesis is about my work on the Reactive Virtual Trainer project. The reactive virtual trainer is an automated virtual reality sports coach. It has sensors to perceive how the user performs the exercise and an avatar on a screen that can give verbal feedback and demonstrate exercises. Such a system could be useful when doing a sports workout at home, for assisting a real sports trainer with coaching a group or in a more advanced version, for assisting rehabilitation at home as a virtual physiotherapist. Training with a virtual coach can be more fun than just sitting on a home trainer, furthermore when used to assist real sports coaches, might reduce the number of sports coaches required.

Our work was focussed on the interaction with the user. The goal was to develop a trainer that supports a user with his workout goals like a real sports trainer, by assisting technically with the execution of an exercise and motivating the user to keep working hard. From this broad goal we have picked two main topics on which we have worked. Firstly, we have tried to make users perform exercises at a specified intensity. According to sports workout theory working at different intensities gives different training results: for example increased stamina or an increase in muscle strength. Secondly, we have worked on motivating users by adjusting the feedback the trainer gives. We have broken these topics down into several research questions that contribute to developing a better virtual sports trainer:

1. How can we make a user perform a workout at a specified intensity?
    - 1.1 How do we measure the exercise's intensity?
    - 1.2 Can we influence the exercise's speed?
    - 1.3 Is influencing the speed intuitive to the user?
    - 1.4 Does the exercise speed influence the intensity?
    - 1.5 Can we make the user do the exercise at a specific intensity?
2. Can we enhance a user's motivation in a sports workout by changing the feedback?
    - 2.1 How do users perceive a trainer that simply states mistakes versus a trainer that uses aspects of motivational theory to give feedback?
    - 2.2 Do users prefer a polite trainer or a more rude trainer?
    - 2.3 Does a trainer that uses aspects of motivational theory enhance motivation compared with a trainer that simply states mistakes?
3. How do we evaluate our improvements?
4. How do we implement a virtual sports trainer?

The first aspect we worked on was assisting the user, by making the exercise challenging. Specifically, we have worked on a way to influence the frequency at which users perform an exercise. By increasing the frequency of an exercise, the intensity is increased. The trainer uses a heart rate sensor to measure how intense an exercise is for a user and a Wii Remote to determine the frequency of motions. The trainer adjusts the intensity to try and optimally challenge a user. We have evaluated this system in three experiments with about 20 participants in total. The results have shown that we can intuitively influence the user's speed (and thus the intensity). To optimally challenge each user requires some more work, as the intensity of an exercise varies a great deal per user.

Secondly, we have worked on motivating users during a workout. We have looked at models of motivation and applied aspects of several psychological models of motivation, politeness and mood to the virtual trainer to enhance the training experience. Our trainer prototype gives feedback on mistakes the user makes, and adjusts his feedback depending on whether the user does his best to improve his performance. To evaluate the effects of these motivational mechanisms, we made three videos of a user performing the same workout and the same mistakes with a trainer that respond differently to those mistakes. We showed these videos to 16 users and gave them a questionnaire to rate the trainer on various aspects. The users found the trainers with motivational mechanisms more motivating. For example they felt more competent and said they felt closer to the trainer.

Finally, we developed a software architecture for implementing the virtual trainer application. The trainer was implemented using a multi-agent based approach, where the trainers behavior is produced by several cooperating agents.

Our first contribution is a trainer that does the exercise with the user and intuitively influences the users performance to challenge them. Previous work only influenced the user with verbal comments. The other contribution is a trainer with aspects of several psychological model for generating feedback, previous trainers have a rather simple system for generating audible or visual feedback. Finally, we have designed and implemented a working architecture for a sports trainer application.

This report starts with a background on previous work, motivation and working out. We continue to describe the goals, underlying algorithms, software architecture and implementation. After this are the results and conclusions. We finish with appendices that contain the results and a few things that did not fit into the main text.

Chapter 2

Related work

The idea of developing an virtual interactive trainer or coach is not new. Such systems have been developed for more than 10 years, becoming more advanced and intelligent over time. Perhaps workout and instruction videos were the first virtual trainers, current systems for example use computer vision technology, motion and bio sensors to perceive their users and have interaction models to generate different kinds of feedback when appropriate, using their virtual character.

Starting with the idea of making a truly interactive workout video, Davis and Bobick [1998] combined computer vision technology with recordings of feedback by a real coach. They demonstrated it was possible to develop an interactive virtual trainer with relatively simple technology. Compared to our work, they used much simpler methods but had similar goals.

An example of a more modern system is described by Babu et al. [2005]. Their computer vision system uses markers that can be tracked in 3D space. They have developed a virtual trainer that can demonstrate exercises, describe and show the user's mistakes, and praise correct execution of exercises. Their goals were to give high quality feedback on exercises, to use the system as a virtual physiotherapist.

Some projects have not developed a full interactive trainer application, but rather focussed on one aspect of a virtual trainer and researched how users respond to different solutions.

Chua et al. [2003] executed an experiment using a virtual Tai-Chi trainer, where users could see one or more virtual trainers from different angles. It is determined how effectively users learn the Tai-Chi movements. Showing the trainer from one angle taught users the Tai-Chi exercise just as good as viewing the trainer from multiple angles. Their application did not give feedback, the focus was to see if the possibilites given by virtual reality could enhance the learning process.

Similar experiments are described by IJsselsteijn et al. [2004]. A system is developed in which several parameters can be changed and enjoyment and effectiveness of the training are evaluated. They test a more or less immersive implementation and with or without a virtual coach that gives feedback on the user's efforts. They evaluated motivational factors of their application in a similar way as we did, showing that a more immersive application can enhance performance. They compared their application with a coach to a version without a coach, to see if the presence of a coach would enhance performance. In contrast, we compared the effect on motivation of different styles of giving feedback.

There are many projects in the field of virtual tutors, which instruct students in general tasks. Tutoring systems have been in development for a long time, so there might be some interesting insights.

The Steve project [Johnson and Rickel, 1997] was one of the first virtual reality tutoring systems where the tutor is represented by an embodied agent. In their application students learn tasks like operating a pump, the tutoring agent demonstrates the tasks and can explain why certain steps are necessary. Students wear virtual reality goggles and a glove with sensors to view and interact with the virtual environment. Their goal was to teach users a practical task, while we designed our trainer to motivate users.

In the Jacob project [Evers and Nijholt, 2000] a similar system was developed, but with different goals. They have added multi modal interaction and focussed on software engineering aspects. It is interesting to see how they applied traditional software engineering principles like layering and model-view-controller to a virtual reality agent, where we used a multi agent approach.

Previous work has been done on a virtual trainer at the Human Media Interaction group. Ruttkay et al. [2006] describe the design of a reactive virtual trainer, the system is divided into modules and approaches to implement those modules are proposed. This system was evaluated with users in [Ruttkay and Welbergen, 2008] and their feedback is discussed, as is some work on the system and details of the feedback module. Where their focus was on designing and actually building a virtual trainer, our goal was to make a more effective coach by improving on various aspects of other virtual trainers.

Chapter 3

Functional design

This chapter focusses on the functional design of the trainer prototype and background information. First we give some background on motivation, coaching and tutoring. Then we define an environment and choose a suitable sport. Then we describe a possible scenario in which the trainer is used and from this develop requirements. Finally we discuss our interaction model and describe some exercises and feedback.

3.1 Motivation and Tutoring

Ryan and Deci [2000] discuss general motivational theory. People can have different motivations to do certain activities, for example because they are rewarded for doing that, because they feel it is necessary to do it, or because they enjoy doing it. In general, motivation can be split into two categories: intrinsic (enjoying something) and extrinsic (an external factor that compels you to do something). There are several levels of extrinsic motivation, from for example fear of punishment to adhering to personal values and ideals, ranging from totally external to more internalized. Motivation is much stronger if it is more internalized or even intrinsic.

Different factors can help internalizing motivation: relatedness, feeling related or connected to the instructor or group, perceiving that one is good at something, competent and feeling in control, autonomous.

Concretely, relatedness means there is a mutual respect and care between the instructor and participant. Feeling connected to the instructor or group makes the participant want to leave a good impression and do his best. Feeling competent means that exercises need to be challenging but not to difficult, compliments from the coach can also support this feeling. Feeling in control means the instructor should not overly control everything the participant does, but rather support him autonomously, for example give advice, share insights and offer choice.

Brown and Levinson [1987] examined aspects of motivational theory from a linguistic point of view. Various politeness strategies and their effects on motivational factors are discussed. Utterances can use one or more of the politeness strategies depending on the situation. A model is proposed to select an appropriate utterance.

Johnson et al. [2004], Wang et al. [2005] and Cassell and Bickmore [2003] used these abstract models to adapt feedback from the coach to the state of the student. Varying politeness strategies are applied to feedback a coach gives, to influence the students perception of competence, autonomy and relatedness. For example, instead of bluntly telling a student to move his arms more, the trainer could say “Let’s not forget to work our arms!”. This avoids direct criticism of the student, which could hurt the mutual respect or the student’s feeling of competence.

Face is the public image a person tries to project. There are two parts: *positive face* and *negative face*. Positive face concerns someone's competence, negative face concerns someone's autonomy.

Speech acts that impact the student's feeling of competence are called *positive face threats*, for example "You're doing the exercise wrong!". Utterances that impact the students autonomy, giving someone orders, are called *negative face threats*, for example "Move your legs".

Saying speech acts that are threats to someone's positive or negative face in a more polite way to avoid that face threat, is called *face threat redressing*.

Paleari et al. [2005] used a similar model of politeness, but also included an emotional state of the trainer in generating feedback. They generate feedback based on the users performance and whether the user is trying hard. This gives three basic responses: (a) The user does well, the agent gives positive feedback. (b) The user performs bad but does his best, the agent gives a supportive comment. (c) The user performs bad and does not seem to do his best, the agent will respond negatively.

Johnson et al. [2004] and Wang et al. [2005] give feedback on errors as they occur, but positive feedback only at the end of an exercise. Their research suggests that giving too much positive feedback decreases its impact and might make the student doubt the sincerity. Giving positive feedback only at the end of an exercise seems reasonable, tests can show whether this works well for this application.

Various research on virtual learning environments also looks into display of emotion by virtual tutors. For example the work of Yanghee [2005] shows that display of emotion by a tutor enhances relatedness and thus motivation. There are many possibilities to show emotion. For example complimenting good performance or becoming annoyed by poor engagement.

3.2 Trainer Goals

The goal of the trainer is to support the user to achieve his workout goals. The role of the trainer is that of a sports coach, it gives technical feedback on the execution of exercises, helps plan workouts and tries to motivate the user to keep working hard.

As explained, three things are important to keep people motivated: relatedness, autonomy and competence. Relatedness means the user needs to feel socially connected to the coach or group. Other people are often a strong motivational factor to participate in sports and this effect also applies for virtual humans. Feeling competent means people feel they have achieved something, this good feeling is a strong motivator. Autonomy means people want to choose what they do, no one likes to do things they have to do, chores in the house or other things someone commanded them to do.

Making the user feel more autonomous can be done by letting the user choose certain things himself. For example the focus of the workout can be chosen by the user, one day he might want to work on his upper body and the next day the buttocks and lower extremities can be the focus.

While instructing, the trainer has to give comments on the users performance and execution of the exercises. Such comments can hurt the users feeling of autonomy or competence. Real coaches use various politeness strategies to avoid this. Instead of giving commands, the coach can offer suggestions for improvement or say things indirectly "Working harder trains your body more effectively". Trainers use supportive comments as well for this purpose, if not overdone and made sincerely, after good performance, remarks like "we're working really hard, this feels great!" motivate the user.

The difficulty of exercises is also important. If exercises are too easy, the user will get bored, on the other hand, if they are to hard it will frustrate the user. If the difficulty is right the user feels challenged and his sense of competence grows.

Relatedness means there is a social connection between the user and the trainer. Relatedness can trigger various behaviors between humans, for example a desire to please our peers. This means a user will try hard to keep a real trainer pleased with his performance. This effect would suits the goals of our trainer very well.

Some research suggests this relatedness can also appear between a human and a computer, this is called the persona effect. Research disagrees about how pronounced this effect is, Dehn and van Mulken [2000] are sceptical and theorize the entertainment value of an animated agent produces most of what is described as the persona effect, while Wang et al. [2005] and Paleari et al. [2005] are positive and ascribe their results to this effect. To trigger this persona effect, they added a personality and display of emotions to the trainer.

We will implement a personality in the trainer. The trainer will respond positive if performance is good, respond supportive if effort is high, but performance is lacking and respond negatively if effort and performance are bad. The intention here is, to trigger the student to work hard to please the trainer, at times were his performance is bad. Evaluation of the prototype will show whether this achieves the intended effect.

3.3 Intended environment

Recently, quite some home exercise games have been developed, so called *exergaming applications*. For the Nintendo Wii there is Wii Fit, Sony have developed the Eyetoy: Kinetic and there are several others.

The reactive virtual trainer is intended for the same usage: an entertaining way of getting and staying fit at home. The system is intended to be used on a regular computer using commonly available hardware.

3.4 Choice of sport

In this section we motivate which sport our trainer application will use. We start with some basics on working out. Next we describe the types of feedback a coach can give, which sensors are available for the coach and finally compare several sports according to their possibilities for feedback and required sensors. This allows us to choose a sport which suits our possibilities and goals.

3.4.1 Sports workout basics

A sports program can have different goals: Building cardio vascular fitness, increasing strength, weight loss or getting a more muscular appearance. Each of these goals requires a different type of exercise, cardio exercises, weight training or even stretching. A complete fitness program combines these types of exercise to train all aspects of the body's fitness. Each type of fitness exercise has some points to focus on for an effective workout. Below is a short overview of several different types of training and what is important for that type of training. [Fox and Mathews, 1987]

- **Warming up** warms up the muscles and prepares heart and lungs for exercise. During warming up the heart rate rises and oxygen transport to the muscles increases, allowing for more intense activities.
- **Cardio vascular exercise** trains the hearts and lungs capability to transport oxygen towards the muscles. For cardio vascular exercise the exertion of heart and lungs is important. The exertion can be measured for example by the heart rate, or by the volume of air being breathed in and out. Depending on the exact training goals, exercises can be of short duration and high intensity, of long duration and lower intensity or various combinations of these two.
- **Weight training** exhausts muscles, often with weights or training devices, causing them to grow stronger and larger, or increase their endurance. Muscles can be trained by doing many repetitions of one exercise with a lower weight, or by doing a few repetitions of an exercise with a big weight.

Depending on the type of training, the muscles will increase in maximum strength and size or will increase in endurance.

- **Stretching** Training stiffens and shortens muscles, decreasing flexibility and range of motion. Stretching restores muscles to their original length and restores flexibility. Stretching also helps muscles recover after a heavy workout.

A workout session usually starts with a warming up, to prepare the heart, lungs and muscles for the main workout. The workout then continues with strength training exercises, to increase muscle strength. After this are cardio fitness exercises, to train the cardio vascular fitness. To reduce muscle pains after a heavy workout, a session ends with some stretching. The strength training can be done after cardio vascular exercises, but muscles can be slightly fatigued from the cardio exercises, decreasing the efficiency of the strength training.

3.4.2 Feedback and sensors

A trainer can give different types of feedback, from assistance in compiling a workout program to motivational comments and feedback on the execution of motions. We start by describing what is meant by the different types of feedback the coach will give.

A trainer can motivate a sporter and incite him to train harder. This requires experience, to distinguish between an exhausted sporter who can not work harder and a sporter whose attention is wandering.

Beside motivation, feedback on the motions themself is important for many sports, to prevent injuries, to train the right muscles of even because the exact motions are the goal of the sport. The feedback on motions ranges from simple to very detailed, depending on the goal of the feedback. More detailed feedback requires the trainer to have better sensors. Giving the feedback can be done by demonstrating and explaining the motions or by giving a (short) verbal comment.

For many sports, compiling a workout is done using the student's goals as input. During training, the workout can also be adapted to the students performance: If the student struggles to keep up, the coach can adjust the exercises and make them less intense. A combination of vital signals and observing the student is required to be able to assess performance.

A trainer can give feedback on the timing of the student. This can be explicit, telling the student to perform an exercise faster or slower, or implicit, by performing the exercise a little slower or faster than the student, such that the student adapts to the trainer's rhythm.

Combining these types of feedback in the right way will be the challenge. A good coach will compile a challenging workout and push the student to perform at his best, but react quickly and adjust the exercises if they are too hard. A badly implemented trainer might compile a much too hard workout and scold the user for not performing well.

While the focus will be on the feedback the trainer gives, different types of feedback do require different sensors and varying degrees of accuracy. We will now describe different possible sensor solutions.

To give feedback with cardiovascular training exercises, the level of the sporter's effort needs to be measured. This can be done with a heart rate sensor, or with a device that measures the volume of air being breathed in and out [Fox and Mathews, 1987].

To give feedback on the sporters motions, a sensor is needed that can determine the position of his limbs and the orientation of his joints. For this a motion capture suit or a system using a camera could be used. Alternatively, for the prototype, an operator could do this task.

Vision based motion capture is still a very active research domain, as of now there is no off the shelf solution [Moeslund and Granum, 2001, Moeslund et al., 2006]. Precise and reliable motion capture has several technical restrictions (All limbs need to be visible, or the system is not real-time capable, or the system is not sufficiently reliable) and there is no default approach. Forsyth and Ponce [2002] say,

"This topic is not yet well enough understood for there to be a standard solution". Doing simple motion capture is possible, for example using colored markers to determine the position of limbs.

A motion capture suit (either with infrared markers and cameras or with motion sensors) does give precise and reliable results, however the user needs to wear a suit. This is less then ideal when doing physical exercises.

The preferred sensor solution for motion feedback is a camera based solution, however this limits the accuracy of input and therefore the level of detail of the feedback. So this will influence which sports the trainer can be used for.

Motivation can not be measured directly, this will have to be inferred from other measurements: for example whether the level of effort decreases, or whether motions get less precise, slower or smaller.

To give feedback on timing, we need to measure the timing of the students movements. This can be extracted from a computer vision system, or using motion sensors. The user could hold a simple motion sensor in their hands, for example a WiiMote controller.

3.4.3 Description of some sports

Sport or workout programs can combine many types of exercise into a full workout. Other sports only train one aspect. Team sports often emphasize cooperation and tactics, rather than individual strengths.

These differences between sports place different demands on a trainer. We will discuss goals and feedback options of several (somewhat randomly) selected sports, to be able to choose a sport with interesting feedback possibilities for the virtual trainer.

We will restrict ourselves to choosing an individual sport that can be done inside. This is partly because of practical reasons: displaying a virtual trainer and setting up sensors is much easier inside. Nevertheless, these sports already offer enough potential to generate interesting feedback and restricting us to these sports should not limit potential of the virtual trainer.

Information about these sports is partially from literature, partially from personal experience.

- **Yoga** was originally a form of meditation, but of course the relaxing stretch exercises can also be used in a workout [Baptiste, 2002]. The exact shape of motions is important in yoga exercises and a coach gives detailed feedback about these motions. The coach compiles an exercise program for participant depending on their level of experience. There are simple yoga poses for beginners and difficult poses for advanced participants.

- **Aerobics** combines cardiovascular-, strength- and stretching exercises into a complete workout program. Usually aerobics is done in a group to the rhythm of music. The trainer selects exercises that are challenging, train the desired body parts, starts with enough warming up and finishes with cooling down and some stretching. Because the goal is improvement of cardiovascular fitness and strength and the exercises are usually simple and use only the body weight as resistance, the trainer mainly motivates sporters and gives simple feedback on motions.

- **Weight training** is pure strength training. Because heavy weights are used, injuries can easily occur. It is important that the coach explains motions and risk for injuries well and gives good feedback on this. The coach also assists in compiling a workout and with settings goals.

- A trainer needs to give little feedback with **running and cycling** in a gym: Some assistance in setting goals and some motivation. Most sporters do these exercises on their own.

3.4.4 Choice

To select the most suitable sport, the possibilities for feedback are presented and compared below. Together with the comparison of sensor systems we made before, we can now select a sport.

	Simple motion feedback	Detailed motion feedback	Motivation	Workout compilation	Timing
Yoga		X		X	
Aerobics	X		X	X	X
Weight training		X	X	X	
Hometrainer, running			X	X	

Table 3.1: Feedback options for different sports programs

Of the above mentioned sports, aerobics allows the most types of feedback. Weight training is close, but needs detailed motion feedback, which means the motion capture sensor will have to be much better. This might mean the focus of the project would go towards developing a motion sensor, instead of generating and combining feedback.

Summarizing, aerobics allows the most types of feedback and allows us to keep the focus on interaction with the user instead of on sensors. The combination of workout compilation, motivation, motion and timing feedback will give the opportunity for interesting interactions.

3.5 Scenario

The scenario is a prosaic description of a possible session of a student with the trainer, which allows us to extract requirements for the software prototype.

> In the morning, you get out of bed. You stretch your arms and yawn. Your husband is making coffee in the kitchen and shouts "I'm taking a shower first!". Sometimes you go for a run before breakfast, but it's raining. You turn on the home entertainment system, web cam and television.
>
> Your virtual trainer greets you, "Good morning! Time for a workout, eh?". You choose to do a 30 minute core-body workout. The trainer selects a program of exercises that fits this goal. The trainer remembers you have been making good progress lately and some of the exercises were getting too easy. The trainer asks "Shall we make the exercises a little harder?" and you acknowledge.
>
> The workout starts with some basic warming up exercises. The trainer demonstrates all the exercises and observes how you do the exercises through the web cam. After five minutes the trainer registers from your heart rate that you are warmed up and the main workout begins.
>
> The first few exercises are easy, you have done them before. The trainer comments how good you are doing and continues the program.
>
> The next exercise is a new aerobics exercise: double sidestep and squat-jump. The trainer demonstrates and explains what you are going to do: "From you current position, make two big sidesteps to the left, do a deep squat and jump up. Then do the same but to the right. Go!"
>
> Your sidesteps are very small. The trainer notices this and pauses the workout. The trainer says "Watch how I do it" and demonstrates the complete exercise again. You try again but your side steps are still very small. The trainer pauses the exercise again and says "Move

your legs more, make big side steps!" and exaggerates his own side steps. This time you manage, the trainer smiles and says "Good work!".

3.6 Requirements

Based on the scenario we can extract requirements for the system such that it can carry out the previously described tasks. We have grouped the requirements and will discuss several requirements in more detail below.

1. Exercises and performances

    1.1 Exercise program. The trainer can execute an exercise program: explain and demonstrate an exercise, carry it out together with the user and then go to the next exercise.

    1.2 Exercise info. The trainer knows some details about exercises, for example a goal, how intense the exercise is and if possible ways to make the exercise more or less intense. This could be performing it slower, making the movements smaller or otherwise changing a part of the exercise.

    1.3 Tracking performance. The trainer tracks a users performance during the exercises.

    1.4 Reasoning about performance. The trainer can reason about current and recorded performance and deduce whether the user is working hard enough.

    1.5 Adapt exercises. The trainer can adapt exercises to the users performance, making them easier or harder as needed.

2. Sensors

    2.1 Bio sensors. The trainer has a heart rate sensor available to measure how intense an exercise is for a user, and whether the user is warmed up.

    2.2 Operator interface. The trainer has a user interface where the operator can input mistakes the user makes.

    2.3 Motion sensor. The trainer has a motion sensor to track the rhythm of the users movements.

3. Modalities

    3.1 Visual representation The trainer has a visual representation or avatar. The avatar can demonstrate exercises, performs the exercises together with the user and can emphasize certain parts of an exercise to teach the user the correct way to do the exercise. The coach will have simple facial expressions to accompany feedback, for example smiling when he compliments the user.

    3.2 Audio feedback. The trainer can give verbal feedback to the user, while choosing the exercise program, explaining an exercise or commenting on the users performance.

4. Interaction

    4.1 Interaction model. The trainer has an interaction model that produces feedback appropriate for the users current performance. It will give feedback on errors, give supportive or motivational comments, explain and demonstrate exercises.

The operator interface is used to input when the user does the exercises incorrectly. There are several alternatives, for example a motion tracking suit or a computer vision based solution. Using a motion tracking suit or computer vision based approach may even offer more or more accurate information than an operator can provide.

3.7 Interaction model

In section 3.2 the goals of the trainer project were discussed. Starting with the main goal, to assist and motivate a user during a workout session, several subgoals were developed: the trainer needs to use politeness, the trainer will have a personality and mood, the user is presented with choices and the trainer will adjust the difficulty of exercises. In this chapter we go into more detail about these subgoals and combining them.

Real trainers give different feedback on the same mistakes depending on the users performance and their mood. If the user does well, give him positive feedback and show positive emotion. If the user does not do well, but tries hard, give supportive feedback. If the user does not do his best, the trainer can show negative emotion and will give strict or even annoyed comments.

A trainer can also use various politeness strategies to convey criticism and advice in ways that do not harm the social relation between the trainer and the user: criticism and advice can harm the users feeling of competence and autonomy. Whether and how much politeness is necessary depends on the type of criticism and the relation between the trainer and the user. Severe criticism needs a more polite approach, while small tips can just be said as they are.

These two approaches can be combined into one model for feedback, the trainers mood selects the level of politeness the trainer uses, like a real trainer. Depending on the mood, the trainer will then be more or less careful when giving feedback.

3.7.1 Politeness

We will use the politeness model from Johnson et al. [2004] and Wang et al. [2005] (Equations (3.1), (3.2)) to select an appropriate politeness strategy. In these formulas, $Politeness_x$ is the amount we need to redress positive or negative face threats. *Distance* refers to the social distance between the student and the trainer, while *Power* is the social power of the trainer over the student. $Threat_x$ is the inherent positive or negative face threat of this type of feedback. $Augmentation_x$ means the desired amount of augmentation of the students positive or negative face. For example if the trainer thinks the user is not feeling competent, he might want to augment the users feeling of competence.

$$Politeness_{positive} = Distance - Power - Threat_{positive} - Augmentation_{positive} \tag{3.1}$$

$$Politeness_{negative} = Distance - Power - Threat_{negative} - Augmentation_{negative} \tag{3.2}$$

The social power and distance model the relation between the student and coach. They are usually implemented as constant parameters, but could be varied in time to simulate the student and coach getting to know each other. We will use fixed values.

The inherent face threats of feedbacks are a measure of how threatening that general type of feedback is to the users positive or negative face. Similarly, the polite feedback versions can also be ranked by how (non-)threatening that specific feedback is. The outcome of the formula then selects the appropriate polite feedback version.

To find polite feedback versions, recordings of sessions with an experienced instructor were analyzed by Johnson et al. [2004]. They categorized the feedbacks and identified the politeness tactics the real instructor used. Table 3.2 shows some examples. We can use their findings to produce polite versions of the feedbacks our coach gives, in a similar way a real trainer would give polite feedback.

3.7.2 Mood

The augmentation of the student's face indicates the desired level of politeness of the coach, varying this parameter changes how (im-)polite the trainer is towards the user. There are different approaches to

Type of feedback	Politeness strategies
Suggest action	Bald on record, conventional indirectness, joint goal, student goal, question, suggestion, tutor goal
Explain concept	Bald on record, positive politeness, attend to hearer, students goal, impersonalized, off record
Explain tutorial	Bald on record, tutor goal, joint goal, suggestion
Socratic hint	Socratic hint
Action feedback	Bald on record, positive politeness

Table 3.2: Examples of types of feedback and employed politeness tactics employed by real trainers [Johnson et al., 2004]

setting this parameter. It can be a constant value, representing a desired level of politeness. In other research the desired politeness is changed dynamically, it can for example be adapted to the conversation partners politeness. Changing the politeness dynamically can have different effects, real humans also do this, if they become more familiar with someone, on purpose, or if they are in a bad mood.

Hofs et al. [2010] used linguistic analysis to let a tour guide mirror the users level of politeness. Mirroring the users level of politeness is intended to increase the humanness of the guide, making him appear more socially intelligent and create a more believable personality.

The tutor of Paleari et al. [2005] can give feedback in three different styles depending on the users performance: positive if the user performs well, supportive if the users performs bad but is doing his best and moody if the user is not doing his best. This might seem counterproductive, as politeness is designed to motivate the user and being moody because of bad performance should not help. However, this moody behavior can motivate the user even more. An explanation would be that the tutors apparent bad mood activates the social desire to keep him pleased, causing the student to work harder.

Our goal is to motivate the user, so we will try to use this motivating effect of giving moody feedback to our advantage. The rest of this section discusses how we implemented this idea in the trainer prototype.

To include the model from Paleari et al. [2005], where the trainers mood and responses are influenced by the efforts of the user, we need to determine if the current performance is good and whether the user is doing his best.

To determine whether the user is doing his best, we need to define what we mean by doing his best. This could mean, working hard physically, or for example trying to follow the coaches instructions and do the exercise as good as possible. The second definition probably suits the trainer application best: physical effort and heart rate are already topics the trainer will give feedback on. So for trainer, doing your best will mean following the instructions.

A measure of trying to follow the coaches instructions would be the number of mistakes the user makes per time unit. We can track the number of mistakes a user makes per time unit and determine whether this is increasing or decreasing and use this to determine whether the user is doing his best to follow the coaches instructions. For example if the student previously made 2 mistakes during execution of an exercise and now makes 6 mistakes, his current performance is not good.

The trainer will have a mood-score, which is influenced by the users performance, that changes the style of feedback the trainer gives. In the prototype we will calculate the number of mistakes a user made per minute over the last minute and last 3 minutes. Once per minute we will compare those numbers, check whether performance has improved or decreased and then increase or decrease the mood score by 1 point, within the range $[-5, 5]$. The mood-score is then used in the politeness formulas (3.1) and (3.2) as the desired augmentation of positive and negative face, determining the level of politeness. A high mood-score produces polite feedback and a low mood-score produces more blunt feedback.

3.7.3 When and on what error to give feedback

When to give feedback was briefly discussed in the background chapter. Too frequent positive feedback might be annoying, become repetitive to users and have unwanted effects. Too much negative feedback might have bad effects as well: the user needs a little time to adapt to the coaches feedback, getting too frequent negative feedback might be annoying or make the user might feel he is not good at this sport.

To determine when to give negative feedback and about what error to give feedback, there are several relevant parameters: number and type of errors per time-unit, what feedback have we recently given and current performance. If performance is good, we can give feedback less frequently than when performance is bad. We can only give feedback on one error at a time, so we need to choose which error is currently most important. The higher the frequency of an error, the more important it is to give feedback on that error. The coach should not focus entirely on one error though, but also give feedback on less frequent errors from time to time.

$$Importance = Frequency - A * Improvement - B * Mentioned \tag{3.3}$$

In formula (3.3), $Importance$ is an importance score calculated for each type of error. $Frequency$ is the frequency we saw the error over the last time unit, $Improvement$ is the change in frequency compared to the previous time unit (last 1 and 3 minutes). $Mentioned$ is the number of times we recently have given feedback on this type of error. The parameters A and B tune how much each of the variables affects the score of an error, changing behavior from only commenting on the most important error to commenting on errors alternatingly.

In the prototype we used the values $A = 1, B = 5$, which means after feedback on one mistake we favor giving feedback on some other mistake.

Every 30 seconds the trainer prototype calculates the score of all mistakes. If the highest scoring mistake has occurred at least once in the last minute, the trainer then gives feedback on that mistakes.

3.7.4 Adjusting difficulty

Exercises at a challenging yet doable difficulty level provide the most satisfaction when they are completed successfully. To have exercises that are easy enough for beginners and still challenging for more advanced users, the trainer needs exercises for each difficulty. This either means many exercises of varying difficulty, or the trainer needs to be able to change the difficulty of an exercise as needed. For practical reasons, we will implement changing the difficulty of exercises.

The most suitable mechanism for this, is changing the speed at which exercises are executed. By monitoring the heart rate, the coach knows whether the user is working hard and can adjust the difficulty accordingly. If the heart rate is too low the trainer increases his speed, if the heart rate is too high the trainer reduces his speed. There is an upper limit and a lower limit on the trainers speed to keep things reasonable. We hope the user will follow the trainers in increasing the speed, and also increase his own speed.

3.8 Example workout and feedbacks

A workout session consists of different parts. Each session will start with a warming up, to prepare the heart, lungs and muscles for the main workout. The workout then continues with strength training exercises, to increase muscle strength. After this are cardio fitness exercises, to train the cardio vascular fitness. To reduce muscle pains after a heavy workout, a session ends with some stretching.

Each exercise is preceded by an explanation and demonstration, then the trainer and user do the exercise together. The explanation of the exercise includes a description of the movements and also

states the goals of the exercise (warming up, strength training, cardio fitness). While doing the exercise, the trainer gives verbal feedback or stops the exercise if something needs to be demonstrated.

3.8.1 Exercises

Below we describe several exercises, explain how they are done and list the feedbacks a trainer could give.

Side Steps	
Goal	Warming up muscles, preparing heart and lungs for workout.
Instructions	This exercise is for warming up. Stand with your arms hanging beside you. Step sideways with one leg and at the same time spread your arms. Then move your other leg to the new position and let your arms hang again. Then we repeat this in the opposite direction. We will repeat this a few times.
Feedback	• Incorrect movement with the arms or legs • Too small movements with the arms or legs • Movement of arms and legs not synchronized • Not working intensely enough

Jumping Jacks	
Goal	Warming up muscles, preparing heart and lungs for workout.
Instructions	This exercise is for warming up. Stand with your arms hanging beside you. Jump and land with your legs standing wide apart, at the same time spread your arms. Now jump and land with the feet together and simultaneously move your arms close to your body again. We will repeat this a few times.
Feedback	• Incorrect movement with the arms or legs • Too small movements with the arms or legs • Not keeping the back straight • Not working intensely enough

Squats	
Goal	Strength training for legs and buttocks.
Instructions	This exercise trains the strength of your legs and buttocks. Stand with your arms stretched out in front of you. Now bend your knees and lower yourself, maintaining an upright position. Straighten your knees and stand up again. We will repeat this a few times.
Feedback	• Incorrect movement with the arms or legs • Too small movements with the arms or legs • Not keeping the back straight • Not working intensely enough

Head-Shoulders-Knee-Toes	
Goal	Strength training for legs and buttocks.
Instructions	This exercise trains the strength of your legs and buttocks. Stand up straight, touch the top of your head with both hands. Move your arms and touch your shoulders with your hands. Bend a little and touch your knees. Bend further until your hands touch your toes. Repeat in opposite order. At the end, jump and reach as far up with your hands as you can. We will repeat this a few times.
Feedback	• Incorrect movements • Too small movements • Not keeping the back straight • Not working intensely enough

3.8.2 Feedbacks

Below is a list of all feedbacks the trainer can give for the example exercises.

- Incorrect movement of the arms (spoken + visual demonstration)
- Incorrect movement of the legs (spoken + visual demonstration)
- Incorrect movement (spoken + visual demonstration)
- Too small movement of the arms (spoken)
- Too small movement of the legs (spoken)
- Too small movement (spoken)
- Not keeping the back straight (spoken)
- Movement of arms and legs not synchronized (spoken)
- Not working intensely enough (spoken)

We will list polite feedback versions of "too small movement of the legs" as an example, the exact polite versions of the other feedbacks are left out of this report.

In the section that describes the interaction model, we mentioned different types of utterances and which politeness tactics can be used with that type of utterance. The feedbacks of the coach combine the *suggest action* and *action feedback* type of utterance, as they comment on the performance and suggest how to improve it. For "too small movement of the legs", the action feedback is that the movement of the legs was to small, and the suggested action is to move the legs more. For the action feedback we can use the *bald on record* and *positive politeness* strategies, for the suggest action we can use the *bald on record, conventional indirectness, joint goal, student goal, question, suggestion,* and *tutor goal* strategies.

The action feedback can be said using different strategies, first bald on record:

- The movements of your legs are too small!
- You are not moving your legs enough!

Or it can be said using positive politeness:

- You're doing great, but your legs do not move much.
- Wow, we're working really hard! We only make to small motions with our legs.

The suggest action part has many ways to say it using the different strategies. First bald on record:

- Move your legs more!
- Make bigger movements with your legs!
- Move your legs as much as i do!

Or using indirectness:

- Can you see how big movements i make with my legs?

The trainer can motivate the user by referring to his, the users or joint goals:

- I make big motions with my legs, because i want to train them.
- Big movements with your legs train them better!
- Let's make bigger movements with our legs to train them!

The feedback can be phrased as a question:

- Don't you want to move your legs more?
- Can you move your legs more?

Or as a suggestion:

- You could make bigger movements with your legs!
- You could move your legs more!

Chapter 4

Architecture and implementation

This chapter discusses the technical design of the prototype software and the implementation of that design.

4.1 Software design

In this section different aspects of the prototype trainers software system are described. We present state diagrams for the prototype application, describe how the software is divided into modules, go into more detail for some of the modules where relevant and discuss communication between the modules.

4.1.1 State diagram

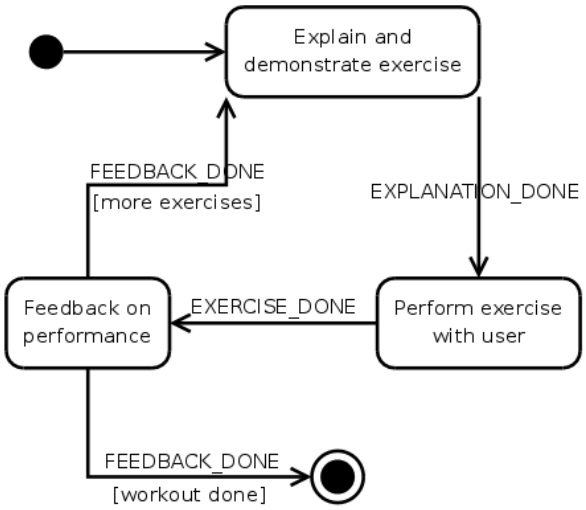


Figure 4.1: Program states during a workout session

Figure 4.1 shows the program states that during a workout session and what events trigger progression to the next state. Figure 4.2 shows the interactions while performing an exercise with the user, this elaborates the "perform exercise with user" state from figure 4.1. Capital labels are events that trigger a state transition and texts in brackets are conditions, which select which transition will be used.

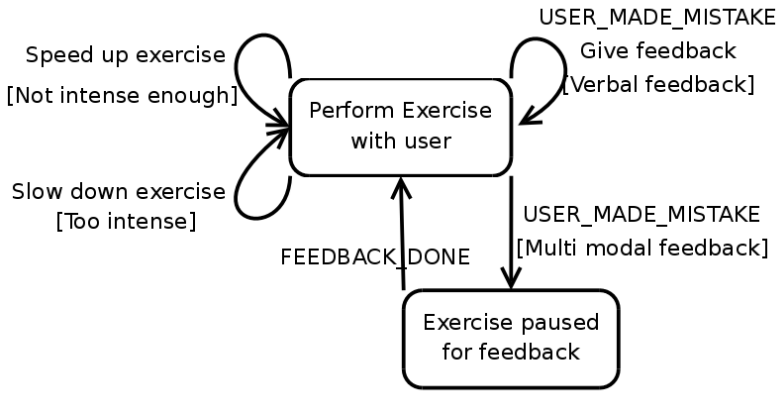


Figure 4.2: Interaction with the user during an exercise. This is a refinement of the "Perform exercise with user" state in figure 4.1. If the feedback is only spoken the trainer can continue doing the exercise in parallel, if the trainer needs to demonstrate an aspect of the exercise (this is labeled as multi-modal in the figure) execution of the exercise needs to be paused until the feedback is done.

The program starts by demonstrating and explaining the first exercise. The trainer then does the exercise together with the user and can give verbal feedback or additional instructions during the exercise. If the type of feedback requires it, the exercise can be paused and the trainer can for example demonstrate something. The intensity of the exercise is adjusted by speeding it up and slowing it down, the trainer will perform motions just before or after the user so the user adjusts his speed. After the exercise the trainer can give a motivational comment.

If the whole workout is done, the program stops. If the workout is not finished, the trainer will continue with the next exercise.

4.1.2 Architecture background

The virtual trainer is a type of software that is commonly called an embodied agent. There is much previous work on implementing embodied agents and there are different approaches that have proven successful, while other approaches have been superseded. Knowing the requirements of our software we can look at several options on implementing the reactive trainer and choose which features suit our application.

Wooldridge [2002] discusses several classes of agent designs and concrete examples, in order of complexity: Advantages, disadvantages and applications of the designs are discussed. Reactive agents are suited for very simple problems. Reasoning agents are mainly used in theoretical environments. Hybrid agents come from the realization that different tasks of an agent have different requirements and combine different layers, each with their own responsibilities.

Hybrid agent designs have some interesting ideas. There are several independent layers which suggest actions for the agent, for example a reactive layer which quickly reacts to input events and one or more deliberative layers which propose a course of action using recorded data or a model of the world. Hybrid agents have a mechanism to select which suggested action is executed. There are different solutions, each with their own merits and drawbacks. Some agents have a control module that receives suggested actions from all layers and selects which suggested action is performed, this could for example be done by assigning priorities to all possible actions. Other designs pass inputs up and suggested actions down through the layers, lower layer actions can then take precedence over higher layer actions (Compare to reflexes in humans). Wooldridge [2002] have a in-depth discussion of the merits of various approaches.

Layered designs are quite flexible: they allow for complex behaviors and also quick responses if needed, however coordinating control between the different layers is a complex issue.

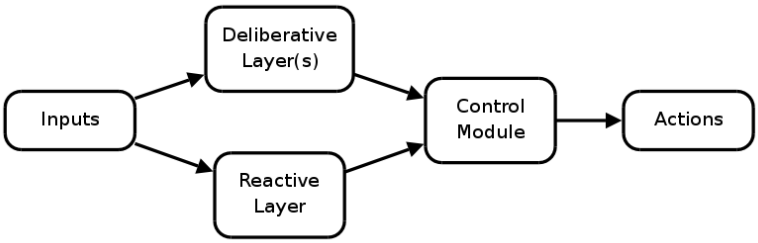


Figure 4.3: TouringMachines [1]: An example of a hybrid agent architecture.

Sloman [2003] has studied architectures of artificial and models of real minds, with a goal of understanding and being able to design (parts of) an artificial mind. He describes different features, their role in agent architectures and presents example architectures for different applications. As the basis, information flows through three stages, from sensory modules through processing modules to action modules. The processing modules are divided into reactive, deliberative and reflective layers by the type of task they perform. Reactive and deliberative layers are the same concepts as in hybrid agents, the reflective layer contains learning tasks, which look at the performance of the system and attempt to learn from previous success or failure. There are some extra mechanisms, memory modules to store information, an alarms module to communicate with or alert other modules, information filters to filter inputs and a control module that decides which actions to execute.

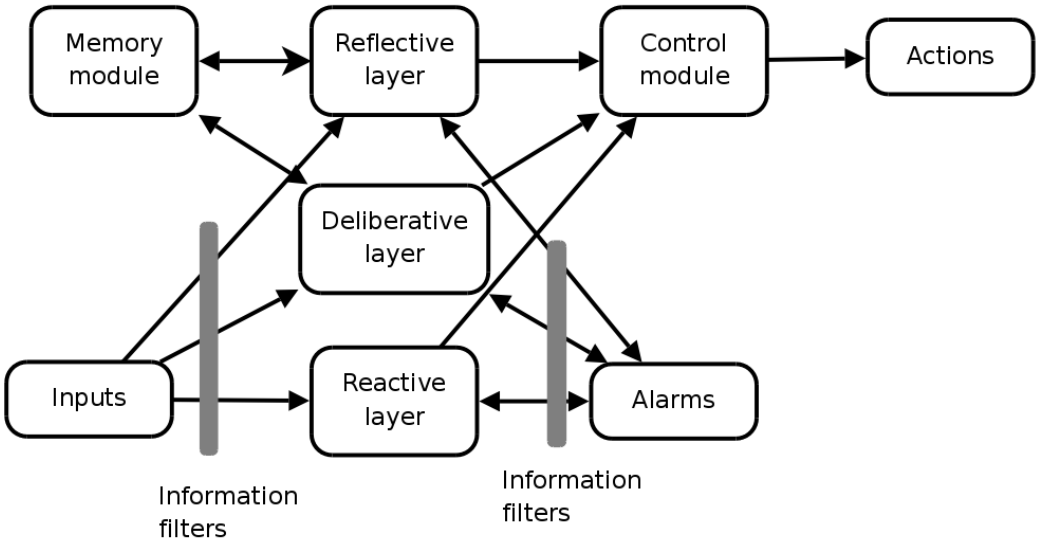


Figure 4.4: The CogAff architecture [Sloman, 2003]

Not all (artificial) minds need all those modules, for example the class of reactive agents only has the lowest layer of processing modules, no memory and no information filters. Features can be selected depending on the requirements.

[1] TouringMachines agents controlled a mobile robot in a virtual world. Touring refers to the robot moving through the virtual world. The name might be a humorous reference to the theoretical computing machines of Alan Turing.

Thórisson et al. [2004, 2008] describe a methodology to design software systems where a single external body is controlled by multiple internal functionalities. An additional goal is to support incremental development.

They propose to split functionality into modules, grouped into perception, decision/planning and action/animation modules. Within each group, modules are classified as low, medium or high level. Modules communicate using blackboards or a publish/subscribe mechanism. A blackboard is a module that stores and distributes messages and has a supervisor that decides who may handle a message. The state of the system should be in the messages exchanged using the blackboards, which implies stateless modules. They reason, if all data is in the messages, it is easier to later add modules that require access to that data.

4.1.3 Considerations for our architecture

There are several considerations that are important to discuss, to understand the reasoning behind the system architecture. We have discussed several other software architectures for agents and the issues they tried to address, we will try to apply their solutions to solve issues in our software design.

Each section below discusses some issues and how we will solve them.

Splitting functionality into modules

The interaction model described previously encompasses much functionality. Wooldridge [2002] discusses some monolithic approaches, using deductive logic and planning algorithms to achieve their goals. This resulted in poor performance, complicated planning algorithms or logic systems are not suited for solving problems in real-time. Sloman [2003], Thórisson et al. [2004, 2008] split the behaviors into many smaller blocks communicating via messages, implementing the behavior using a sort of multi-agent system. This approach allows to change or add modules afterwards, giving a flexible, extensible design.

The modules we split the software into will have well defined responsibilities and we will attempt to keep modules from becoming too complex.

Timing requirements

Some modules require frequent updates and precise timing, for example the speed adjuster and wii input modules cooperate to predict timing of user motions and make the trainer move a little quicker. Doing this in real-time requires frequent inputs and adjustments.

Other modules do not have these strict timing requirements. For example updating the trainers mood or giving feedback needs to happen regularly, but it does not matter if it happens a few milliseconds earlier or later.

The software designs we discussed organize modules into layers, according to timing requirements and type of task. Input modules act immediately upon input events. A reactive layer for simple behaviors which require quick action. A deliberative layer for tasks that use some reasoning or planning to choose their actions. The trainer does not have reflective tasks (yet).

Communications system

There are several approaches for communication between modules. Sloman uses an alarms module to signal other modules and (persistent) memory for storing and sharing data. Thórisson uses blackboards and stateless modules, keeping as much state and data as practical in the messages that are exchanged. Some systems do without a central communication system, only passing messages directly between modules.

With an alarm system Sloman mean a centralized messaging system, for example a publish subscribe event service. A publish subscribe event service is a place where modules can register themselves to

receive specific types of events and publish events so all registered modules receive their updates. The centralized messaging means, a module does not need to know who wants to receive his messages or is interested in his events. This can be useful if one module needs to send messages or events to many other modules, or if more modules are added later.

A blackboard is also a centralized communication system. There are different types of blackboards, but they have common goals: communication between many modules and storing the systems state. With some blackboards, modules need to poll for new messages, with others the modules have a callback. Some blackboards have filtering functions to limit unnecessary callbacks. As the state of the system is stored in the messages on the blackboard, multiple modules might receive a message, but only one can alter the systems state. Blackboards use a supervisor module to control this, which selects who may respond to a message.

While a blackboard is technically similar to a publish subscribe service, it has a different intended function: distributing and coordinating work orders versus exchanging messages.

Modules can also communicate directly, possibly using the observer pattern. This makes the system more tightly coupled, making later changes more difficult. But it does eliminate a possibly complex central communication system.

Our trainer is intended to be extendable, this means we need a blackboard or other central event service. Using a blackboard means modules exchange tasks or work orders, using a publish subscribe message framework modules exchange information. So far our system has been designed in terms of how information is processed through algorithms to provide data for the next algorithm, the data is central in our design. A publish subscribe messaging framework is most suited for this goal.

Avatar control

There are multiple modules competing for control of the avatar. Sometimes their interests conflict and we need to coordinate which module can control the avatar at which time. The avatar library can execute multiple behaviors in parallel, but only if they do not require control of the same body parts (for example, a facial expression can be in parallel with an arm gesture).

Depending on the state of the exercise, there are different interaction options and different modules can have control of the avatar. Figure 4.1 and 4.2 show the program states, as visible for the user, and illustrate what interaction options there are in what state during the workout.

Interaction of the avatar with the user should be a continuous process. This means, just like in reality, behaviors and motions transform smoothly to the next behavior. The avatar has a behavior scheduler that plans behaviors ahead to achieve this. The scheduler can interrupt or replace planned behaviors if needed.

Summarizing, we have the following issues:

- Coordinating control of the avatar functions between modules.
- Different modules can have control of the avatar, depending on system state.
- Interaction is continuous: behavior needs to be scheduled before it starts and may be interrupted a later time.

The avatar has a scheduler that can plan behaviors ahead and interrupt planned behaviors. Modules will schedule behaviors and interrupt running behavior for feedback as needed. Interrupting planned behavior means, that behavior and behaviors after it need to be re-scheduled. The avatar control module keeps track of scheduled behaviors and interruptions to re-schedule the interrupted behaviors.

Since modules may or may not control the avatar, depending on the system state, modules need to know what state the program is in. If the system state changes, all interested modules are notified. Each

module is programmed with the states in which it may control the avatar, and only produces behaviors in states were it is allowed to and when it want to control the avatar. For example, the error feedback module only schedules behavior during an exercise and at most twice per minute, except during the last repetitions of an exercise.

For each state one module is defined that controls advancing to the next state. To allow behaviors to be scheduled ahead of time, the avatar control module tracks when the behaviors of the current state are going to end and announces the upcoming state change. Modules can then queue behaviors for the next state and the avatar can smoothly transition to the next behavior.

Activating modules

The modules all need to do work at different intervals, or in response to different messages. We need a mechanism that activates the right module at the right time. For the modules that only need to be activated in response to messages this is trivial, we activate them when a relevant message is sent.

The rest of the modules needs to do processing at regular intervals, for example update some statistic every second. These modules can either run in their own thread of control or be activated using a timer. Conceptually they run in parallel and we will choose a suitable implementation.

4.1.4 Modules

The system will consist of many small modules performing a part of the systems function. Below is a list of modules with a short description of their tasks, what modules they interact with and when they need to be activated. See figure 4.5 for an illustration of the modules and their interactions.

The grouping of modules into layers might be debatable. Currently, modules that only process input events and only are activated by events are in the "inputs" category. Modules in the processing stage are split into reactive and deliberative, depending on whether they do significant processing and use non-trivial algorithms to do their task.

The communication mechanism described before, a blackboard or a publish subscribe event module, is not present in the figure to avoid cluttering the image. It is used for communicating between modules nonetheless.

Low level input

Low level input modules receive input data directly from a sensor device or the operator. No further processing is done by the low level input modules, data is immediately forwarded to the input processing modules. These modules are activated by events.

- **Wii input:** reads motion data from the WiiMote controller. The WiiMote provides acceleration data on the X, Y and Z axis in m/s^2.
- **Bio input:** reads raw heart rate data from a heart-rate sensor. The heart rate sensor transmits the current heart rate once per second.
- **Operator console:** allows the operator to input mistakes the user made. The operator takes the place of a camera based input, this is called a wizard-of-oz setup. Also, the operator can input some parameters, social power and social distance, and the operator console shows status information like the mood and whether the WiiMote and heart rate sensor are connected.

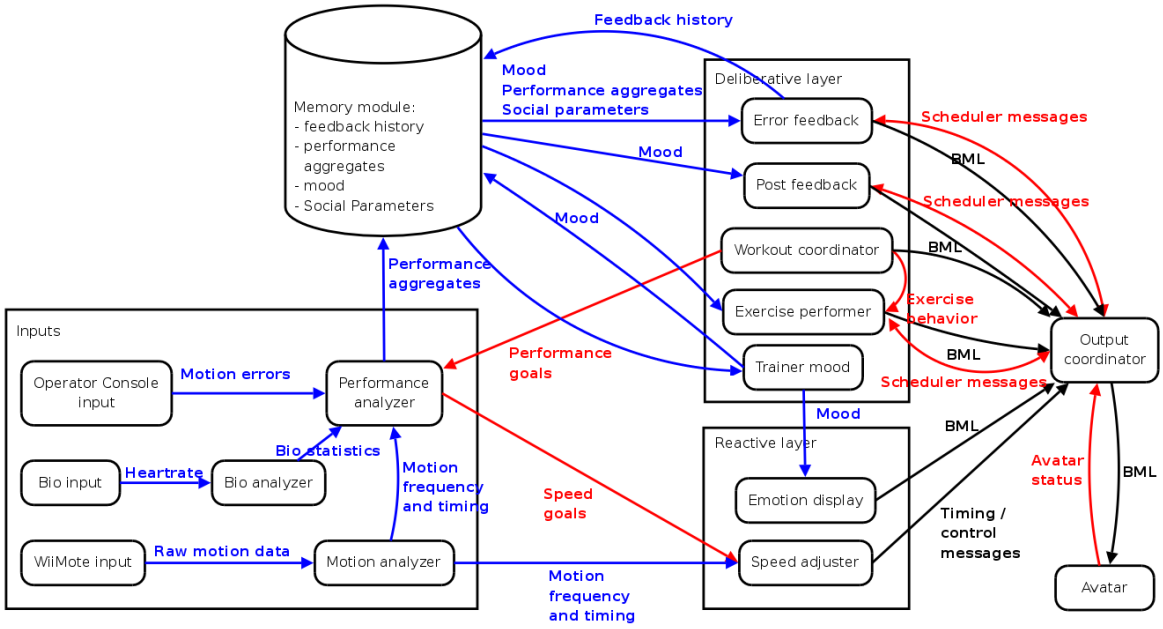


Figure 4.5: The trainer architecture. Blue lines and text represent data messages, red lines and text are control messages and black lines and text are actions. BML is a kind of scripting language for behavior of an avatar. The system state related messages are left out, as they clutter the image to much. See the description of the output coordinator to read which modules transmit and receive those messages.

Input processing modules

The input processing modules process input data into a form the rest of system can use. These modules are activated by events.

- **Motion analyzer:** processes motion data from the wii input module. Calculates frequency and timing of motions for the speed adjuster. See below for more details.
- **Performance analyzer:** processes input data from the operator console, bio and motion analyzer. All input data go through this module, which among other things calculates average number and type of errors. See below for details.

The motion analyzer cooperates with the performance analyzer and speed adjuster to implement the algorithm from 3.7.4 that adjusts the speed (and thus difficulty) of the exercise to optimally challenge the user. For this, the motion analyzer calculates a motion frequency and timing of motion extrema. The performance analyzer knows the exercise goals (how intense the exercise should be; this is set by the workout controller and can be different for every exercise) and uses the current exercise intensity to determine speed goals (slower, faster or ok), which are sent to the speed adjuster. The avatar then accepts predicted motion timing data from the speed adjuster and sets the timing of exercise motions.

The performance analyzer receives and processes all input data into a form the rest of the system can use. It implements (part of) the algorithms in 3.7.2: the average number of errors over the last 1 and 3 minutes is calculated for each type of error, also a total number of errors over the last 1 and 3 minutes is calculated. These averages are stored in the memory module.

Memory module

The memory module is used as a central data storage: Modules store data in it, which is processed further by other modules. This module is passive and does no processing of it's own. It stores the following data:

- Trainer mood. Integer number, range $[-5, 5]$.
- Average number of errors over the last minute and last three minutes, for each type of error.
- Total average number of errors over the last minute and last three minutes. These are two numbers, the average number of errors per minute over the last and last three minutes, they are totals for all types of errors.
- History of feedbacks the trainer has given. This is a list of all feedbacks the trainer has given during this workout.

See figure 4.5 for an illustration of which module stores and accesses what data. The average number of errors per type and for all errors are stored on disk, so performance in the next workout can be compared to the current workout.

Reactive layer modules

Small, algorithmically simple behavioral modules that react directly to events from the input modules. Behavior of these modules can be controlled by the deliberative layer modules, by setting goals or parameters.

The speed adjuster was already discussed briefly with the input modules. It receives current motion timings and frequency from the motion analyzer and speed goals (accelerate, slow down or ok) from the performance analyzer. This timing of motions of the avatar is then advanced or retarded or alter the users speed. Over a time of 2 seconds, the timing of the avatar is advanced/retarded to the desired setting.

Deliberative layer modules

The deliberative layer modules use more elaborate algorithms than the reactive modules and their timing is less critical.

- **Trainer mood:** determines the trainers mood. Activated at an interval.
- **Workout coordinator:** Determine, explain and start exercises; programs other modules with the exercise goals and tracks the exercise status. Activated by events.
- **Error feedback:** gives feedback on mistakes the user makes. Determines when and about what mistake to give feedback. Activated at an interval.
- **Post feedback:** gives positive comments (if applicable) after an exercise. Activated by events.
- **Exercise performer** performs the exercise with the user if needed.

The algorithm in 3.7.2 is used by the trainer mood module to calculate the trainers mood. Total number of errors over the last 1 and 3 minutes is fetched from the memory module and compared, if the number of errors increased, the mood worsens and if the number of errors decreased, the trainer becomes happier. Adjusting the mood is currently done once per minute, which worked fine.

The workout coordinator has a database of exercises, containing instructions and a demonstration, motions for the trainer, and some data about the exercise: goals (intended intensity), which muscle groups are trained, number of repetitions and duration. In the beginning, the user selects what the focus of the workout will be and the workout coordinator compiles exercises that fit the focus. The workout coordinator then proceeds to explain and start each exercise. Starting an exercise means transmitting exercise goals to the performance analyzer, giving the exercise performer the motions and broadcasting a state-change message that the exercise has started.

The exercise performer receives the exercise motions and data like number of repetitions from the workout coordinator. It executes the exercise together with the user.

The error feedback module uses the trainer mood to determine how ofter it will give feedback. We will use a linear relation, where a better mood means less feedback. The exact parameter needs to be determined by tests. To determine what feedback to give, it uses the algorithms from 3.7.3 and 3.7.1. From the memory module, the average number of errors per type and previous feedbacks are retrieved and the most important type of error is determined. The trainer mood and the selected error are then used to determine what feedback will be given. The error feedback module has a list of possible feedbacks per exercise, polite versions of those, and information about those feedbacks: inherent face threat for the type of feedback, face threat redress for the polite versions and whether the feedback is audio only or multi modal. The appropriate polite feedback is then determined and scheduled for output.

The post feedback gives supportive or positive comments after an exercise. It uses the trainer mood to determine what comment is given. It has a list of possible feedbacks, and for each feedback it has a mood when this feedback can be given.

Output coordinator

The output coordinator determines which module controls the speech and motions functions of the avatar, tracks when behaviors are finished and allows modules to schedule new behaviors. See 4.1.3 for a background of the algorithm. Activated only by events.

The output coordinator uses the avatar libraries scheduler to plan behaviors ahead and facilitate smooth transitions. Modules send behavior (BML scripts) through the output coordinator and indicate whether the behavior is exclusive, can run in parallel and if it needs to interrupt previously planned behaviors. If behavior is interrupted, all future behaviors are cancelled. The output coordinator broadcasts a message so modules can re-schedule the cancelled behaviors.

All modules know the workout state and know when they may be activated, they only schedule new behaviors when they are allowed to take control of the avatar. Some modules have more heuristics, the error feedback only gives feedback once every (half) minute depending on the current performance.

For each state we define a module that is responsible for advancing the system to the next state. When that modules want to advance the system state, it first broadcasts a heads-up message for the upcoming state change. This allows modules to queue behavior for the next state, making for smoother transitions. When an upcoming state-change is announced, no more behavior is scheduled for the current state.

The output coordinator also signals modules when a scheduled behavior starts, finishes or is interrupted. If a behavior is interrupted, a module can for example re-schedule it. The module currently responsible for advancing the system state can announce the upcoming state change, if that modules last behavior of the current state starts.

If a behavior is interrupted while running, that behavior is cancelled. For technical reasons, it is not possible to pause and resume a behavior at a random point. This means that behavior start all over. This means it might be practical to cut behaviors that can be interrupted into smaller pieces, for example schedule each repetition of an exercise as a separate behavior.

In figures 4.1 and 4.2 we showed the states, transitions and interactions as the user sees them. The states as the user observes them, are not necessarily the most practical set of states to use while imple-

menting the software. The states in the program will be used so modules know when they may control the avatar. This gives us other states that what the user observes. Fewer states could be used if modules use additional heuristics to determine when they can be activated, for example the previous state or number of the current repetition. Using states to distinguish all situations where different sets of modules can be active makes it clearer what can happen and helps making the trainers behavior more consistent.

- Present workouts
- Explain exercise
- Exercise starting
- Perform exercise
- Exercise ending
- Exercise done

In the "present workouts" state the user can choose which workout he wants to do today. After the user has chosen a workout, the "explain workout" state is entered and the trainer explains the next exercise. The "exercise starting" state is for the first few repetitions of each exercise, letting the user get used to the exercise and not immediately bombarding him with mistakes. The "perform exercise" state starts after a few repetitions of the exercise, here the user gets feedback on his mistakes. When the exercise is almost done the "exercise ending" state is entered and there is no more feedback. After an exercise the system enters the "exercise done" state and gives positive or supportive comments depending on the performance. After this, or immediately if there is no need for positive or supportive comments, the system goes back into the "explain exercise" state.

Table 4.1 lists all modules that can control the avatar, all states of the system and what action each module can do in that state.

	Error feedback	Post feedback	Workout coordinator	Speed adjuster	Exercise performer
Present workouts			1 (A V)		
Explain exercise			2 (A V)		
Exercise starting					3(V)
Perform exercise	4 (A or AV)			6 (P)	3(V)
Exercise ending				6 (P)	3(V)
Exercise done		5 (A)			

Table 4.1: States, modules and actions each module can do in each state. (1) Present available workouts and let the user choose. (2) Explain an exercise and demonstrate how to perform it. (3) Do the exercise motions together with the user. (4) Give feedback on a mistake the user made, for example tell him to move faster or explain corrected motions. (5) Give supportive or positive comments after the exercise. (6) Adjust the speed of the exercise. (7) Display the trainers mood using facial expressions. The symbols between brackets denote the type of output used: A and V denote audible or visual output, P means the action is parallel and does not need exclusive control of the avatar.

In the present workouts and explain exercise states, the workout coordinator is responsible for advancing the system state. In the exercise starting, ending and perform exercise states, the exercise performer is responsible. In the exercise done state, the post feedback module is responsible.

4.1.5 Messages and interactions

This section discusses the messages exchanged throughout the system, their intended function and shows some examples of how they are exchanged between the modules.

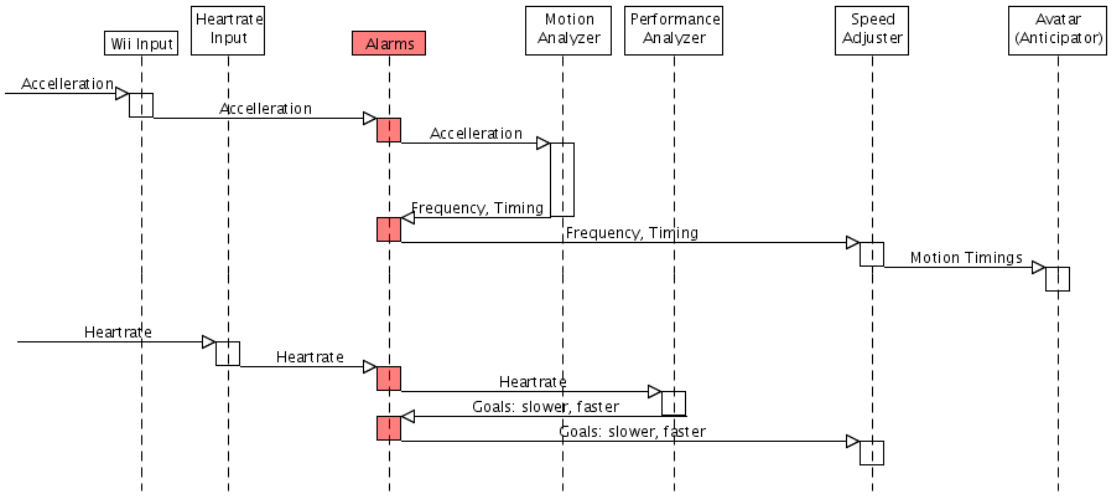


Figure 4.6: Interaction between the modules, when adjusting the exercise speed.

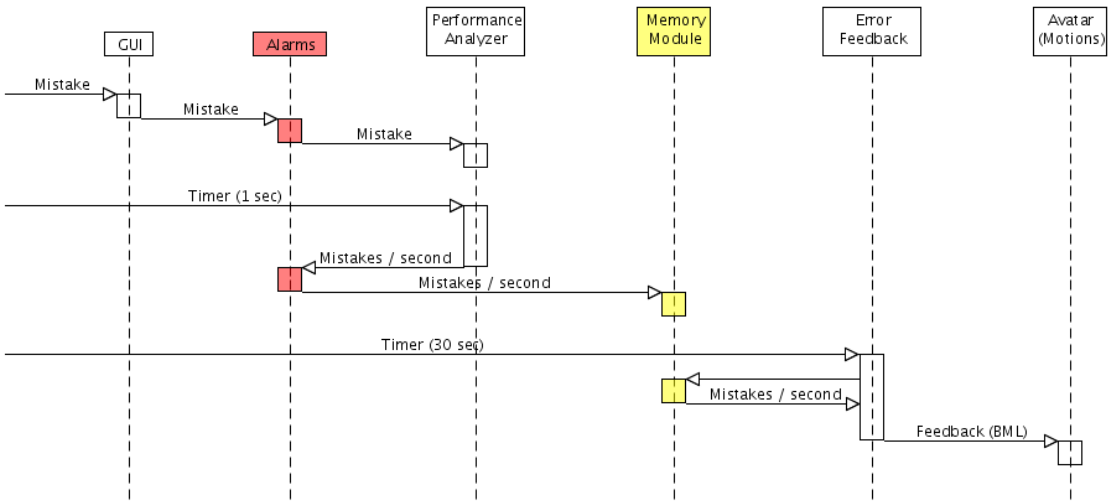


Figure 4.7: Interaction between the modules, when giving feedback on a mistake.

Figure 4.5, the system architecture, showed all modules in the system and the data being exchanged between those modules. Figures 4.6 and 4.7 elaborate two examples of what data is exchanged between the modules when adjusting the speed to the users performance and what data is exchanged when giving feedback on a mistake.

The data exchanged between the modules in packaged in messages. Below is a list of all messages exchanged in the system and their function.

- **Input-Wii** new data received from a WiiMote controller.
- **Input-Motion** motion data, contains frequency and extrema timings.
- **Input-Heartrate** heart rate data.
- **Input-Mistake** a mistake was entered by the operator, contains type of mistake.
- **Input-Social** social parameters were altered by the operator.

- **State-Change** the system state has changed, contains the new state.
- **State-HeadsUp** the system state is about to change, contains the new state. Behavior is now scheduled for the next state.

- **Goal-Performance** Informs the performance analyzer of the goals for this exercise (Intended heart rate).
- **Goal-Speed** Informs the speed adjuster of the intended speed.
- **Goal-Exercise** Informs the exercise performer about the new exercise. Which motions and number of repetitions.

- **BML-Start** A scheduled item was started.
- **BML-Cancel** A scheduled item was interrupted (cancelled). The module can reschedule it.
- **BML-Done** A scheduled item was completed.

- **Control-Start** Start the workout.
- **Control-Pause** Pause the workout.
- **Control-Stop** Stop the workout.

4.2 Implementation notes

This section discusses some aspects of the implementation. For the most part the original design worked well, but it had to be altered in a few areas to better work with external software libraries. Mainly the avatar animation library had some quite specific requirements.

4.2.1 Bluetooth driver framework

Our input devices, the *HxM Zephyr Heartrate Monitor* and the *Nintendo Wii Remote*, both use bluetooth to connect to a workstation. We have developed a small framework that handles scanning for bluetooth devices and contains drivers for the two input devices we use. Additionally we implemented the option to record and replay input data. This can for example be used to record an exercise once and then use the recorded data during development work.

4.2.2 Threads

There are multiple active threads in our application, some care needs to be taken so that the threads do not conflict. Since modules only communicate using messages, that is the only place we need to protect threads from each other.

The input modules have separate threads for device input that send messages, the GUI also sends messages from his own thread. Luckily there are few modules that receive messages from different threads: mainly the performance analyzer and the speed adjuster. We added some special safeguards there.

The output modules communicates with the avatar animation library. That library needs special attention when being accessed from multiple threads. It does offer an interface that queues requests to the library and performs the actions in one thread, for cases where the library is used in a multi-threaded environment.

4.2.3 Common base classes

There are several areas where code can be shared between the modules. For example, all modules that generate behavior (BML) need to track queued behaviors and re-schedule them if they are cancelled.

Similarly we have created base classes for other areas: for example all behaviors share common functionality.

4.2.4 Event or timer based?

In the initial design, many modules only responded to messages. After each received message calculations are done, average values updated etc. This works well for data that are updated frequency, but for example errors might only be generated once or twice per workout. In the time between their time-averages are incorrect.

During implementation this issue was discovered and time-averages are now generated on a timer basis. Error averages are generated every 30 seconds, frequency and predicted motions are generated twice per second.

4.2.5 Motion analyzer

We have tried several approaches to determine the users frequency of motions from the accelerometer data the WiiMote controller gives. As the orientation of the WiiMote controller is unknown and frequently changes, we chose to use only the length of the acceleration vector for further calculations.

First attempt: simple state machine

The first approach was based upon the assumption that the WiiMote would be still at turn points of motions and moving in between. This should allow us to determine the turn points of motions based on whether the WiiMote is still or moving.

Implementation was simple enough, track the current state (Still or Moving) and keep a list of states from the last 10 seconds. The average number of still states per second should be the frequency of the motions.

This approach did not give good results, the calculated speed was very unreliable. Constant, fluent motions accelerate very gently and the WiiMote can appear to be still even if it is moving. Hectic motions are always accelerating, they appear to never come to a still point. With some experimentation it was possible to move the WiiMote such that the frequency could be calculated, however this is not practical for experiments with users.

Second attempt: using frequency spectrum analysis

The second idea was to use the repetitive nature of the exercises. In the frequency spectrum this should show as high amplitudes for the most evident frequencies. The frequency with the highest amplitude should be the frequency at which the user performs the exercise.

While this should work well in theory, there are some problems. Only a short period (10 seconds) of input data is analyzed, the fast Fourier algorithm we used divides the frequency spectrum in slices and gives total values for each slice, and the input data is noisy to the point where it can be hard to interpret manually.

For "nice" input data (for example, hanging the WiiMote from a cord and letting it swing) this did give good results. For input data from actually performing an exercise, the outcome varied. For example, changes in speed mean there are several frequencies with high amplitudes. Figure 4.8 show input data and a frequency analysis.

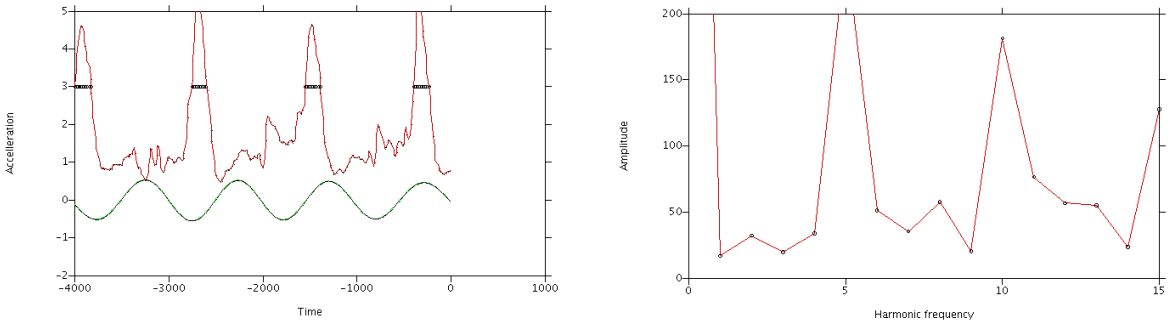


Figure 4.8: Charts of accellerometer input data and the frequency analysis of that data. The frequency with the highest amplitude is drawn in the chart of the input data. The peaks happen at turn points of motions. The markings indicate where the acceleration is above 3g.

Third attempt: revised state machine

The idea for the third approach came from a visual analysis of accellerometer data. As visible in figure 4.8 there are pronounced peaks in the signal. These peaks happen at turn points of motions. The abrupt change in direction produces a short high acceleration. We defined two states: Peak, Normal. The peaks are defined as consecutive acellerometer inputs over 3g. The time between the peaks gives us the frequency of the motions.

This approach actually performed acceptable for input data from performing an exercise. Very slow motions produce lower acceleration peaks, which are not detected by the algorithm. Very hectic motions means there are more peaks, for example a "normal" motion would produce one peak (at the upper turn point) per repetition of an exercise, faster motions can produce a peak at the upper and lower turn point giving two peaks per repetition. Hectic motions can produce many peaks and give unreliable results.

Initally, we wanted to use the measured speed as the basis for the speed of the animation: Let the trainer move at 110% or 90% of the users speed. Measurement errors resulted in errors in the animations of the trainer. For example, the trainer would suddenly move much too fast. Therefore we calculated the average frequency over the last few seconds, which smoothes out measurement errors.

During the evaluations, the trainer application merely logged the measured frequency but did not use it as the basis for animating the trainer, so calculating the average frequency was not necessary.

4.2.6 Speed Adjustment: Synchronization points and anticipators

To adjust the intensity of an exercise, the virtual trainer speeds exercises up or slows them down. The Elckerlyc avatar animation system has a nice feature for this, called synchronization points. One or more frames in an animation of the avatar can be marked as synchronization points. These synchronization points can be assigned fixed time values, or the time values can be changed dynamically during execution using an anticipator. An anticipator is a container for synchronization points and includes some functions to change timing or animations. The recorded motions are played faster or slower, such that the syncpoints defined in the animation are reached at the times specified in the anticipator. This feature can be used for different applications, starting a motion at a specified future time or to play recorded motions at a different speed.

(a) Start (c) Jump (e) End

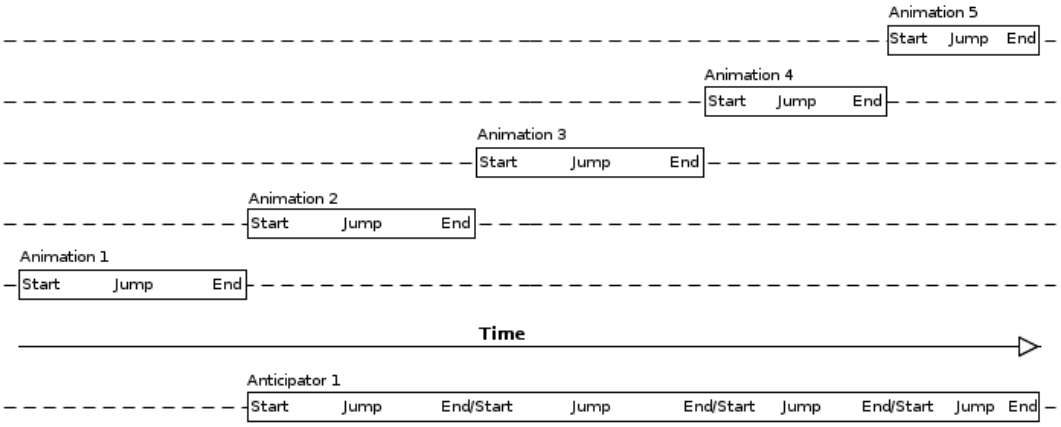


Figure 4.9: Synchronization points moving closer together as the trainer speeds up the exercise. The first animation is played at a fixed speed, the rest are coupled to an anticipator. An anticipator has a list of syncpoints and several functions to change the time in between the syncpoints.

For the usage in our trainer prototype, we have the following requirements:

- The trainer needs to change the speed of animations, play them faster or slower.
- It is not precisely know up front at which time animations will begin: they simply start after the explanations of an exercise. We need to move the synchronization points to the right time once that is known.
- Since the trainer can do different exercises, we need to generate the synchronization points as needed.

In the trainer prototype, we use the anticipator as an anchor for synchronization points, the timing of the synchronization points is relative to the start time of the anticipator. The first (few) animation

cycles of an exercise are played at a fixed speed, when they start, we calculate the start time for the animations with synchronization points and update the anticipators start time.

To change the speed of an exercise, the synchronization points are then moved closer together or further apart.

Chapter 5

Evaluation

In the previous chapter several abstract goals were chosen for the trainer, the requirements were developed and a prototype was designed to fulfill those goals and requirements. In this chapter an evaluation method is developed to determine if and how much the prototype meets the previously chosen goals.

We defined the goals of the trainer application as supporting the user in training to achieve his workout goals. Like a real sports coach, it helps plan workouts, gives technical assistance and tries to motivate the user to keep working hard. We designed and implemented new approaches for the last two aspects. So this gives us two aspects to evaluate, user motivation and assistance during the workout. We discuss evaluation of each aspect separately, to develop evaluations that best suit each of these aspects.

Before we design our own evaluation, we discuss evaluating interactive embodied agents a little bit.

5.1 Background

Dehn and van Mulken [2000] discuss evaluating embodied agents and give an overview of how various projects evaluated their agents, what their results were and discuss several more or less obvious mistakes that have been made in those projects. A common mistake is attributing improved performance of an agent to a specific change, while it is in fact ambiguous what change improved that performance.

Dooley [2001] discuss social research methods, in particular how to design experiments and do evaluations. The most important aspects of social research are reliability and validity. Reliability means the degree to which a consistent aspect is measured, rather than random error. Validity means the things that was actually measured is also what was intended to be measured.

For a reliable experiment, the test itself must be reliable and environmental factors need to be consistent. Reliability of a test can be influenced by many things, clarity and ambiguousness of questions, standardizing of instructions and question format or judgement of observers that rate participants. Asking a question multiple times with different phrasings reduces the effect of random errors.

When designing a questionnaire for an experiment, there are several issues concerning reliability to look out for:

- **Item wording** can affect outcome by means of ambiguity, combined questions, limited answering options, question bias or sensitive topics.
- **Questionnaire length** is important as asking more questions can improve reliability, but patience and interest of subjects limits the length.
- **Ordering effects** means the influence earlier items can have on later questions. Best practice is to ask general questions before specific questions, to start with the least threatening and most interesting items and to cluster questions by topic, instructions or question format.

- **Format and response coding.** A questionnaire should be uncluttered and well readable, the flow through the questions needs to be clear and the instructions need to be clear. For open questions, categorizing needs to be standardized and with much care.

Validity concerns what was measured and what conclusions may be drawn. Often an effect is measured rather than a cause. Drawing conclusions must be done carefully because the meaning of collected data can be non-trivial. Careful design of the experiment beforehand helps avoid errors.

Validity errors can occur during the experiment. The results can be contaminated by intentions and beliefs the experimenter or the subject have about the experiment.

When the experimenters unintentionally influences the experiment, this is called experimenter expectancy or the self-fulfilling prophecy. Especially in cases where the experimenter is also the observer this can bias the results. But also how the experimenter gives instructions or treats subjects during an experiment can leave hints and cues that bias results.

If the subjects knows (or thinks he knows) the goals of an experiment this can influence results, this effect is called demand characteristics. What the subject knows about the experiment can come from instructions, questions or might develop during the experiment. The placebo effect falls in this category. There are various ways to prevent subjects from learning the goals of the experiment or minimize unwanted effects.

5.2 Evaluating speed adjustment

We have implemented one mechanism for the trainer to technically assist the user during the workout: the trainer can try to influence the users rhythm. A faster rhythm gives a more intense workout, a slower rhythm is less intense. For an exercise a goal-range for the heart rate can be specified and the trainer then tries to keep the user within that range.

We will evaluate two aspects of this system: does it function as intended and is it intuitive to the user. Functioning means, can we influence the users speed. Being intuitive means, do users need instructions or do they behave as intended automatically.

To test whether we can influence the users speed users will perform two exercise with the trainer. First we want to test if our basic approach of influencing speed works, then we will determine if our algorithm can keep the users heart rate within a defined range. Both exercises will start with a warming up, and then continue from there. During the first exercise the trainer tries to increase the heart rate. During the second exercise, the trainer increases or decreases speed to keep the user in the goal heart rate range.

We will record performance data during the exercises: heart rate, user speed and current state of the trainers speed influencing algorithm. From the recorded performance data we can then determine if the trainer can influence the user.

Determining if the system is intuitive means finding out if the user picks up the trainer speed-influencing intentions without further instructions. This means, we can not give explicit instructions to the users, in particular users should not be told that the trainer will try to influence their speed.

We will tell the user to follow the rhythm of an audio signal, for example a beep or a spoken word "Jump". This gives a rhythm to the user that is linked to the motions, without explicitly asking to follow the motions. The first 10 repetitions the audio signal will be played in synch with the trainers motions, then the audio signal stops and only the trainers motions indicate the rhythm. After another 10 repetitions, the trainer will start to influence the users speed.

5.3 Evaluating motivation

We have implemented several mechanisms in the trainer to enhance student motivation during a workout session. We want to evaluate whether these mechanisms do indeed enhance motivation.

Because subjects might perform with varying success, an interactive evaluation of motivation (where every participant performs a workout with the trainer) might give a different experience for each subject and thus the results would not be comparable. Additionally the prototype requires an observer to input mistakes, another potential for unreliability. To solve this, we will record exercises and let participants rate motivational factors after viewing the recordings.

Because motivation is not an easily measurable quantity, users can not rate motivational factors directly, but they will need to compare motivational factors between different scenarios. This means a baseline will be used against which scenarios with different motivational settings are compared.

An obvious choice as baseline for comparison would be the trainer prototype with the motivational mechanisms disabled, where the trainer simply reports errors to the user.

The goal of this evaluation is to determine whether the motivational mechanisms function as intended and can indeed enhance motivation and improve the training experience. To improve the validity of results we will compare multiple setups with the motivational mechanisms with the baseline. If we do only one comparison of a trainer with motivational techniques versus the baseline, users might for example dislike the specific setup of the trainer and the results would suggest the motivational techniques are not suitable. Comparing multiple different setups should give more valid results.

We will choose two types of trainer, tune the interaction to suit the type of trainer and evaluate those against the baseline setup.

The interaction and motivation mechanisms of the trainer can be tuned to alter the behavior. The trainers base mood can be tuned from bad to good, altering the amount of politeness (face threat redress) the trainer uses. The trainer can be configured to give motivational comments during or after an exercise, or even scold the user after a particularly bad performance. Feedback can be given as pure verbal comments, or if that is not effective the workout can be interrupted for further instructions. The frequency of error reporting can be changed.

The following three seem reasonable choices:

- Friendly, polite trainer setup: good base mood, motivational comments during exercise, infrequent feedback.
- Rude, impatient trainer: bad base mood, no motivational comments, frequent feedback and interrupts exercise to give additional instructions.
- Baseline trainer: motivational mechanisms disabled: just reports errors as they occur.

We will record performing the exercise and the trainer feedback separately. If they were recorded together, our opinion about this version of the trainer could influence behavior and influence the evaluation users give. The recordings of different trainer versions will be played in random order. Most users do not blindly participate in the experiment, but form a mental model about the evaluation which influences their answers. Playing the recordings in random order should lessen this effect.

As one exercise is quite short, it is difficult to properly show the differences between the trainer versions in one exercise. Additionally, participants might experience the trainer versions differently depending on the exercise. For these reasons we use a short workout of several different exercises to demonstrate the differences between the trainer versions. The workout will be reasonably short, to avoid boring the participants, but long enough that the differences between setups become clear. Please see appendix C for the exact workout used.

All instructions will be included with the evaluation forms or written down separately: Oral instructions could differ per participant.

We will record some data about the participants, age, sex and experience with group sports.

We will use a questionnaire to record what participants thought about the trainer prototype. There are four general topics we are interested in: workout experience, which we divide into motivational factors and assistance factors, and finally the character traits of the trainer.

The original goal was to motivate and assist a user during training, we test that goal with the questions about training experience. Autonomy, competency and relatedness directly influence motivation, according to literature. The assistance factors cover whether the trainer assisted properly with the execution of exercises. Since we designed the coaches to have different character traits, we want to know whether users noticed those character traits.

Please view appendix D for a copy of the questionnaire.

Chapter 6

Results

6.1 Speed influencing

After the prototype application was fully functional, we did a series of pre-tests to prepare for the evaluation. This showed us some weak areas of the trainer that needed to be addressed before we could start the evaluation.

After the discussion of pre-tests we divided the evaluation into three parts, as mentioned in the evaluation design: can we influence the users speed, can we use the speed influencing to get and keep the heart rate within a range and is this intuitive to use.

The participants were about half male and half female. Most did not have experience with aerobics specifically, but did have experience with other sports or e.g. dancing.

6.1.1 Pretests

The first approach to influencing the users speed was to let the trainer move half a second before the user. This did not work well because the motion analyzer took too long to pick up speed changes (It uses the average speed over the last 10 seconds). Users would speed up, the trainer would not follow immediately and users slowed down before the trainer had picked up the new speed. Additionally, measurement errors meant the trainer could suddenly start moving at absurd speeds, which confused users and made it hard to do an exercise with the trainer.

Our second try was to move 10% faster than the user. This somewhat solved the problem of the trainers speed lagging behind, but did not solve the issue with measurement errors.

To work around the problem with measurement errors, we chose a base speed for the trainer and sped up or slowed down every few seconds. The user is no longer the leader, the trainer dictates the speed.

Little experimentation showed that it was possible to do the exercise (Jumping Jacks) at speeds up to 1.5 Hz, then it became too fast. To get 1 minute of measurements with a final speed of 1.5 Hz means we can increase the speed by 5% about 10 times. We started the exercise with 5 repetitions at a fixed speed of 0.8 Hz and the increase the speed by 5% every 5 seconds.

6.1.2 Can we do it?

For the first real test we had 3 subjects. They performed 50 repetitions of Jumping Jacks with the trainer. The motion analyzer used a 10 second average. The first 5 repetitions where at a speed of 0.8 Hz and then increased by 5 % every 5 seconds. The tests where captured on camera to review the results.

See appendix E for all charts of the first tests. The measurements look rather chaotic (figure 6.1) and it is not clear if the speed increased. Reviewing the videos shows that users lost the pace, made hectic speed changes to regain the pace, only to lose the pace again. However, their speed did increase.

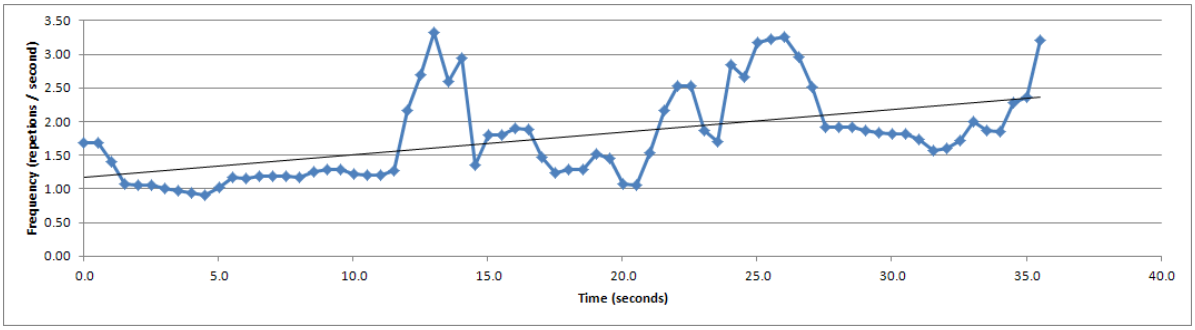


Figure 6.1: The measurement is quite chaotic because the user lost the pace and made hectic speed changes to regain the pace.

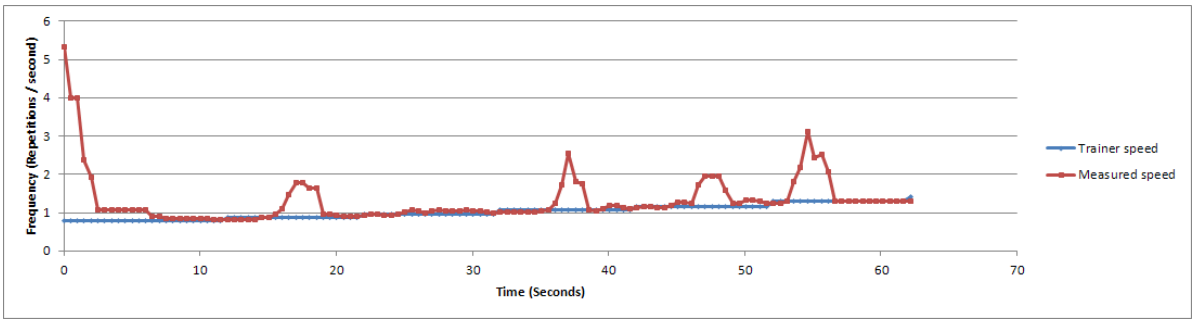


Figure 6.2: Well readable speed measurement.

The second measurements used 10 repetitions at a fixed speed and then increased by 10% every 10 seconds. The motion analyzer calculated a 2.5 second average to let measurement errors have a lower impact on results. We used 2 subjects.

Appendix E has all the charts for this second measurement, they show the results improved a lot using this method. (See figure 6.2) The users where able to follow the trainers increase in speed very well. The second chart in the appendix shows an interesting effect, as users move faster the accellerometer signal changes it's pattern and has two peaks per repetition instead of one: the measured frequency appears doubled.

Summarizing, it is indeed possible to influence the users speed.

6.1.3 Keeping the heart rate within range

For this test there were 6 participants. They performed 50 repetitions of Side Steps. The first 10 repetitions were at a constant speed of 0.6 Hz, then the speed increased or decreased 10% every 10 seconds depending on the performance. If the heart rate was below the desired range the speed increased, if the heart rate was above the speed decreased.

We used side steps because it is a less intense exercise, a more intense exercise might require a warming up to get the heart rate into the desired range. With a more intense exercise, the trainer would be moving very fast before the users heart rate is at the warmed up level.

Please see appendix F for charts of the speed and heart rate of users. We made the goal heart rate adjustable, because the exercises intensity varied greatly per user. Figure 6.3 shows an example.

The trainer speeding up and slowing down does influence the users heartrate, but the exercise produced a very high heartrate in some users and barely raised the heartrate for other users. In two cases it worked as expected. Just speeding up the exercise will not get all users into a defined range: they can not keep

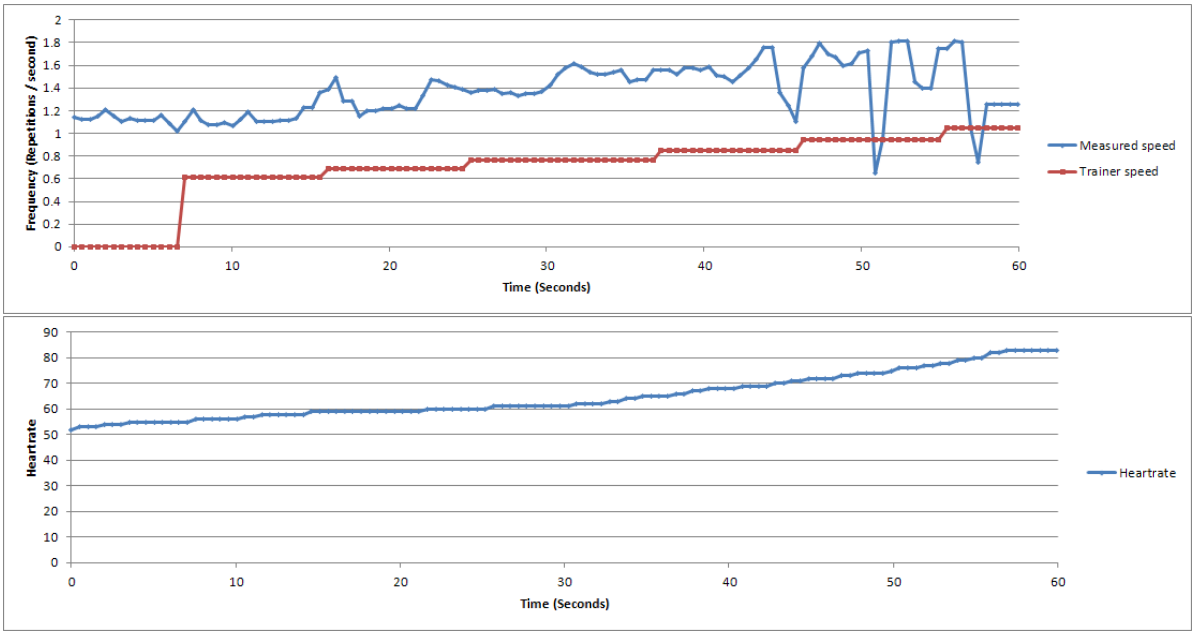


Figure 6.3: Trainer and measured speed, heartrate. The trainer can influence the heartrate, but the exercise is not intense enough for the user.

up with the trainers speed but the exercise is still not very intense for them.

6.1.4 Intuitiveness

For this test we had 6 participants. They performed 50 repetitions of Jumping Jacks. The trainer started with 8 repetitions at a constant speed where he said "Jump" every time, then 8 more repetitions at a constant speed without comment. After this the speed increased 5% every 5 seconds.

As discussed in the evaluation design, this experiment tested the intuitiveness of influencing the users speed. All instructions for the test where spoken by the trainer application, and avoided instructions to follow the trainers speed. Some of the experiments where recorded on camera.

Please see appendix G for all charts of the measured and trainers speed. Figure 6.4 shows an example measurement.

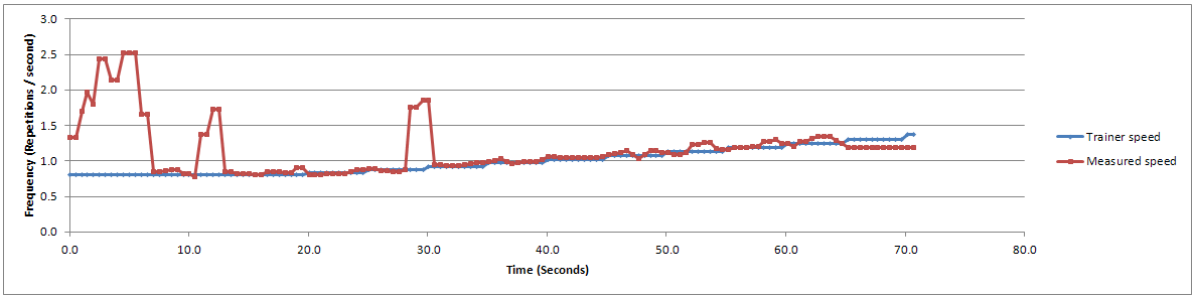


Figure 6.4: The user follows the trainers speed increase very well, even without explicit instructions.

The speed influencing worked intuitively for all but one users, the one user stopped speeding up halfway through the exercise. Why is unclear. For one user the measured speed is chaotic/wrong, but the video recording showed he did indeed speed up.

Several users remarked "The trainer keeps going faster", without prior knowledge that was going to happen.

6.1.5 Review of the measurement method

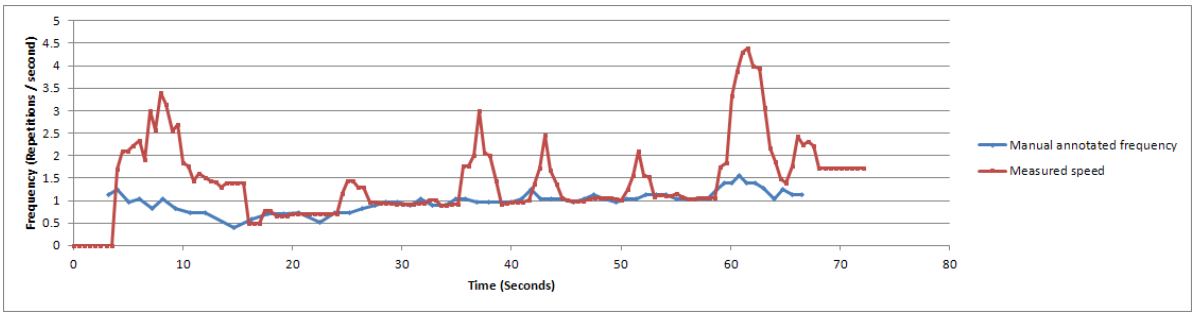


Figure 6.5: The chart shows the measured frequency as our motion analyzer module produces it and the manually annotated frequency of motions from the video recording.

To check the correctness of our measurement algorithm, we made a video of a user performing the exercise with the trainer application and manually marked each repetition of the exercise. Figure 6.5 shows the results.

Most measurements are correct, but the sudden peaks in the measured frequency are errors. For example if the user makes a brief hectic movement or if the WiiMote accidentally hits the users leg, these measurement errors can occur.

Fluid motions produce the best results. Users familiar with aerobics exercise (mostly girls) or users that are good dancers (For example break dancers) give much better measurements than user users. Please see the results section for examples of charts, some measurements are chaotic and other much better readable.

6.2 Motivational mechanisms

The setup of this evaluation is as discussed in the evaluation design. We prepared three videos that showed how the trainer versions responded to a execution of workout. (See appendix C) We marked the videos A, B and C so users did not know which video was what trainer. Users were given the same instructions: "We will show you three videos of different virtual aerobics trainers. A user performs a workout with each trainer and gets different feedback from each trainer. Please imagine you are doing the workout and how you would experience doing a workout with that trainer." The videos were played in random order and users filled in one page of the questionnaire after each video. After the users had seen all videos, we asked them to write down additional comments, e.g. how they would describe the different trainer versions and anything else worth noting.

There were 16 participants that filled in the questionnaire, all male and with ages from 20 to 35. Of the 16 participants, 2 had no experience with sports training supervised by a coach, 8 had little experience (1 or 2 sports, less than 3 years) and 5 had a lot of experience (multiple sports, more than 3 years).

Please view appendix H for the results of the questionnaire and written comments of users. Appendix D has the questionnaire as given to users.

We start with an explanation of the statistical methods we use to examine the results. We have split the discussion of the results into 5 parts, four for the topics in the questionnaire and one for additional comments of the users.

6.2.1 Methodology

We have analyzed the data from the questionnaire with statistical procedures. These statistical procedures make some assumptions about the data. We will start with a discussion of these assumptions.

The scale we used for measuring the respondents answers is a 5-point likert scale, for example described in DeVellis [2003]. We have treated these data as interval data [Dooley, 2001] for the statistical tests.

We showed each user three videos and let them fill out three questionnaires, this type of experimental design is called dependant or a repeated measures design.

The statistical methods we plan to use further assume normally distributed data. Since we have a rather small group of 16 respondents, the distribution of answers does not always look like a normal distribution. The normality of data can be tested, for example with a "Goodness of fit" test like the Pearson Chi Square test. Boneau [1960] discuss what effect violations of the normality of data have on the t test and the F test. The t test and F test are pretty robust and they come up with some pretty useable conclusions: the size of groups being compared should be roughly equal and the distribution of data should not have major visible artefacts.

A chart with the distribution of answers shows us if there are visible artefacts. Where artefacts are present we discuss these. Charts of the average and standard deviation of each question illustrate the results of the statistical tests.

We have also compared data with a different method: for each question we ranked the trainer versions per user and counted how often each trainer version was ranked best or worst. This should give the same results as the statistical tests.

Since we want to compare three trainer versions we will start with a repeated measures F test. This test can tell us whether the three trainer versions are rated differently, but not in what way. The advantage of this test is, it can compare 2 and more statistical variables at once. This lowers the chance of false conclusions, because each statistical test has a inherent error margin. [Zar, 1999]

To see what the actual difference between each pair of trainer versions is, the dependant t test is used. Because each statistical test has an inherent error margin (we chose 5%) and we are doing 3 tests, we need to correct for this. This is called a Bonferroni correction, the intended error margin is divided by the number of experiments to ensure the desired error level is met.

As a sort of integrity check of the questionnaire, we verify the answers users gave match the design of the questions. In the questions about motivation, we asked for the users level of motivation ("I felt motivated") and for contributing factors according to the psychological model. This means, there should be a high correlation between motivation and the contributing factors. Similarly, some questions were synonymous or opposite in meaning to other questions. Here we also expect a high (positive respectively negative) correlation.

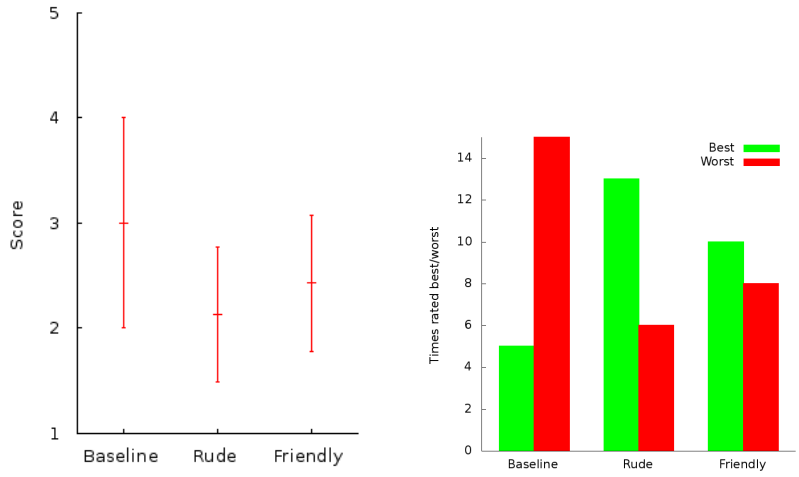
6.2.2 Training experience

The questions in the "Training experience" category are general questions about topics we felt contribute to a good training experience. The following sections go more into the detail of some of these topics.

In the design of the evaluation we wrote *"The goal of this evaluation is to determine whether the motivational mechanisms function as intended and can indeed enhance motivation and improve the training experience."* We will try to answer that from the questionnaire results.

There are significant differences (See table 6.1) only for the last question about training experience. Figure 6.6 shows the statistics (average, standard deviation) and ranking of the trainer versions for the question "The training was assisted well". The rude and friendly trainer scored significantly better than the baseline trainer. The scores for the rude and friendly trainer are not significantly different.

If we look at the distribution of the responses to "The training was motivating" in figure 6.7, we see something interesting. The distribution for the friendly trainer looks like a proper normal distribution, the



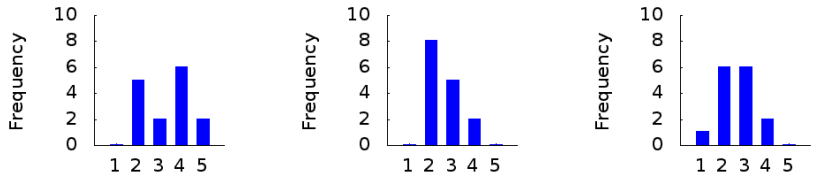
(a) "The training was assisted well" (b) "The training was assisted well"

Figure 6.6: Statistics (average, standard deviation) and ranking (best, worst) of trainer versions.

Questions	**F Test**	**T tests** **Baseline vs Rude**	**Baseline vs Friendly**	**Rude vs Friendly**
"The training was effective"	0.15	-	-	-
"The training was motivating"	0.06	-	-	-
"The training was assisted well"	0.0	0.0	0.02	0.26

Table 6.1: Results of statistical tests for the training experience questions. T tests are blank where the F test is not significant. For the F test a score below 0.05 means a significant difference, for the t tests we consider the difference significant if the score is below 0.02.

rude trainer is a little skewed but that might be because we had only 16 respondents. The responses for the baseline trainer however do look not like a normal distribution. It looks like half the people consider the training motivating and the other half disagrees. However, the Pearson Chi Squared normality test indicates the data is normally distributed (But with a rather low chance of $P = 0.07$).



(a) Baseline (b) Rude (c) Friendly

Figure 6.7: Distribution of responses to "The training was motivating" (1=Totally agree, 2=Agree, 3=Neutral, 4=Disagree, 5=Totally disagree)

6.2.3 Motivational factors

The questions about motivational factors cover the three basic components of motivation: autonomy, competence and relatedness. High scores for these factors should contribute to a higher motivation. (See section 3.1 for a background on this model of motivation)

We asked two synonymous questions about each of these topics, also the general question "I felt motivated" was asked in a different phrasing in the training experience category. Table 6.2 shows correlation between the synonymously phrased questions. For the questions about competency the correlation is quite low, so for example participants perceived a different meaning for the two questions or there is a large random factor involved.

Questions	**Correlation**
"The training was motivating" and "I felt motivated"	0.78 ($sig = 0.00$)
"I felt in control" and "I was in charge"	0.79 ($sig = 0.00$)
"I felt competent" and "I performed well"	0.33 ($sig = 0.03$)
"I felt close to the trainer" and "I got along with the trainer"	0.65 ($sig = 0.00$)

Table 6.2: Correlation between synonymously phrased questions. A significance below 0.05 means a valid correlation.

Question	**Correlation**
"I felt in control"	0.30 ($sig = 0.05$)
"I was in charge"	0.08 ($sig = 0.60$)
"I felt competent"	0.50 ($sig = 0.00$)
"I performed well"	0.36 ($sig = 0.02$)
"I felt close to the trainer"	0.62 ($sig = 0.00$)
"I got along with the trainer"	0.62 ($sig = 0.00$)

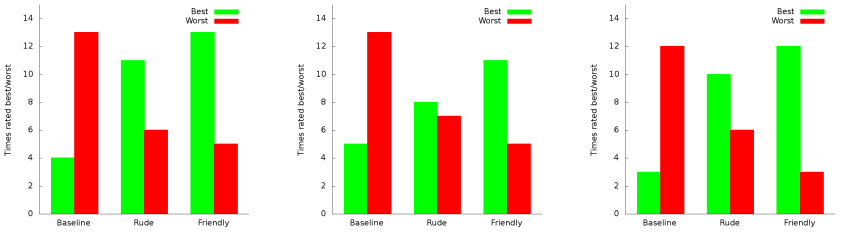
Table 6.3: Correlation between motivation ("I felt motivated") and contributing factors, the correlation is valid with a significance below 0.05

Table 6.3 displays the correlation between perceived motivation (Question "I felt motivated") and the questions about contributing factors according to our motivational model. For all contributing factors there is a positive correlation, but for some that correlation is very low, nonexistent or not significant. For the questions with a high correlation (Let's say 0.5 and higher), we can say that a good result on that contributing factor correlates with a higher perceived motivation.

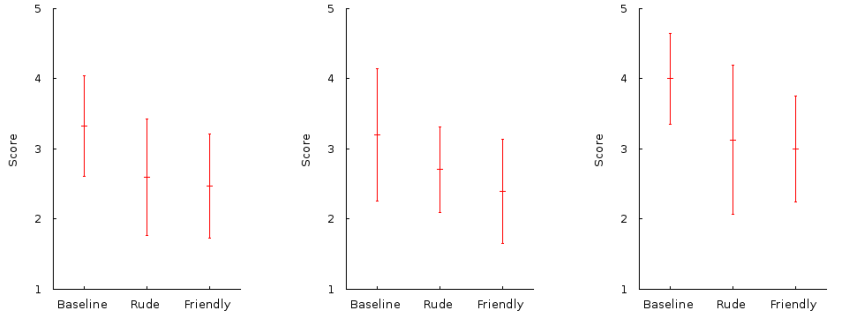
For the two questions about autonomy ("I felt in control" and "I was in charge") there is no significant difference between the trainer versions. This is not surprising, each trainer version is the same in choosing and leading the exercises. For all other trainer versions there are significant differences between the trainer versions (See appendix H)

Figure 6.8 shows statistics and rankings of the trainer versions for some of the questions. The baseline trainer scores worst, the rude and friendly trainer score similar and better.

It is interesting that there appears to be a difference between the rude and friendly trainer for the questions about competency and relatedness, the friendly trainer seems to score slightly better. However this is not statistically significant.

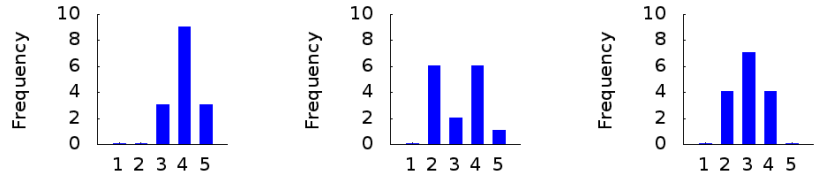


(a) "I felt motivated" (b) "I felt competent" (c) "I felt close to the trainer"



(d) "I felt motivated" (e) "I felt competent" (f) "I felt close to the trainer"

Figure 6.8: Statistics (Average, standard deviation) and ranking (best, worst) of trainer versions for the questions about motivation and contributing factors. Only part of the charts are shown, full results are in appendix H.



(a) Baseline (b) Rude (c) Friendly

Figure 6.9: Distribution of responses to "I felt close to the trainer" (1=Totally agree, 2=Agree, 3=Neutral, 4=Disagree, 5=Totally disagree)

Please see figure 6.9. The distribution for the baseline and friendly trainer look like a normal distribution. The distribution for the rude trainer shows two peaks, indicating a difference in preference under the participants. Indeed the Pearson Chi Squared test indicates it is very unlikely ($P = 0.04$) that this data is normally distributed.

6.2.4 Assistance factors

The trainer versions have a different style of assisting the user while performing exercises. The questions concerning assistance where intended to capture some data about the perceived differences in assistance.

The first question in this section was previously asked, phrased differently. There should be a high correlation, table 6.4 confirms that.

Questions	Correlation
"The training was assisted well" and "The trainer assisted me well"	0.7 ($sig = 0.00$)

Table 6.4: Correlation between synonymously phrased questions. A significance below 0.05 means a valid correlation.

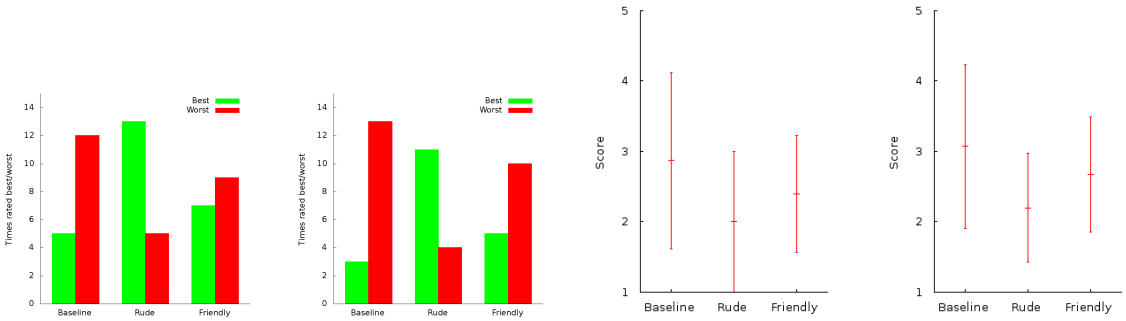
Where the first question in the assistance category is very general, the remaining questions each concern an item that might contribute to assisting a user properly. Table 6.5 shows the correlation of each question with the first question "the trainer assisted me well". This indicates if there is a relation between assistance and the contributing factors we inquired about.

Question	Correlation
"The trainer explains enough"	0.55 ($sig = 0.00$)
"The trainer explains too much"	0.16 ($sig = 0.30$)
"The trainer adjusts to my performance"	0.49 ($sig = 0.00$)
"The trainer acknowledges my performance"	0.34 ($sig = 0.02$)

Table 6.5: Correlation between assistance ("The trainer assisted me well") and contributing factors. A significance below 0.05 means a valid correlation.

For the questions "The trainer explains enough" and "The trainer adjusts to my performance" there is a decent correlation with "The trainer assisted me well". In the case of "The trainer explains enough", both questions are phrased positively. It is widely accepted that people have a natural tendency to agree with positively phrased questions, so in retrospect the choice of words for those questions was probably bad. But for "The trainer adjusts to my performance" the correlation does not look like an artefact of bad wording.

The correlation between "The trainer assisted me well" and "The trainer adjusts to my performance" does not necessarily mean there is a direct connection, other differences between the trainer versions (apart from adjusting to the users performance) might be the reason for the correlation.



(a) "The trainer explains enough" (b) "The trainer adjusts to my performance" (c) "The trainer explains enough" (d) "The trainer adjusts to my performance"

Figure 6.10: Statistics and ranking of trainer versions for some assistance factors.

Please see appendix H for the results of the statistical tests. For the general question "The trainer assisted me well" the difference between the trainer versions is not significant. For the more detailed questions there is a significant difference between the trainer versions, but only a few of the individual tests show a significant difference. Repeating the questionnaire with more exaggerated differences between the

trainers and more respondents could show more significant results.

Figure 6.10 shows statistics ans the ranking of the trainer versions for some assistance factors. The rude trainer score better than the baseline and friendly trainer, but only the difference with the baseline is significant.

It is interesting that only the rude trainer scores high on adjusting to the performance, the friendly trainer also adjusts his feedback to the users performance. Apparently this was only noticeable in the rude trainer. Perhaps the friendly trainer is so polite, people do not notice him adjusting to the performance.

6.2.5 Trainer character

The questions about the trainers character traits don not tell us which trainer is the best or worst. However we designed each trainer to have a specific personality, with these questions we can check if the users perceived the trainer versions the way we intended them.

Question	**Correlation**
"The trainer is happy" and "The trainer is annoyed"	-0.16 ($sig = 0.28$)
"The trainer is supportive" and "The trainer is critical"	0.11 ($sig = 0.47$)
"The trainer is patient" and "The trainer is irritable"	-0.41 ($sig = 0.01$)
"The trainer is rude" and "The trainer is polite"	-0.60 ($sig = 0.00$)

Table 6.6: Correlation between opposite character traits. Significance below 0.05 means a valid correlation

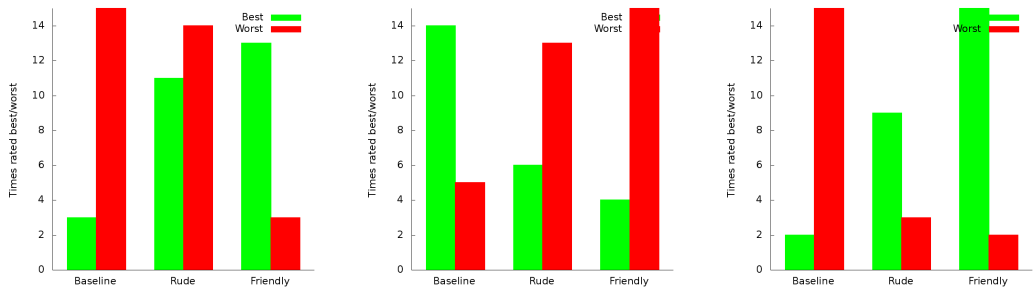
The questions about character traits were intended as four pairs questions about opposite character traits, this can for example identify users that answer questions randomly. Table 6.6 shows the correlations between these pairs. Since they were designed as opposites a large negative correlation is expected. For some pairs of opposites this is correct and significant, other character traits are not as opposed as intended.

For all character traits, except critical and patient, there is a significant difference between the trainer versions. For patience the results are right on the border of significant, a larger group of respondents would probably have shown a significant difference. For all character traits the rude and friendly trainer score similar, better than the baseline. For politeness the friendly trainer even scores significantly better than the rude trainer.

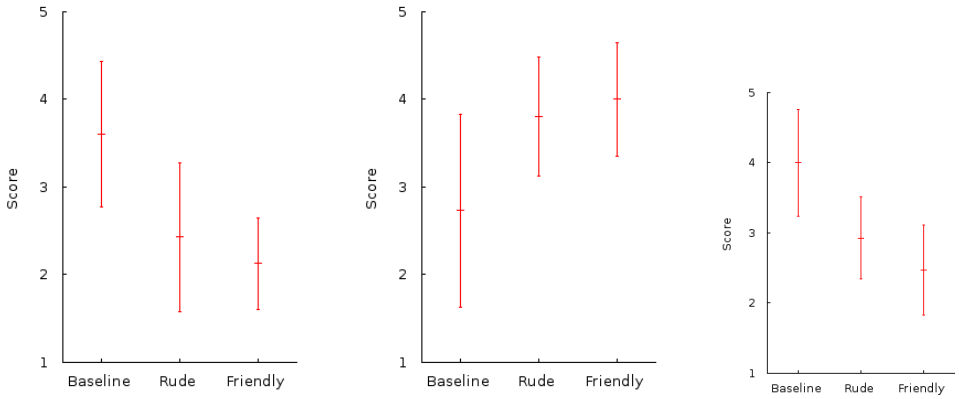
The rude and friendly trainer versions were designed to have different characters: the rude trainer was meant to be rude and impatient, the friendly trainer was meant to be friendly and polite. The baseline trainer was not meant to have a specific character, but it is interesting to see what traits users ascribe to it. Please see appendix H for the full results, below is a listing of the character traits each trainer version scores high on.

- **Baseline** : annoyed, critical, irritable, rude
- **Rude** : happy, critical, polite
- **Friendly** : happy, supportive, patient, polite

The friendly trainer scores as expected. The rude trainer is apparently not perceived as very rude. Participants see mainly bad traits in the baseline trainer.



(a) "The trainer is supportive" (b) "The trainer is rude" (c) "The trainer is polite"



(d) "The trainer is supportive" (e) "The trainer is rude" (f) "The trainer is polite"

Figure 6.11: Statistics (average, standard deviation) and ranking (best, worst) of trainer versions for a few character traits. The rude trainer actually scores quite high on politeness.

6.2.6 Free form comments

Appendix H lists free form comments users wrote down after viewing the videos and completing the questionnaire. We encouraged users to write down comments about things they liked, disliked, general comments or how they would describe the different trainer versions. Here we will briefly discuss the comments that occur most frequently.

Most comments were about the text to speech voice of the prototype, users complained about the quality and/or understandability and suggested to use voice samples. Users also noted the text to speech generator produces an emotionless voice. If the trainer is annoyed about the performance and makes blunt comments, it should sound angry.

Many comments were also on the topic of timing of feedback. In the videos, the trainer gives feedback exactly each 30 seconds. Users disliked that, because the person in the video is frequently making a mistake for quite some time before the trainer gives feedback, until then the trainer does not seem to notice the mistake.

Several users remarked that it was good how the rude trainer interrupted the exercise after a repeated mistake, to give additional instructions and demonstrate the exercise.

Two users said all the trainer versions stayed very polite while facing repeated mistakes: a real human trainer would react much more annoyed.

Two users described the baseline trainer as a robot, the descriptions for the other trainer versions varied, people perceived the rude trainer as somewhat strict or blunt, the friendly trainer was described

soft and the nicest (friendliest).

Chapter 7

Discussion and future work

Here we discuss the results we have presented before. The discussion is split into three parts: influencing the users speed, the motivational mechanisms and the trainer's software architecture.

7.1 Speed influencing

Our algorithm to influence the user's speed has given good results. As long as users have sufficient time (in our case, about 10 seconds) to adapt to the new speed, they pick up speed changes very well. Changing the speed is even intuitive: users did not need instructions to understand the intention of the trainer when he changed the speed at which he performed the exercise.

While developing and evaluating the prototype we did get some ideas for future experiments or improvements:

- The motion analyzer requires some work. It frequently produces measurement errors, if the user moves hectically, the results are practically useless.
- The frequency at which the user performs the exercise tells us quite some interesting things: the user can or can not keep up with the trainer, the user sometimes loses the pace for a short while. We should do something with this information.
- The trainer currently always performs exercise synchronized with the user to indicate the tempo. Perhaps the trainer could use different ways to indicate the tempo when things are going well, for example snap his fingers or indicate the tempo verbally. As soon as the user loses pace, the trainer could start demonstrating the tempo again by performing the exercise along with the user.

To challenge each user optimally requires some more work. The intensity of an exercise varies a lot depending on the user. Users have a different level of fitness and the heartrate associated with a level of fitness changes with age. For some users the current approach worked, but not for all users.

While performing the experiments and from discussion with participants we gathered some ideas to improve upon the current approach, so that we can challenge each user. Some ideas for future work:

- We observed that it takes some time for the heart rate to rise even when the exercise is already challenging. A warming up could get the heart rate to an elevated level before starting the actual workout.
- The trainer needs multiple exercises with a similar goal and different basic intensity. The exercise that best matches the current users fitness level can then be selected. For example, the jumping jacks exercise is similar to sidesteps, but much more intense.

- Add an algorithm that can plan a workout: begin with a warming up, then some cardio exercises and so on. The algorithm should be able to adjust the exercises to the users performance: switch to a more or less intense exercise.

7.2 Motivational mechanisms

The differences respondents perceived between the trainer versions were not statistically significant for the first three broad questions, *"The training was effective"*, *"The training was motivating"* and *"The training was assisted well"*. For the more detailed questions (about the same topics) we did find significant differences. An explanation is that users might tend not to give extreme answers for broadly phrased questions, or only if they perceive a very clear difference.

For the motivational factors, competency *"I felt competent"* and relatedness *"I felt close to the trainer"*, the rude trainer and friendly trainer both scored significantly better than the baseline trainer. For autonomy *"I felt in control"*, there was no difference. There also was a large correlation between the motivational factors and motivation level (*"I felt motivated"*). The differences for the contributing factors, combined with the fact that the psychological model of motivation where we got the factors from is widely accepted, suggest our motivational mechanisms did indeed improve motivation. For autonomy there was no significant difference, this is probably because the trainers did not differ on aspects that would influence autonomy. For example, in none of the trainer versions the user chooses the workout.

Concerning the assistance factors, there were significant differences for the last two questions: *"The trainer adjusts to my performance"* and *"The trainer acknowledges my performance"*. These two questions also had a significant positive correlation with the general impression of assistance, *"The trainer assisted me well"*. While that alone does indicate a causal link, it would be interesting to explore it further: A human sports trainer also adjusts his feedback to the performance of the sporter. In comments, users indicated they liked that the rude trainer interrupted the training for extra explanations and a demonstration of the exercise. This might explain why the rude trainer even scored slightly better than the friendly trainer (but not statistically significant).

For the character traits the rude trainer and the friendly trainer scored much better than the baseline trainer. Actually, *"The trainer is critical"* is the only question without significant differences. Generally speaking, the baseline trainer scored high for the bad character traits and the rude trainer and friendly trainer scored high on the positive character traits. It is interesting that the rude trainer did not score high on being rude, but actually was rated high on politeness. In comments, users called the rude trainer critical and the friendly trainer soft. They also said a real coach would be much more annoyed with repeated mistakes. Apparently the rude trainer was perceived differently from how we designed it.

Overall, the baseline trainer scored worse than the rude trainer and the friendly trainer. For some items there were differences between the rude and friendly trainer, but most were not statistically significant. An experiment with a larger number of respondents might find more statistically significant differences between the rude and friendly trainer.

There were several comments about the specific moments at which the trainer gave feedback: the trainer responded to late, the timing of feedback was wrong and similar comments. Our current approach was to determine the most important error once every 30 seconds. This approach inherently means there will be some time between a mistake and the feedback. Human trainers react to mistakes as they occur, but choose more intelligently on which mistakes they give feedback. Since there were multiple comments about this, future work should look into this.

For some of the questions there appeared to be two groups in the answers. This is interesting, because it suggests people have a differing personal preference in that matter. This was also evident from comments users made, some preferred the rude trainer while others disliked the more blunt feedback. Perhaps this is related to the users' personality.

Where we calculated correlations between the results, they mostly matched our expectations. The applied model of motivation was confirmed by the questionnaire and the design of the questions (synonyms, opposites) was reflected in the answers. This probably indicates reliable and valid answers.

7.3 Trainer architecture

Overall, the trainer architecture performed very well. We were able to implement all the desired functionalities without major problems. There were only minor changes from the original architecture. In the implementation notes section we already mentioned some issues we encountered during the implementation, we briefly discuss them here.

We had some issues related to multithreading, having multiple threads in one program. Especially the input modules each have a separate thread. A next version of the trainer should consider this more during the design.

In a few cases we could have kept the responsibilities of a module clearer and more compact. For example, the output module has a lot of responsibilities, this makes the implementation complex and opaque.

Users commented that the trainer did not respond to mistakes immediately, but only after some time. This was a deliberate choice, but if we come up with an algorithm that determines in real-time for which errors and at what moment to give feedback, the trainer architecture can handle the change. Modules can respond to events (messages) and to timers, a real-time algorithm for giving feedback on errors could react to an event rather than a timer.

Chapter 8

Conclusions

We set out to improve interaction of the virtual trainer with users. The goal was to develop a trainer that supports a user with his workout goals like a real sports trainer: assist technically with the execution of an exercise and motivate the user to keep working hard. From this broad goal we have picked two main topics: we tried to make users perform exercises at a specified intensity and we have worked on motivating users by adjusting the feedback the trainer gives. We have evaluated these two aspects of the trainer with users, to determine if and how our ideas improved the trainers performance. Finally we have developed a software architecture for implementing the virtual trainer application.

8.1 Speed influencing

We worked on influencing the speed at which users perform a workout, to be able to better challenge users during a workout. At the start of this document we broke this problem down to the following questions:

1. How do we measure the exercise's intensity?
2. Can we influence the exercise's speed?
3. Is influencing the speed intuitive to the user?
4. Does the exercise speed influence the intensity?
5. Can we make the user do the exercise at a specific intensity?

The ideas behind our approach are, that performing an exercise at a higher speed is more intense and an exercise is most challenging when done at the optimal intensity. The trainer prototype uses a heartrate sensor to measure the intensity and a wii mote (accelerometer) to measure the speed of an exercise. To influence the users speed, the trainer moves faster or slower, with the intention that the user picks up the change in speed. In our first and second test the trainer did an exercise with the user, increasing the speed. The first test had instructions, the second was for intuitivity and avoided any instructions about was would happen. The third test used the heart-rate sensor to make a user work out at a fixed heart rate.

The evaluations showed that our prototype can influence the users speed in an intuitive way. We have learned that increasing the speed does influence the intensity, however the intensity of a specific exercise differs per user. To make a user work out at a specified intensity requires more than just changing the speed, for example a system that can choose a more or less intense exercise.

Our work has resulted in a system that can influence the user's speed in an intuitive way. We have suggested concrete additions with which the system can make a user work out at a specific intensity. Working out at different intensities gives different training results: For example increased stamina or increased strength. Usually a human sports trainer looks at the users training goals and level of fittness to compile a workout that suits his goals. Using our system, a virtual trainer could be developed that takes over some tasks from a human colleague.

8.2 Motivational mechanisms

Regarding motivating users during a workout, we have applied aspects of several psychological models of motivation, politeness and mood to the virtual trainer, to enhance the training experience. We started with the following research questions:

1. How do users perceive a trainer that simply states mistakes versus a trainer that uses aspects of motivational theory to give feedback?
2. Do users prefer a polite trainer or a more rude trainer?
3. Does a trainer who uses aspects of motivational theory enhance motivation compared to a trainer that simply states mistakes?

Research by Ryan and Deci [2000] states that motivation is influenced by three factors: autonomy, competency and relatedness. His work shows, that enhancing these factors increases motivation. Brown and Levinson [1987] proposes a model of politeness, where polite versions of feedback are used to avoid threatening a persons feeling of competency and autonomy. Paleari et al. [2005] added a mood to a tutoring agent. Their agent responds positive if the user performs well, supportive if the user performs bad but tries hard and blunt if the users does not do his best.

We combined these models in our prototype. The trainer has a mood that is influenced by the users performance and the trainer chooses feedback of varying politeness, depending on that mood. To evaluate the effects of these motivational mechanisms, we made a video of a user performing a workout with the trainer. In the video the user makes several mistakes. We made three versions of the video, where the trainer responds differently to those mistakes: a baseline without the motivational mechanisms, a friendly and a more rude trainer. We showed these videos to 16 users and gave them a questionnaire to rate the trainer on various aspects.

Participants found the rude trainer and the friendly trainer more motivating than the baseline trainer, they were rated higher for the factors competency and relatedness. For autonomy, the differences between the trainer versions were not statistically significant. Participant found the rude trainer and the friendly trainer assisted the training better. Furthermore, there was a significant correlation between the rating of assistance, the rating of adjusting to the users performance and the rating of acknowledging the users performance. Concerning character traits of the trainer versions, the rude trainer and the friendly trainer scored higher for positive attributes (happy, polite, supportive, patient) and the baseline trainer scored highest on negative attributes (annoyed, irritable, rude). The differences in rating between the rude trainer and the friendly trainer were not statistically significant for all but one question (politeness).

Motivating sporters is one of the main tasks of a trainer, a virtual trainer that can better motivate users is a step forward. Especially for people that sport not for fun, but for example to lose weight or look fitter, a boost in motivation might be a help to achieve their goals. Our work has explored possible advantages of implementing motivational mechanisms (mood, politeness) in a virtual trainer. The results show users perceived improved performance in various areas. Free-form comments from users during the evaluation showed areas where the trainer is still lacking.

8.3 Trainer architecture

The virtual trainer needed a software architecture that not only suits the requirements of this project, but which can be used for future work as well. We have looked at previous work in this area of software design by Wooldridge [2002], Sloman [2003], Thórisson et al. [2008] and designed a software architecture for a virtual trainer. We have used a multi agent approach, where multiple agents generate behavior in parallel. The design has performed well, we did not encounter major issues, any changes needed during implementation fit in with the original design.

8.4 Final words

We have worked on being able to optimally challenge a user with an exercise and we have worked on on motivating users during the workout. Those two topics are perhaps the most important aspects of a sports trainer. We have found concrete areas where future work can build on current results and apply the insights from this research. While the virtual trainer is still far from a real sports coach, we did make a step forward and opened the door for a new generation of virtual trainers. Isn't a sport trainer that motivates and challenges you a big improvement over a workout video or sitting on a home trainer?

Appendices

Appendix A

An aerobics class

To gain insight in the chosen sport for the trainer, i have attended an aerobics class several times. The aerobics teacher has provided to me instructional videos for an aerobics trainer, containing exercises, goals of the exercise and common mistakes.

The aerobics class was held in a large gymnastics hall, there was one instructor and about 50 participants. The instructor played a music tape in the background as a tempo indicator. She explained and demonstrated exercises and performed the exercises with the participants.

Most exercises were combinations of multiple motions which were gradually extended. For example, an exercise could start as a sidestep, after 4 repetitions the instructor would add a lunge at both sides of the sidestep: sidestep left, lunge, sidestep right, lunge.

As the group was 50 people, the instructor could/did not give hints to individual participants, but rather gave general instructions or demonstrated the exercise again if many students made mistakes.

The workout was divided into several parts. First there was a warming up, to increase heartrate and prepare the muscles for real exercise. Then there were cardio exercises, greatly increasing the heartrate and stressing the cardiovascular system. After this were strength exercises, to train the muscles. Particularly the muscles of the legs, bottom and belly were worked out. Finally there was a cooling down and some stretching exercises.

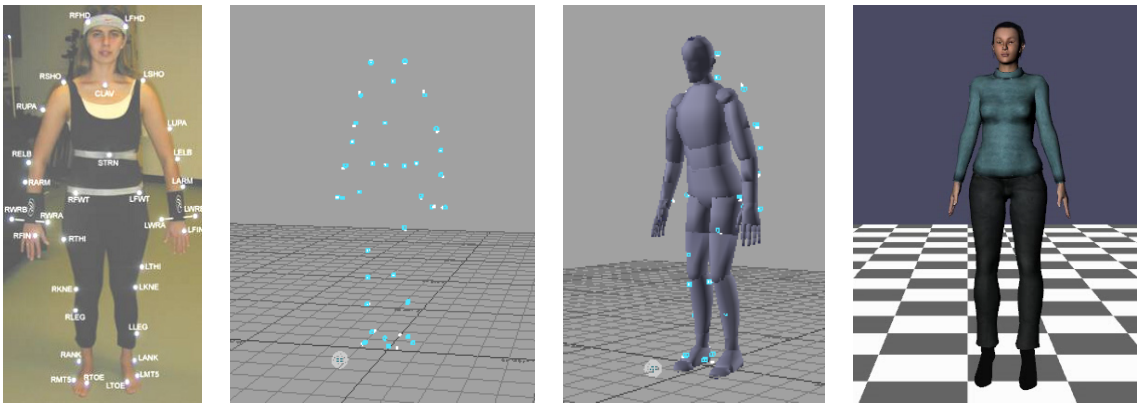
Appendix B

Motion capture: Making the trainer move

For the trainer to perform realistic motions, we have recorded motions of a real human and processed the data so the trainer avatar can perform those motions. This process is called motion capture. There were three major parts to this process:

1. Recording ("capturing") motions.
2. Processing the recorded motions onto an animated character.
3. Import the motions in the trainer application.

Recording the exercise motions was done at the motion lab of the university. A vicon motion capture system was used. The vicon system uses infrared reflecting markers that are placed on the body, the system determines the position using multiple infrared cameras. During recording, the room is darkened and illuminated with infrared lights. The markers are placed at fixed positions on the body, see figure B.1. For each body part or limb, there is at least one marker. The person to be recorded wears skin-tight clothes and the markers are secured using double-sided tape.



(a) Marker placement (b) Recorded marker positions (c) Animating a character (d) Trainer performing the motions.

Figure B.1: Steps in animating the virtual trainer.

After the motions are recorded, they are processed to let an animated character perform the recorded motions. We have used the MotionBuilder suite for this. First the markers are labeled. This is done

through the whole recording: markers are sometimes briefly obscured by body parts and then reappear later in the recording. Those are all assigned the same label, the position is interpolated for the frames where the markers are missing. To aid the interpolation, constraints like a fixed distance between adjacent markers can be defined.

Once all markers are labelled and the parts in between are interpolated, a virtual character is added. The virtual character is resized to the proportions match the proportions of the person that was recorded and the markers are assigned to the corresponding body parts of a character. Then we have a character that performs the same motions as the person we recorded.

The last step done in MotionBuilder, is to import the skeleton of the virtual trainer. The motions of the character we animated before are mapped onto that skeleton.

The final part of the process is getting the virtual trainer to perform the processed motions. For this we convert the motions into a format the virtual trainer software can handle. The motions are then cut into suitable pieces: a starting part, a main part that will be repeated and an end part.

The new motions are then added to the trainers repertoire.

Appendix C

Evaluation workout

Below is the workout performed in the videos used for evaluating the trainers motivational mechanisms. For each trainer setup the responses to the users mistakes are shown.

	Baseline, neutral trainer	Impolite, rude trainer	Polite, friendly trainer
Warming up: sidesteps (30s) - Forget the arms	"Move your arms!"	"Don't forget your arms!"	"Let's move our arms more!"
Exercise: jumping jacks (1m30) - Forget the arms	"Move your arms"	"Your arms!"	"Don't forget your arms"
- Keep forgetting the arms	"Move your arms"	"You keep forgetting your arm!" + interrupt exercise for explanations "You have done better."	"Move your arms!"
Exercise: knee bends (1m)	"You move too slow!"	"Watch your pace!"	"Good!" "Keep pushing yourself!"
Cooling down: sidesteps (30s)			"Well done, remember your arms next time!"

Appendix D

Motivation questionnaire

Participant data

Age :	...
Gender :	...
Group sport experience :	.....
	.....
	.....
	.....

Trainer evaluation form

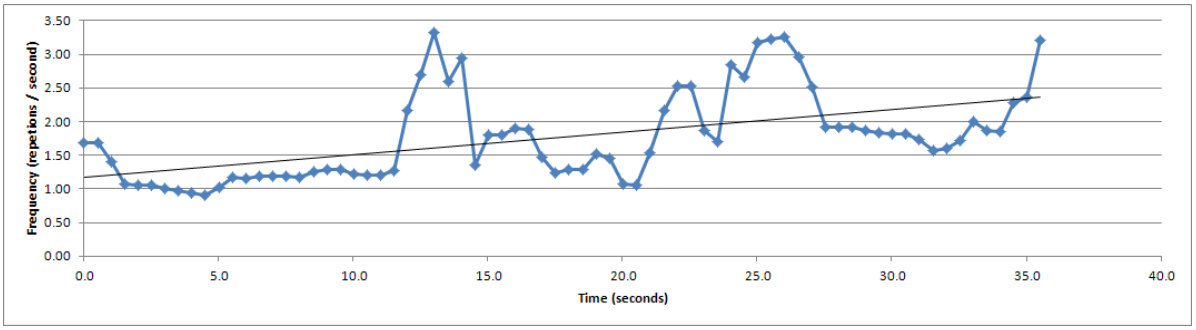
While viewing the video, please imagine yourself as the student. Answer the questions below as if you were the one doing a workout with the trainer.

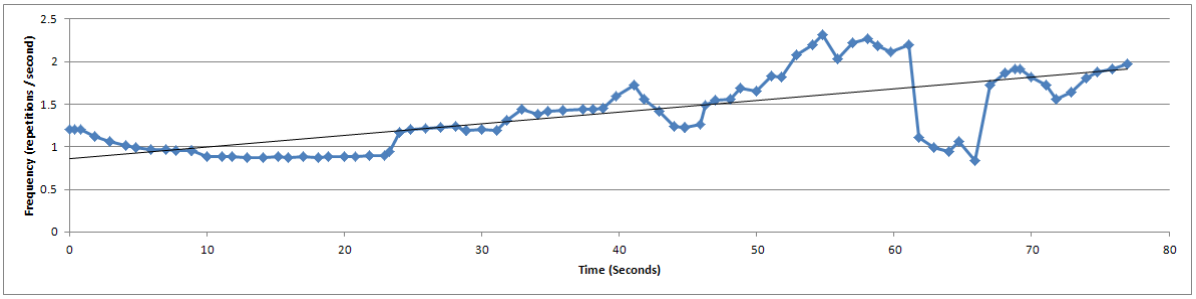
	Totally agree	Agree	Neutral	Disagree	Totally disagree
Training experience					
The training was effective	.	.	.	.	.
The training was motivating	.	.	.	.	.
The training was assisted well	.	.	.	.	.
Motivational factors					
I felt motivated	.	.	.	.	.
I felt in control	.	.	.	.	.
I felt competent	.	.	.	.	.
I felt close to the trainer	.	.	.	.	.
I was in charge	.	.	.	.	.
I performed well	.	.	.	.	.
I got along with the trainer	.	.	.	.	.
Assistance factors					
The trainer assisted me well	.	.	.	.	.
The trainer explains enough	.	.	.	.	.
The trainer explains too much	.	.	.	.	.
The trainer adjusts to my performance	.	.	.	.	.
The trainer acknowledges my performance	.	.	.	.	.
Trainer character					
The trainer is happy	.	.	.	.	.
The trainer is annoyed	.	.	.	.	.
The trainer is supportive	.	.	.	.	.
The trainer is critical	.	.	.	.	.
The trainer is patient	.	.	.	.	.
The trainer is irritable	.	.	.	.	.
The trainer is rude	.	.	.	.	.
The trainer is polite	.	.	.	.	.

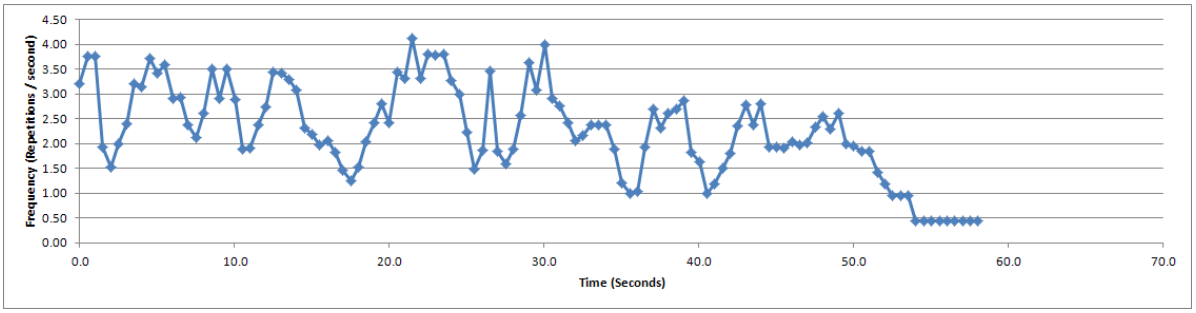
Appendix E

Results: Speed adjustment feasibility

These first three charts are of workouts where the trainers started with 5 repetitions at a constant speed and then the trainers speed increased 5 percent per 5 seconds. This was difficult for users to keep up with the speed, and gave chaotic measurements.

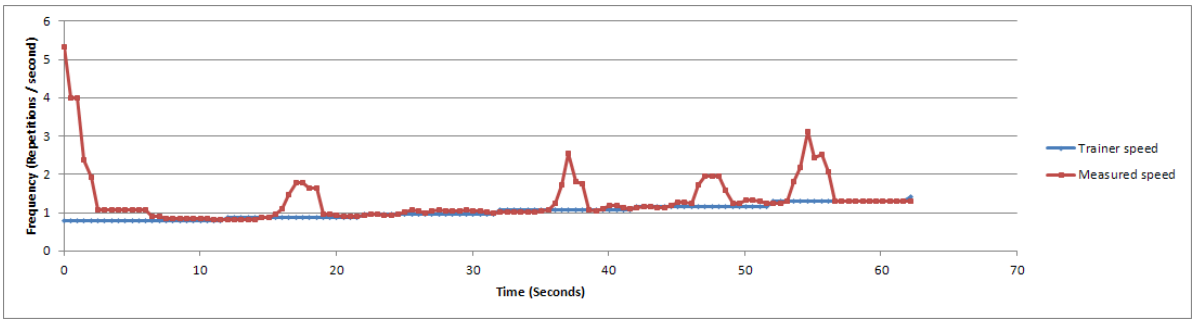


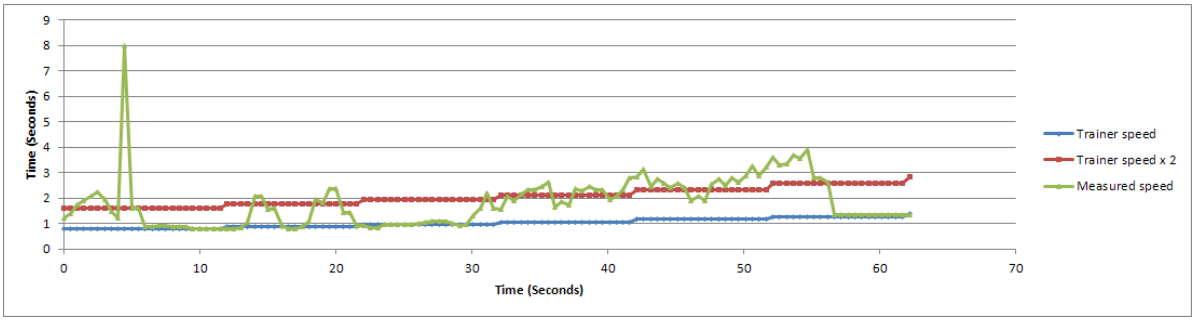




These charts are of workouts where the trainers started with 10 repetitions at a constant speed and then the trainers speed increased 10 percent per 10 seconds. This made it much easier to keep the speed

and good results.





Appendix F

Results: Keeping the heartrate in a range

For these workouts we used the sidesteps exercise, this is a less intense exercise than jumping jacks. First tests showed that jumping jacks produced quite high heart rates, but the heart take some time to rise high. To properly test influencing of the heartrate, a warming up would be necessary.

The goal heartrate was 100 and 120 for a few other subjects. This is marked with the charts. The application was altered to allow heartrate changes from the control interface, as the heart rates of users varied substantially.

The first charts are of test subjects where the heartrate did not rise to the intended goal. The exercise was not intense enough for these users: speeding up infinitely is not possible. After this we have one chart of a user where the heart rate was above the goal to begin with. It did fall in the end however. For this user, the exercise was too intense. Finally we have several sets of charts where the users had a high heartrate, but the trainer reached his lower speed limit and would not slow down more. At last there are charts of an experiment which worked out as intended: the users heartrate rises and falls out of range and speed changes bring the heartrate back in range.

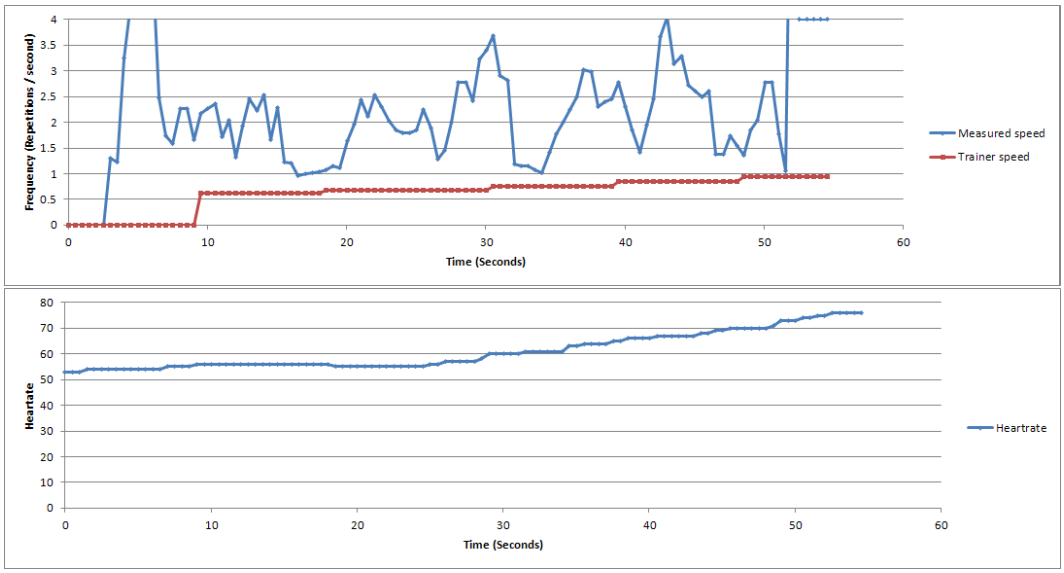


Figure F.1: Participant 1: Trainer and measured speed, heartrate. Goal heartrate was 120, but was not reached.

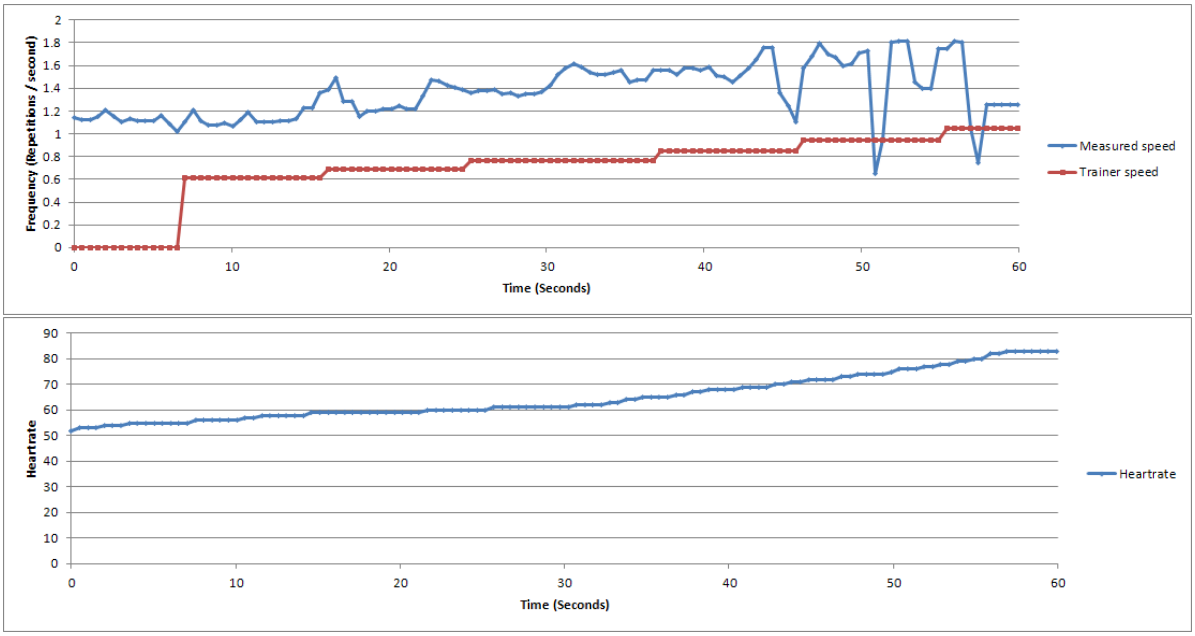


Figure F.2: Participant 2: Trainer and measured speed, heartrate. Goal heartrate was 120, but was not reached.

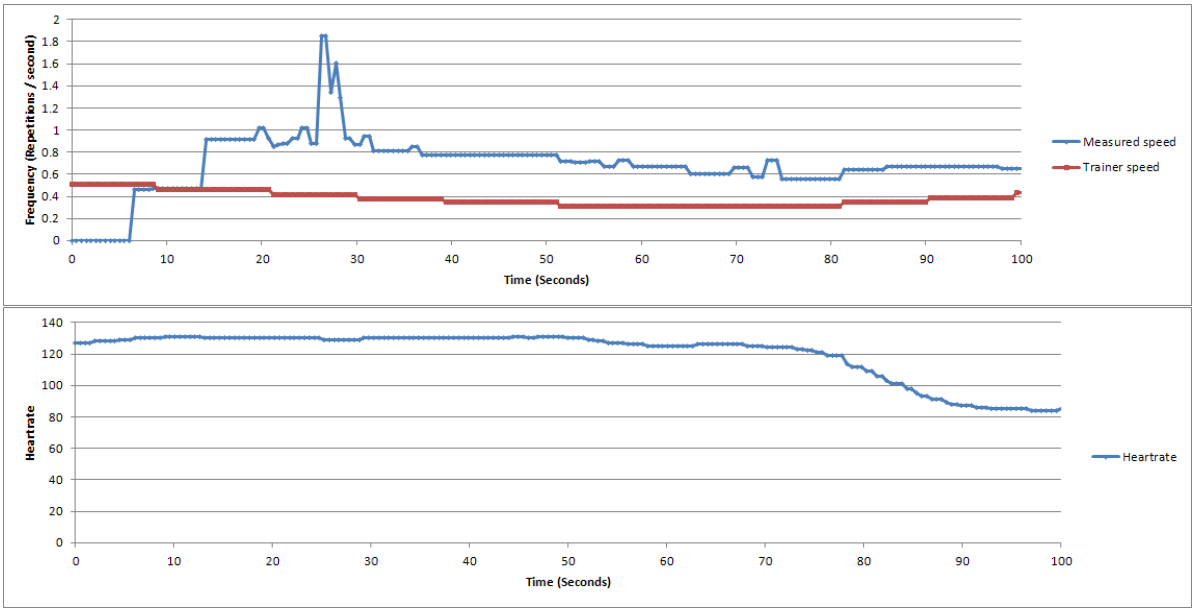


Figure F.3: Participant 3: Trainer and measured speed, heartrate. Goal heartrate was 100, but user was way above that.

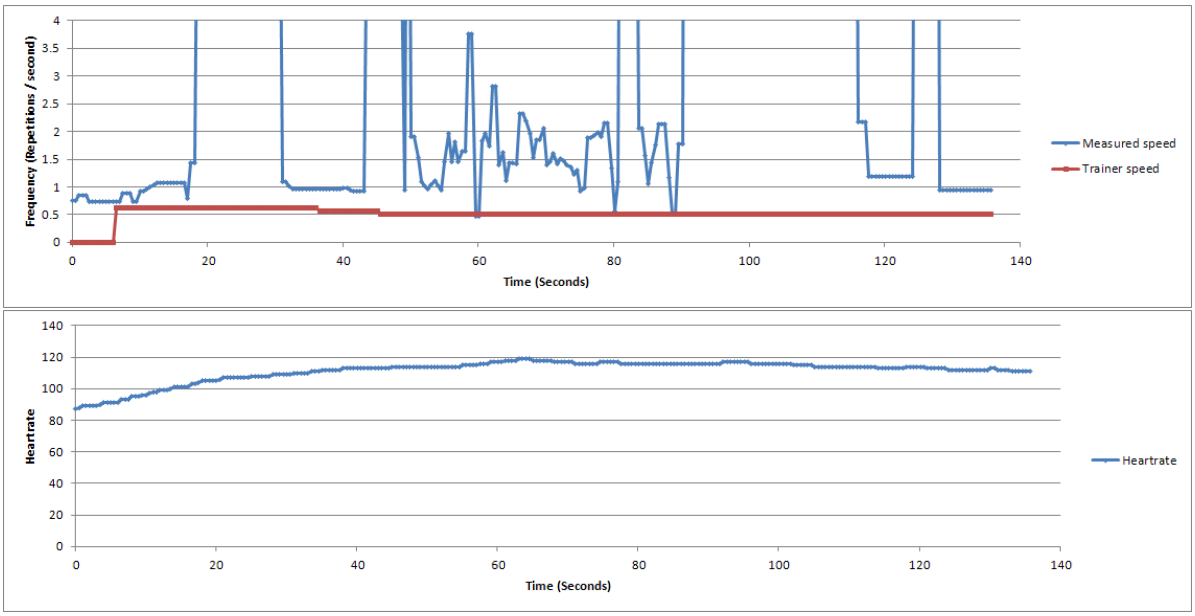


Figure F.4: Participant 4: Trainer and measured speed, heartrate. Goal heartrate was 100, but user was way above that.

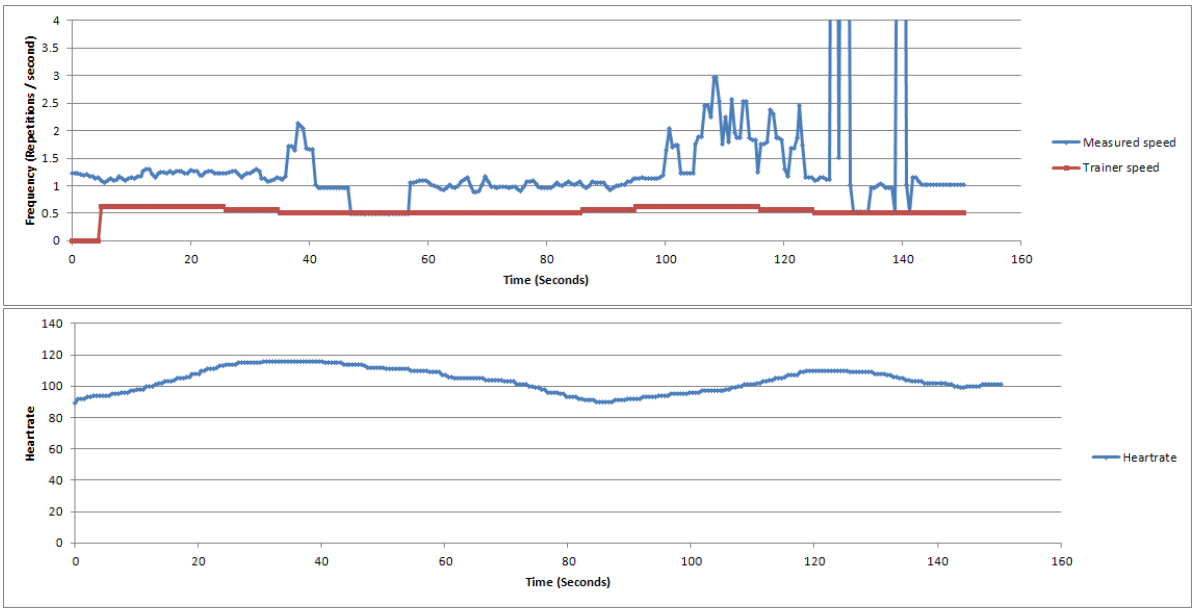


Figure F.5: Participant 5: Trainer and measured speed, heartrate. Goal heartrate was 100. Heartrate rise and fell out of range, but this was corrected by speed adjustments.

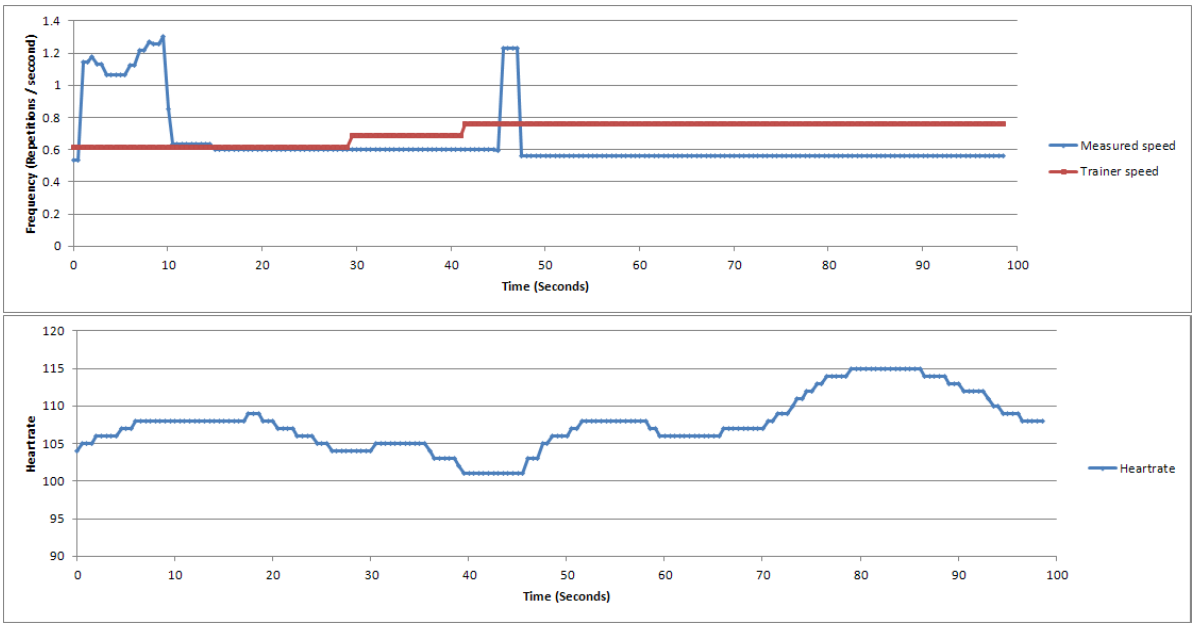
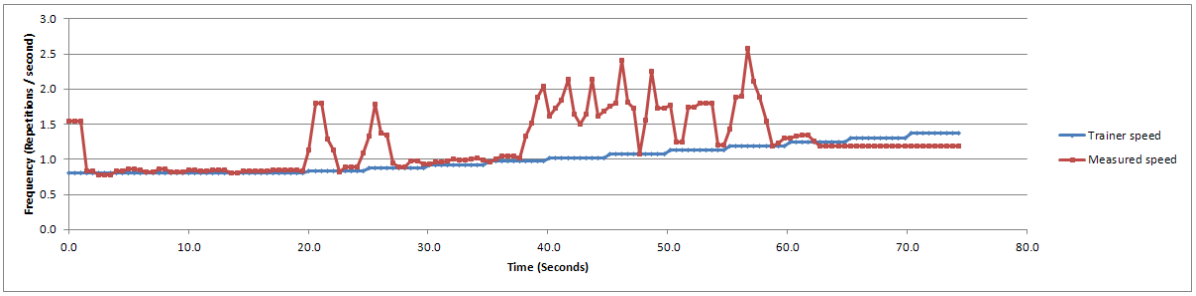


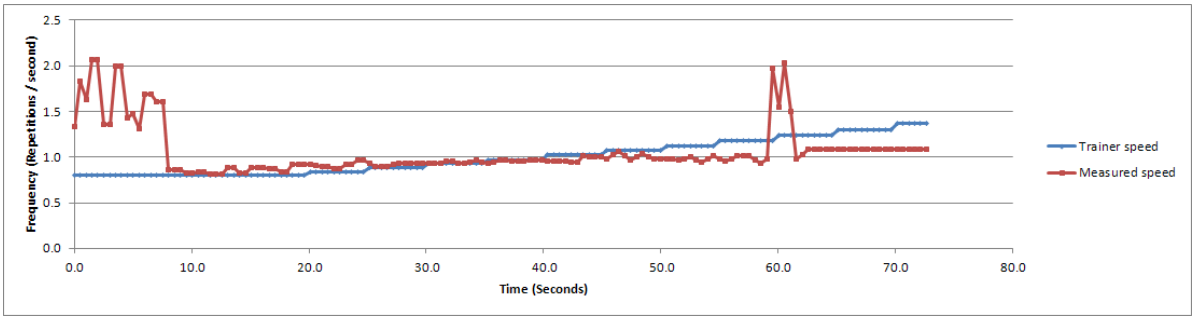
Figure F.6: Participant 6: Trainer and measured speed, heartrate. Goal heartrate was 110. Heartrate got into the range and stayed within range.

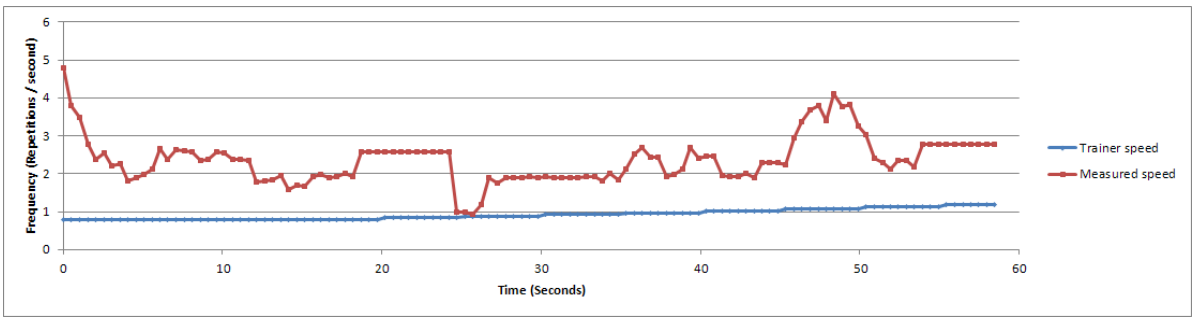
Appendix G

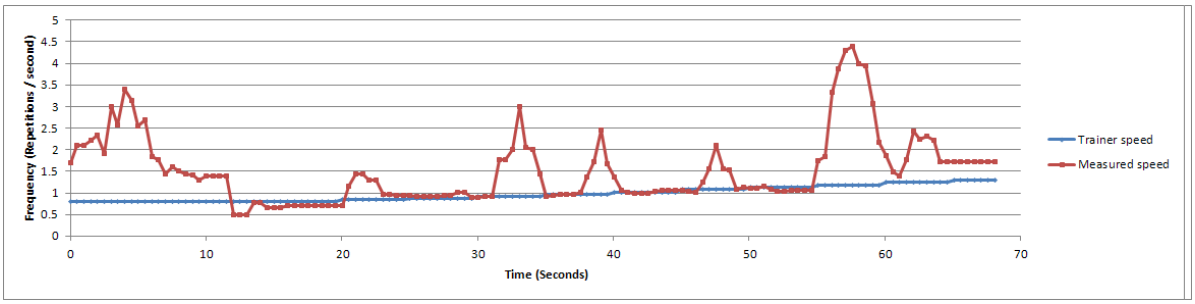
Results: Speed adjustment intuitiveness

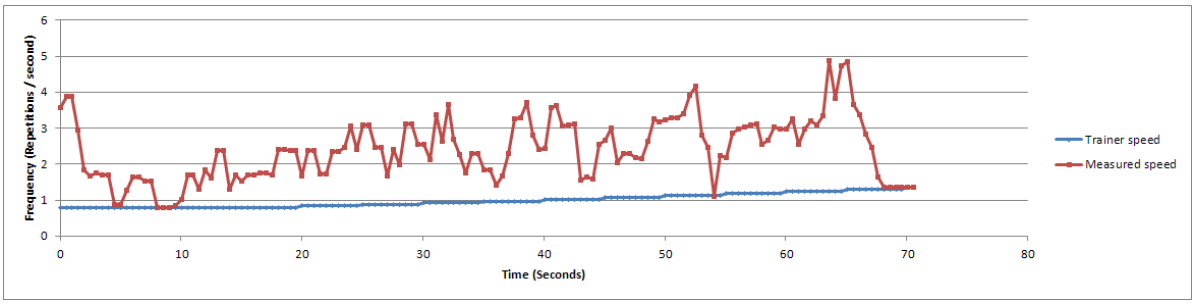
For theses series of tests, the trainer started with 16 repetitions at a fixed frequency of 0.8 repetitions per second. The first 8 repetitions the trainer said "Jump" synchronous with the motions. The second 8 where without comment. After this there where 50 repetitions where the trainers speed increased 5 percent per 5 seconds.

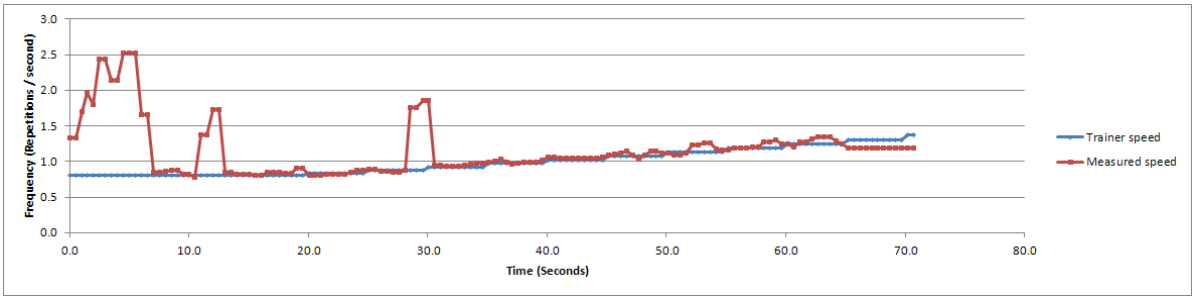







Frequency (Repetitions / second)
Time (Seconds)
Trainer speed
Measured speed


Frequency (Repetitions / second)
Time (Seconds)
Trainer speed
Measured speed


Frequency (Repetitions / second)
Time (Seconds)
Trainer speed
Measured speed

Appendix H

Results: motivation questionnaires

The questionnaire was filled in by 16 users, all male, with ages from 20 to 35. Of the 16 participants, 2 had no experience with sports training supervised by a coach, 8 had little experience (1 or 2 sports, less than 3 years) and 5 had a lot of experience (multiple sports, more than 3 years).

Below is a list of verbal or written comments by participants, grouped together (and with a count) where applicable.

- Text-to-speech is of bad quality/hard to understand. (6)
- Text-to-speech shows no emotion. (4)
- Replace text-to-speech with voice samples? (3)

- Feedback is too late. (4)

- Interruption with extra explanations and demonstration is a good thing. (3)
- Conclusive remarks after workout are a good thing.
- What the trainer actually says is good.

- A human trainer would be much more annoyed about repeated mistakes. (2)
- A human trainer would give positive feedback after improvements.

- Baseline trainer was annoying, naggy.
- Rude trainer was blunt, but motivating.
- Rude trainer is most human-like.
- Baseline is like a robot. (2)
- Friendly trainer is very soft, rude trainer is more strict,
- Friendly trainer is the nicest.
- Rude trainer is the best.
- Video quality is not optimal.
- Differences between trainers are hard to notice.

- Only show face during exercise explanations?

- Being friendly is useful at school, a trainer should be quite explicit.
- Criticism on abilities should be polite, criticism on willingness to work hard should be explicit.
- The physical component of the exercise is missing, that changes the experience of getting feedback.

	Best			Worst		
	Baseline	**Rude**	**Friendly**	**Baseline**	**Rude**	**Friendly**
The training was effective	9	13	13	14	8	10
The training was motivating	5	8	7	12	6	6
The training was assisted well	5	13	10	15	6	8
I felt motivated	4	11	13	13	6	5
I felt in control	7	10	13	12	9	6
I felt competent	5	8	11	13	7	5
I felt close to the trainer	3	10	12	12	6	3
I was in charge	6	8	11	12	10	7
I performed well	5	7	13	11	8	3
I got along with the trainer	4	8	10	13	7	5
The trainer assisted me well	5	10	10	11	5	7
The trainer explains enough	5	13	7	12	5	9
The trainer explains too much	8	14	13	15	9	10
The trainer adjusts to my performance	3	11	5	13	4	10
The trainer acknowledges my performance	2	9	11	13	4	3
The trainer is happy	4	11	12	15	5	5
The trainer is annoyed	13	5	4	4	9	13
The trainer is supportive	3	11	13	15	4	3
The trainer is critical	12	11	6	8	8	13
The trainer is patient	9	8	14	12	12	7
The trainer is irritable	13	6	5	6	11	13
The trainer is rude	14	6	4	5	13	15
The trainer is polite	2	9	15	15	3	2

Table H.1: Ranking of trainer versions, the table shows how often each trainer version is ranked best or worst.

	Baseline					Rude					Friendly				
Distribution	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
The training was effective	1	6	7	1	0	1	10	4	0	0	1	10	3	1	0
The training was motivating	0	5	2	6	2	0	8	5	2	0	1	6	6	2	0
The training was assisted well	0	6	4	4	1	1	12	1	1	0	0	9	4	1	0
I felt motivated	0	1	9	4	1	0	9	3	3	0	0	10	3	2	0
I felt in control	0	5	6	3	1	0	6	7	1	1	0	8	5	2	0
I felt competent	0	4	5	5	1	0	5	8	1	0	1	8	5	1	0
I felt close to the trainer	0	0	3	9	3	0	6	2	6	1	0	4	7	4	0
I was in charge	0	3	8	3	1	0	5	6	3	1	0	8	3	4	0
I performed well	0	3	5	6	1	0	4	7	4	0	0	7	8	0	0
I got along with the trainer	0	1	5	5	4	1	3	7	3	1	1	3	7	4	0
The trainer assisted me well	1	5	6	1	2	2	8	2	2	0	2	8	4	1	0
The trainer explains enough	1	7	2	3	2	4	9	1	0	1	1	9	3	2	0
The trainer explains too much	0	0	1	7	7	0	1	0	10	3	0	0	1	12	2
The trainer adjusts to my performance	1	4	5	3	2	2	9	3	1	0	1	5	7	2	0
The trainer acknowledges my performance	1	2	2	9	1	2	7	5	1	0	1	11	1	1	0
The trainer is happy	0	0	4	6	5	0	1	13	0	1	0	4	8	2	1
The trainer is annoyed	0	8	4	2	1	0	1	6	6	2	0	0	4	8	3
The trainer is supportive	0	2	3	9	1	0	10	3	0	1	1	11	3	0	0
The trainer is critical	2	10	2	1	0	1	9	2	1	1	0	8	3	2	2
The trainer is patient	0	7	5	2	1	0	6	8	0	1	0	12	3	0	0
The trainer is irritable	0	6	6	3	0	0	1	6	6	2	0	0	6	5	4
The trainer is rude	2	4	6	2	1	0	0	5	8	2	0	0	3	9	3
The trainer is polite	0	0	4	7	4	0	3	10	2	0	1	6	8	0	0

Table H.2: Distribution of answers to the questionnaire. (1=Totally agree, 2=Agree, 3=Neutral, 4=Disagree, 5=Totally disagree)

	Baseline	Rude	Friendly
The training was effective	$\mu = 2.53$, $\sigma = 0.74$	$\mu = 2.20$, $\sigma = 0.56$	$\mu = 2.27$, $\sigma = 0.70$
The training was motivating	$\mu = 3.33$, $\sigma = 1.11$	$\mu = 2.60$, $\sigma = 0.74$	$\mu = 2.60$, $\sigma = 0.83$
The training was assisted well	$\mu = 3.00$, $\sigma = 1.00$	$\mu = 2.13$, $\sigma = 0.64$	$\mu = 2.43$, $\sigma = 0.65$
I felt motivated	$\mu = 3.33$, $\sigma = 0.72$	$\mu = 2.60$, $\sigma = 0.83$	$\mu = 2.47$, $\sigma = 0.74$
I felt in control	$\mu = 3.00$, $\sigma = 0.93$	$\mu = 2.80$, $\sigma = 0.86$	$\mu = 2.60$, $\sigma = 0.74$
I felt competent	$\mu = 3.20$, $\sigma = 0.94$	$\mu = 2.71$, $\sigma = 0.61$	$\mu = 2.40$, $\sigma = 0.74$
I felt close to the trainer	$\mu = 4.00$, $\sigma = 0.65$	$\mu = 3.13$, $\sigma = 1.06$	$\mu = 3.00$, $\sigma = 0.76$
I was in charge	$\mu = 3.13$, $\sigma = 0.83$	$\mu = 3.00$, $\sigma = 0.93$	$\mu = 2.73$, $\sigma = 0.88$
I performed well	$\mu = 3.33$, $\sigma = 0.90$	$\mu = 3.00$, $\sigma = 0.76$	$\mu = 2.53$, $\sigma = 0.52$
I got along with the trainer	$\mu = 3.80$, $\sigma = 0.94$	$\mu = 3.00$, $\sigma = 1.00$	$\mu = 2.93$, $\sigma = 0.88$
The trainer assisted me well	$\mu = 2.87$, $\sigma = 1.13$	$\mu = 2.29$, $\sigma = 0.91$	$\mu = 2.27$, $\sigma = 0.80$
The trainer explains enough	$\mu = 2.87$, $\sigma = 1.25$	$\mu = 2.00$, $\sigma = 1.00$	$\mu = 2.40$, $\sigma = 0.83$
The trainer explains too much	$\mu = 4.40$, $\sigma = 0.63$	$\mu = 4.07$, $\sigma = 0.73$	$\mu = 4.07$, $\sigma = 0.46$
The trainer adjusts to my performance	$\mu = 3.07$, $\sigma = 1.16$	$\mu = 2.20$, $\sigma = 0.77$	$\mu = 2.67$, $\sigma = 0.82$
The trainer acknowledges my performance	$\mu = 3.47$, $\sigma = 1.06$	$\mu = 2.33$, $\sigma = 0.82$	$\mu = 2.14$, $\sigma = 0.66$
The trainer is happy	$\mu = 4.07$, $\sigma = 0.80$	$\mu = 3.07$, $\sigma = 0.59$	$\mu = 3.00$, $\sigma = 0.85$
The trainer is annoyed	$\mu = 2.73$, $\sigma = 0.96$	$\mu = 3.60$, $\sigma = 0.83$	$\mu = 3.93$, $\sigma = 0.70$
The trainer is supportive	$\mu = 3.60$, $\sigma = 0.83$	$\mu = 2.43$, $\sigma = 0.85$	$\mu = 2.13$, $\sigma = 0.52$
The trainer is critical	$\mu = 2.13$, $\sigma = 0.74$	$\mu = 2.43$, $\sigma = 1.02$	$\mu = 2.87$, $\sigma = 1.13$
The trainer is patient	$\mu = 2.80$, $\sigma = 0.94$	$\mu = 2.73$, $\sigma = 0.80$	$\mu = 2.20$, $\sigma = 0.41$
The trainer is irritable	$\mu = 2.80$, $\sigma = 0.77$	$\mu = 3.60$, $\sigma = 0.83$	$\mu = 3.87$, $\sigma = 0.83$
The trainer is rude	$\mu = 2.73$, $\sigma = 1.10$	$\mu = 3.80$, $\sigma = 0.68$	$\mu = 4.00$, $\sigma = 0.65$
The trainer is polite	$\mu = 4.00$, $\sigma = 0.76$	$\mu = 2.93$, $\sigma = 0.59$	$\mu = 2.47$, $\sigma = 0.64$

Table H.3: Average and standard deviation of the answers to the questionnaire. The answers were coded as numbers: 1=Totally agree, 2=Agree, 3=Neutral, 4=Disagree, 5=Totally disagree.

Questions	F Test	T tests		
		Baseline vs Rude	Baseline vs Friendly	Rude vs Friendly
The training was effective	0.15	-	-	-
The training was motivating	0.06	-	-	-
The training was assisted well	0.00	0.00 *	0.02 *	0.26
I felt motivated	0.01	0.01 *	0.01 *	0.65
I felt in control	0.08	-	-	-
I felt competent	0.01	0.08	0.01 *	0.14
I felt close to the trainer	0.00	0.01 *	0.00 *	0.67
I was in charge	0.16	-	-	-
I performed well	0.01	0.21	0.01 *	0.03
I got along with the trainer	0.02	0.03	0.01 *	0.84
The trainer assisted me well	0.12	-	-	-
The trainer explains enough	0.06	0.06	0.20	0.14
The trainer explains too much	0.03	0.03	0.02 *	0.67
The trainer adjusts to my performance	0.01	0.00 *	0.16	0.07
The trainer acknowledges my performance	0.00	0.00 *	0.00 *	0.49
The trainer is happy	0.00	0.00 *	0.00 *	0.72
The trainer is annoyed	0.00	0.02 *	0.00 *	0.10
The trainer is supportive	0.00	0.00 *	0.00 *	0.16
The trainer is critical	0.06	-	-	-
The trainer is patient	0.03	0.75	0.04	0.03
The trainer is irritable	0.00	0.01 *	0.00 *	0.30
The trainer is rude	0.00	0.01 *	0.00 *	0.19
The trainer is polite	0.00	0.00 *	0.00 *	0.01 *

Table H.4: Results of statistical tests. T tests are blank where the F test is not significant. For the F test a score below 0.05 means a significant difference, for the t tests we consider the difference significant if the score is below 0.02. The significant differences are marked with an asterisk.

Bibliography

Sabarish Babu, C. Zanbaka, J. Jackson, T-O. Chung, B. Lok, M. C. Shin, and L. F. Hodges. Virtual human physiotherapist framework for personalized training and rehabilitation. Short paper, Graphics Interface 2005, 2005.

B. Baptiste. *Poweryoga: Meditatie door beweging.* Karakter Uitgevers BV, 2002. ISBN 90-6112-901-X.

C. Alan Boneau. The effects of violations of assumptions underlying the t-test. *Psychological bulletin*, 37(1):49–64, 1960.

Penelope Brown and Stephen C. Levinson. *Politeness : Some Universals in Language Usage (Studies in Interactional Sociolinguistics).* Cambridge University Press, February 1987. ISBN 0521313554. URL `http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0521313554`.

Justine Cassell and Timothy Bickmore. Negotiated collusion: Modeling social language and its relationship effects in intelligent agents. *User Modeling and User-Adapted Interaction*, 13(1-2):89–132, 2003. ISSN 0924-1868. doi: http://dx.doi.org/10.1023/A:1024026532471.

Philo Tan Chua, Rebecca Crivella, Bo Daly, Ning Hu, Russ Schaaf, David Ventura, Todd Camill, Jessica Hodgins, and Randy Pausch. Training for physical tasks in virtual environments: Tai chi. In *VR '03: Proceedings of the IEEE Virtual Reality 2003*, page 87, Washington, DC, USA, 2003. IEEE Computer Society. ISBN 0-7695-1882-6.

James Davis and Aaron F. Bobick. Virtual pat: A virtual personal aerobics trainer. In *Workshop on Perceptual User Interfaces*, pages 13–18, 1998.

Doris M. Dehn and Susanne van Mulken. The impact of animated interface agents: a review of empirical research. *Int. J. Hum.-Comput. Stud.*, 52(1):1–22, 2000.

Robert F. DeVellis. *Scale Development: Theory and Applications.* Sage Publications, 2003. ISBN 0-7619-2605-4.

Kevin Dooley. *Social Research Methods, 4th edition.* Prentice Hall, 2001.

Marc Evers and Anton Nijholt. Jacob - an animated instruction agent in virtual reality. In *Advances in Multimodal Interfaces - ICMI 2000*, Lecture Notes in Computer Science 1948, pages 526–533, 2000.

David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach.* Prentice Hall Professional Technical Reference, 2002. ISBN 0130851981.

E.L. Fox and D.K. Mathews. *Fysiologie van lichamelijke opvoeding en sport.* Uitgeverij De Tijdstroom, 1987. ISBN 90-352-1091-3.

Dennis Hofs, Mariët Theune, and Rieks op den Akker. Natural interaction with a virtual guide in a virtual environment: A multimodal dialogue system. *Journal on Multimodal User Interfaces*, 3(1-2): 141–153, March 2010. URL `http://doc.utwente.nl/70362/`. Open Access.

W. IJsselsteijn, Y. de Kort, J. Westerink, M. de Jager, and R. Bonants. Fun and sports: Enhancing the home fitness experience. In *ICEC 2004: Proceedings of the 3rd International Conference on Entertainment Computing*, pages 46–56, Berlin, 2004. Springer Verlag.

W. Lewis Johnson and Jeff Rickel. Steve: an animated pedagogical agent for procedural training in virtual environments. *SIGART Bulletin*, 8(1-4):16–21, 1997. ISSN 0163-5719.

W. Lewis Johnson, Paola Rizzo, Ir. Wauter Bosma, Sander Kole, Mattijs Ghijsen, and ir. Herwin Welbergen van. Generating socially appropriate tutorial dialog. In *Affective dialogue systems : tutorial and research workshop, ADS 2004, Kloster Irsee, Germany, June 14-16, 2004*, volume LNCS 3068 of *Lecture notes in computer science ; 3068*, pages 254–264, 2004. URL `http://doc.utwente.nl/49178/`. ISBN= 3-540-22143-3.

Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Comput. Vis. Image Underst.*, 81(3):231–268, 2001. ISSN 1077-3142. doi: http://dx.doi.org/10.1006/cviu.2000.0897.

Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Comput. Vis. Image Underst.*, 104(2):90–126, 2006. ISSN 1077-3142. doi: http://dx.doi.org/10.1016/j.cviu.2006.08.002.

M. Paleari, C. Lisetti, and M. Lethonen. Valerie: A virtual agent for environmental learning, reacting and emotional interaction. In *Workshop on motivation and affect in educational software, 2005, Amsterdam*, 2005.

Z.M. Ruttkay, J. Zwiers, H. van Welbergen, and D. Reidsma. Towards a reactive virtual trainer. In *Proceedings of the 6th International Conference on Intelligent Virtual Agents*, volume 4133 of *Lecture notes in Computer Science*, pages 292–303, Berlin, August 2006. Springer Verlag.

Zsófia Ruttkay and Herwin Welbergen. Elbows higher! performing, observing and correcting exercises by a virtual trainer. In *IVA '08: Proceedings of the 8th international conference on Intelligent Virtual Agents*, pages 409–416, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85482-1.

Richard M. Ryan and Edward L. Deci. Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary Educational Psychology*, 25(1):54 – 67, 2000. ISSN 0361-476X. doi: DOI:10.1006/ceps.1999.1020. URL `http://www.sciencedirect.com/science/article/B6WD1-45FCGCS-S/2/1a21868a298d5dd9621273184b59c000`.

Aaron Sloman. The cognition and affect project: Architectures, architecture-schemas, and the new science of mind, 2003.

Kristinn R. Thórisson, Hrvoje Benko, Denis Abramov, Andrew Arnold, Sameer Maskey, and Aruchunan Vaseekaran. Constructionist design methodology for interactive intelligences. *A.I. MAGAZINE*, 25(4): 77–90, 2004.

Kristinn R. Thórisson, Gudny Ragna Jonsdottir, and Eric Nivel. Methods for complex single-mind architecture designs. In *AAMAS '08: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, pages 1273–1276, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 978-0-9817381-2-X.

Ning Wang, W. Lewis Johnson, Paola Rizzo, Erin Shaw, and Richard E. Mayer. Experimental evaluation of polite interaction tactics for pedagogical agents. In *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*, pages 12–19, New York, NY, USA, 2005. ACM. ISBN 1-58113-894-6. doi: http://doi.acm.org/10.1145/1040830.1040845.

Michael Wooldridge. *Introduction to MultiAgent Systems*. John Wiley & Sons, 2002. ISBN 047149691X. URL `http://www.worldcat.org/isbn/047149691X`.

K. Yanghee. Empathetic virtual peers enhanced learner interest and self-efficacy. In *Workshop on motivation and affect in educational software, 2005, Amsterdam*, 2005.

Jerrold H. Zar. *Biostatistical Analysis, Fourth Edition*. Prentice Hall, 1999. ISBN 0-13-081542-X.