Proceedings of
Umeå's 16th Student Conference in Computing Science

USCCS 2013

Suna Bensch, Frank Drewes (editors)

UMINF 13.01
ISSN-0348-0542

UMEÅ UNIVERSITY
Department of Computing Science
901 87 UMEÅ, SWEDEN

Preface

Umeå's Student Conference in Computer Science is the highlight of the conference course in our Computer Science curriculum. The idea and objective of the course is to give the students a forum where they can develop their own ideas in a scientific manner, thereby improving their understanding of how research is conducted and how the achieved results should be presented according to scientific standards. The conference format was chosen to provide a comparatively realistic environment in which the research results can be presented.

A student who participates in the course first selects a topic and a number of research questions that he or she is interested in. If the topic is accepted, then the student outlines a paper and composes an annotated bibliography to give a survey of the field. The main work consists in conducting the actual research that answers the questions asked, and convincingly reporting the results in a scientific paper. Each submitted paper receives two or more reviews written by members of the department. If the reviews are favourable, the paper is accepted, meaning that the student is given the opportunity to present his/her work at the concluding conference, and to submit a final version that will be included in the proceedings. The course thus gives an introduction to independent research, scientific writing, and oral presentation.

This offering of the course was the fifteenth. The conference received nine submissions which were carefully reviewed by the reviewers listed on the following page. We are very grateful to the reviewers who did a very good job within a very short time frame. As a result of the reviewing process, seven submissions were accepted for presentation at the conference. Unfortunately, one of them was not revised in time for these proceedings. The remaining six constitute this volume.

Umeå, 2 January 2013

Suna Bensch
Frank Drewes

Organizing Committee

Suna Bensch
Frank Drewes

With special thanks to the reviewers

Jayalakshmi Baskar
Suna Bensch
Martin Berglund
Henrik Björklund
Johanna Björklund
Frank Drewes
Håkan Gulliksson
Thomas Hellström
Lars Karlsson
Juan Carlos Nieves
Ola Ringdahl
Dipak Surie
Niklas Zechner

Table of Contents

Success Factors for E-Sport Games

Oscar Bornemark

Department of Computing Science
Umeå University, Sweden
id08obk@cs.umu.se

Abstract. Some multiplayer video games rise to become successful in the field of e-sports. This paper tries to find out why this is the case. We compare a set of seven video game properties with six established e-sport games in an attempt to find out whether there are properties of multiplayer video games that all established e-sport games have in common.

1 Introduction

E-sport is a general term to describe the play of video games competitively. It is an ever growing phenomenon and is increasing in popularity for each year [1]. Despite of this, little research has been done in the area which makes it an interesting topic to investigate. *Video games* is a generic term for all types of digital games, played and used on some kind of screen. This includes arcade machines, handheld devices, game consoles (such as the Xbox 360 and Playstation 3) and computer games. This paper will focus on the latter two types – games developed for game consoles and desktop computers.

Among the vast amount of multiplayer games developed each year, only a fraction establish themselves as e-sport games. Why do some rise to become the norm among e-sport tournaments while others do not? Although not all multiplayer games strive to become established as e-sport games, it is interesting to know what really makes the difference. Which properties make an e-sport title stand out to other mutiplayer games?

The hypothesis of this paper is that there are properties of multiplayer video games that all established e-sport games have in common. The paper will be structured by first giving some background on the world of video games and e-sport, followed by a set of chosen multiplayer video game properties. It will then in two steps be determined which of these properties are to be found in e-sport games: By investigation of previously done research and by examination of established e-sport games. After a summary, a game evaluation will be done where significant properties will be tested against multiplayer video games *not* considered as e-sport games. The paper will then end with a discussion of the findings.

S. Bensch, F. Drewes (Eds.): Umeå's 16th Student Conference in Computing Science USCCS 2013, pp. 1–12, January 2013.

2 Background

This chapter serves as an introduction into the world of multiplayer video games, competitive gaming and e-sports.

2.1 Multiplayer Games

A multiplayer game is, as the name suggests, a game designed to be played by multiple people at the same time. Board games and other forms of non-digital games designed for multiple people requires more than one active participant for them to be playable. Chess, for example, is meant to, and have to, be played simultaneously by two people. Video games on the other hand have long had the ability to offer themselves as the necessary opponent or counterpart – often in some form of artificial intelligence or pre-designed game structure. Playing a game of Tic-tac-toe on a computer, more often than not it gives the opportunity to face an AI controlled opponent rather than having to find a friend to play with. Hence the concept of multiplayer video games is not as fully self-evident as we first may think.

The first multiplayer video games date as far back as 1958 when the American physicist William Higinbotham developed *Tennis for Two*, a tennis simulator designed for the analog computer Donner Model 30. The game featured two handheld controllers and a five-inch oscilloscope monitor to play on. It is widely regarded as one of the very first electronic games to display motion [9] and is consequently one of the very first multiplayer games since it was designed to be played between two people.

In 1972, the classic Atari game *Pong* made multiplayer games even more popular. Pong laid the groundworks for what has been called The golden age of arcade video games, as it was among the first games to be played in arcade cabinets and simultaneously the first successful commercial video game ever [8]. Both *Tennis for Two* and *Pong* were based on tennis, even though they had different graphical approaches – the former portrayed the tennis court from the side and the latter from above. Multiplayer games and real life sports share a lot of basic properties so it is not unexpected that sport games were among the first multiplayer video games to be designed by game developers.

In the years to follow, a wide variety of multiplayer gametypes have evolved. First person shooters have always been popular, as have real time strategy games. Some games which primarily are made as singleplayer games sometimes ship a multiplayer play mode to broaden their market share. It can also prolongs the game's lifetime, as seen with *Diablo II* (Blizzard Entertainment, 2000).

2.2 Competitive Gaming and E-sport

As multiplay video games became more and more popular, people started to compete against each other in a more serious way. The spread of personal computers made more people able to play and the rapid development of the Internet and game net-code allowed for lower in-game latency between gamers. E-sport mainly focuses on competition and although in general any computer game that allows playing against another could be a possible discipline in e-sport, there are certain core games, which are most popular even from a worldwide perspective [1]. The classic game *Doom* (id Software, 1993) laid the foundation of competitive gaming for the PC and its sequel *Doom II* (1994) was the game of choice for one of the first offline PC gaming tournaments ever: *Deathmatch '95* in 1995. Ever since, tournaments of all sizes have emerged. Some are trying to claim the title as the "world championships", such as *The World Cyber Games*, *Electronic Sports World Cup* and *World eSports Games*, similar to how there are many "world titles" in professional boxing.

The term "e-sports" or "electronic sports" dates back to the late nineties [15]. In Asia, and particularly South Korea, e-sports culture have taken a firm grip of society. Since its release in 1998, the game *StarCraft* (Blizzard Entertainment) has since been the center pillar of gaming in the country, allowing for highly paid professional players and dedicated national television channels, broadcasting the game [15]. Today, e-sport is either recognized or accepted as a sport in over 60 countries around the world, especially in Asian countries[1].

3 Video Game Properties

Video games differ from each other in a wide variety of ways. The game designer has to consider a large number of options when planning, designing and building a new game and since each decision can have a great impact on the finished result it is important to choose the appropriate properties as early as possible. Some of these properties are fundamental for gameplay and game design while others can be seen as mere extra features, implemented to enhance the experience for the players. The more fundamental properties are usually set early in the game design phase because of their large influence on later decisions. *Game genre* and *theme* are two such types of properties. Designing a real time strategy game is a completely different kind of task than designing a sport game or a 3D shooter for example. The less fundamental properties are easier to change, remove or add later on in the development process. Will the player be engaged in some kind of *resource management*? Will the gameplay have *limited time* or are the players free to play as long as they want?

In this section, a set of video game properties will be derived from research made in the area of video game design and video games in general. The properties will be easily recognisable and distinguishable from each other and compose the base material of the game comparison. Each property will be described and explained why it has been chosen.

Symmetry In the scientific area of game theory, a *symmetric game* is one where each player has the same set of actions and each player's evaluation of an

[1] Jesper Thiborg. *eSport and governing bodies – An outline for a research project and preliminary results* (2009), URL: `http://195.178.227.4/bitstream/handle/2043/10746/esport.pdf?sequence=1` Accessed 2012-11-17.

outcome depends only on her action and that of her opponent, not on whether she is player one or player two [10]. This can be applied to video games and e-sports as well. In first person shooters the controllable character often starts with exactly the same preferences and conditions as the opponent's character. It would not matter if suddenly the characters switched, game strategies would still be applicable. In other games, such as real time strategy (RTS) games, this does not always apply. It is common for RTS games to have multiple "character" choices, although here the characters are represented by nations, races, armies or other kinds of larger scale teams. These teams differ from each other in great ways, often consisting of totally different sets of units and buildings. If two players in a multiplayer RTS game each choose a different kind of side to play with the game would not be symmetric.

This can also be applied to the virtual world the game is played in. A virtual football field in a football video game is very symmetric. Maps played in RTS games are usually symmetric, not to favour either side but this is not always the case. Maps played in first person shooters (FPS) on the other hand are almost never symmetric as they often represent real life areas, buildings or other realistic, none-symmetric settings. Still, it is important to note that a game with an asymmetric virtual world still can consist of symmetric gameplay. For example, in the team based FPS *Counter-Strike* two opposing teams with different agendas and conditions face off each other in an asymmetric game world. However, seen over the whole life span of a game the gameplay is still symmetric, based on the teams changing sides halfway in.

Henceforth, when we refer to symmetric games we mean games with symmetric gameplay.

Perfect Information The attribute of perfect information is another element derived from the science of game theory. In a game with perfect information, each player knows everything there is to know about the game at any given time and never will a player know more about the game than another player [4]. A classic example of a game with perfect information is chess, but Go and Tic-tac-toe also possess this characteristic. Card games usually on the other hand are a good example of games with imperfect information – the players never know the whole game state since most of the cards in play are hidden from each other.

In the area of multiplayer video games, there is a clear division of how to perceive the game. Either the opponents share and play on the same computer screen or TV monitor or they have to use their own separate screens. This is often based upon which type of genre the video game belongs to. Sport and fighting games are often played on one single screen, on which the players each control their unit or units. In most other genres, each player competes on their own computer or gaming console, which means they have their own private screen to interact with. These two scenarios represent the two different kinds of game information: Perfect information on shared screens and imperfect information on separate screens.

Time and Space Delimited Most real life sports are limited in both time and space. It could be a football game played on a fixed football field over 90 minutes, or a shot putter, bound to the putter circle and the one minute of preparation before the actual shot. In multiplayer video games this applies as well. Timewise, games can be constrained by adding time limits to different game element, such as rounds or maps. Spacewise, multiplayer video games are usually played in finite game worlds and therefore limited in space, even though some games, such as *The Lords of the Rings Online* (Turbine Inc, 2007) and *Fuel* (Asobo Studio, 2009), consist of game worlds so large they might be considered as "unlimited".

Genre Finding the right genre of a video game can be difficult as games can have crisscrossing criteria for genre determination but also since the very notion of genre is controversial [3]. There are however a few well established genres in the world of e-sports and multiplayer video games (with some already mentioned). Apperley [2] has analysed four major genres, namely *simulation*, *strategy*, *action* and *role-playing*. He also identified different subgenres which in many cases are more applicable than the four very broad, earlier mentioned genres, such as *sport* and *driving simulations*, *real time strategy* (RTS), *turn-based strategy* (TBS), *first-person shooters* (FPS) and *massive multi-player online role-playing games* (MMORPG). He fails to identify *fighting games* as a subgenre but it could be assumed belonging to the action genre. Another subgenre not mentioned by Apperley is the comparatively new *multiplayer online battle arena* (MOBA) [12], which consists of a mixture of RTS and RPG elements. It was made popular after the release of the *WarCraft 3* modification *Defense of the Ancients* in 2005.

Spectator Mode Spectatorship have from the earliest days of video games had a central role [14]. Just as people stood in line to observe great arcade players battle each other during the heydays of the arcade machine, people still find it fascinating, entertaining and interesting to watch great computer and console gamers do what they to best. E-sport has definitely gained legitimacy as a viable spectator sport [13, 7]. Earlier this year, 567 000 simultaneous viewers watched the final match of the *Dota 2* (Valve Cooperation, 2011) tournament "The International 2", streamed via the Internet[2]. An astonishing feat.

One of the reasons Valve could achieve this impressive feat was due to the built in spectator mode. The game allows users to watch active matches played by others through the game itself. The users can then use a spectator interface to cycle through all the active players, see their stats and equipment and follow their movements and actions. All in real time. Not so elaborated spectator modes can be found in other games, where users can join ongoing matches and join a "spectator team".

E-sport Pro-active Developers A big difference between e-sports and real life sports is the concept of ownership. While no company or individual can

[2] blog.dota2.com. *The 2012 International* (2012), URL: `http://blog.dota2.com/2012/09/the-2012-international/` Accessed 2012-12-08.

claim ownership of a real life sport (not to be confused with governing bodies such as *FIFA* or *The International Olympic Committee*), video games on the other hand are with very few exceptions developed, maintained and owned by people and companies. It is therefore all up to the developer to modify the rules and surroundings of the game until it reaches success. Some games are designed strictly with e-sport in mind[3] and the best way for such games to become successful and achieve profit is to be recognized and established as an e-sport game to a wide audience. For this to happen, the developer can release the game, sit back and hope for it to rise to fame by its own. The developer can also try to be pro-active in the e-sport community. One way of doing this is by organizing and funding e-sport tournaments with the expectation of the game getting a high boost of player and media coverage and attention.

It should be noted that strictly speaking, this attribute is not a game property but rather an outside influence affecting the outcome of the game. Nevertheless, it has been taking into account because of its considerable impact on whether or not a game becomes a successful e-sport title.

When evaluating this property, a game will be considered having an e-sport pro-active developer if the developer have either organized or funded some kind of public e-sport tournament with the game after its release.

Eye-hand Coordination and Reflex Based If a gamer is choosing a new multiplayer video game to play, one of the first things to consider may be the gameplay the game is offering. Some games are meant to be played at a slow pace, such as *The Settlers* (Blue Byte Software, 1993) and *Plants vs Zombies* (PopCap Games, 2009) while others are extremely fast and intensive, such as *Quake 3* (id Software, 1999) and *Need for Speed: Hot Pursuit* (Criterion Games, 2010). Fast games usually requires a high coordination and reaction time capacity, which in its self is no easy task. Combined with controllers with up to 13 buttons[4], it makes it even harder. This has led to video game players possessing decreased reaction times and increased hand-eye coordination and augmented manual dexterity than non-gamers [5].

Considering this wide variety of different gameplay approaches it is interesting to study which side of the span e-sport games ranks – the quick, reflex based side with high requirements of eye-hand coordination, or the slower side, allowing for more elaborate strategic thinking and laid back decisions.

4 Properties of E-sport Games

Seven properties found in multiplayer video games have been identified. The next step is to determine which of these properties are also found in e-sport

[3] Dustin Browder. *The Game Design of StarCraft 2: Designing An E-Sport* (2009), URL: `http://gdcvault.com/play/1014488/The-Game-Design-of-STARCRAFT` Accessed 2012-12-03.

[4] Microsoft.com. *Xbox 360 Controller*, URL: `http://www.microsoft.com/hardware/en-us/p/xbox-360-controller-for-windows` Accessed 2012-12-04.

games. This will be done in two steps: By investigate previous research and by examination of established e-sport games. Two of the identified properties, *time and space delimitation* and *Eye-hand coordination and reflex based*, will be the subject of the former kind of investigation, while the other five will be the subject of the latter.

4.1 Previously Done Research

E-sports is unfortunately sparsely covered by the scientific community. It is a relatively new phenomenon and there is much research to be done in the area. However, useful sources of information concerning the two previously mentioned preferences have been found. Jonasson and Thiborg [6] argues that a good e-sport game, where the players are competing individually or in a team, is delimited in time and space. They do not elaborate further on this but it can be assumed that the opposite holds as well – games *not* delimited in time and space are *not* suitable for e-sports.

Jonasson and Thiborg also give evidence of the importance of eye-hand coordination and reflexes when playing e-sport games. They argue that e-sport players rely on physical skills and require rapid and accurate coordination between the hands and the eyes. They even claim that no real life sport requires such a diversified coordination of the fingers as e-sports [6]. Jonasson and Thiborg are not the only ones highlighting this property. Rai and Yan [11] have studied the next generation of e-sports infrastructure and conclude the following:

> *"Usually, e-Sports put a higher demand on the players' abilities than common computer games, not only superb coordination capacity between hands and eyes, rapid response capacity and skilful handling capacity of mouse and keyboard, but also complex strategic and tactical thinking ability."*

4.2 Established E-sport Games

In this section, five of the identified properties – *symmetry*, *perfect information*, *genre*, *spectator mode* and *e-sport pro-active developers* – will be investigated. This will be done by a study of six established e-sport games – *StarCraft II* (SC2) (Blizzard Entertainment, 2010), *Counter-Strike: Global Offensive* (CS:GO) (Valve Corporation, 2012), *Halo 4* (343 Industries, 2012), *Super Street Fighter IV* (SSF4) (Capcom, 2010), *League of Legends* (LoL) (Riot Games, 2009) and *FIFA 13* (Electronic Arts, 2012). These six games are all commonly found at the highest level of gaming competition and regularly chosen by big e-sport event organizers.

Most of these games are follow-ups of earlier released copies in the same game series. For example, the FIFA game series is getting an addition every year by Electronic Arts (a tradition dating back to 1993[5]), providing mostly

[5] Britannica. *FIFA (Electronic game series)*, URL: `http://www.britannica.com/EBchecked/topic/1468273/FIFA` Accessed 2012-12-08.

small gameplay updates and enhanced graphics. StarCraft II was released twelve years after its predecessor but still heavily relied on the same type of game mechanics and virtual world. Therefore, because of these often minor changes of fundamental gameplay between versions in a single game series, earlier instances of the six games have been taken into consideration if needed.

The findings of the properties in the six chosen e-sport games are summarized in Table 1.

	SC2	**CS:GO**	**Halo 4**	**SSF4**	**LoL**	**FIFA 12**
Symmetric	No	Yes	Yes	No	No	Yes
Perfect information	No	No	No	Yes	No	Yes
Genre	RTS	FPS	FPS	Fighting	MOBA	Sport
Spectator mode	Yes	Yes	No	Yes	Yes	No
E-sport pro-active	Yes[6]	Yes[7]	No	Yes[8]	Yes[9]	Yes[10]

Table 1. The properties combined with six established e-sport games

No funded or organized events were found when researching Halo 4 and its developer 343 Industries.

4.3 Summary and Analysis

From a scientific point of view, it seems clear that a good e-sport game requires high levels of eye-hand coordination and fast reflexes of those performing. The importance of highly developed fine motor skills in the fingers also stands out as a must-have criteria. It is not as clear whether a good e-sport game has to be delimited in time and space, but according to Jonasson and Thiborg that seems to be the case. The problem is that the property does not fit completely if compared with the six chosen established e-sport games. They are all delimited in space – all games are played on confined areas or maps – but they are all not delimited in time. StarCraft II and League of Legends matches do not have a set time limit and could in theory be played forever (but in reality are finished

[6] Battle.net. *The StarCraft II World Championship Series*, URL: `http://eu.battle.net/sc2/en/blog/4012602` Accessed 2012-12-08.

[7] Counter-Strike.net Blog. *CS:GO Community Event at PAX*, URL: `http://blog.counter-strike.net/index.php/2012/08/4795/` Accessed 2012-12-08.

[8] Streetfighter 25th Anniversary. *25th Anniversary Tournaments*, URL: `http://www.streetfighter25.com/en/tournaments/` Accessed 2012-12-08.

[9] LeagueofLegends.com. *League of Legends Season Two*, URL: `http://competitive.eune.leagueoflegends.com/competitive/season-2/introduction` Accessed 2012-12-08.

[10] EA Sports Arena. *Play for cash*, URL: `http://www.easports.com/arena` Accessed 2012-12-08.

mostly within 10 to 45 minutes). It is unsure what Jonasson and Thiborg would say if presented with these examples.

Moving on to the property evaluation with the help from the six e-sport titles, all in all it is hard to draw any real hard conclusions. Half of the games presented *symmetric gameplay* while the other half did not. It other words, this property does seem to have little or no impact whether a multiplayer video game will work as a e-sport game or not. Concerning *perfect information*, only two out of the six games – Super Street Fighter 4 and FIFA 12 – possessed this property but since they both for a long time have shown great success in the e-sport field, it seems like even this property have no great impact. The common denominator for the two games is the shared screen between the competitors, a trait not interfering with any real aspect of e-sports.

The *genre* property is more interesting since except from CS:GO and Halo 4 which are both first person shooters, it differs in all games. This is probably because of the e-sport community's wish to have a diverse set of games to watch and play. The two FPS titles have even separated themselves by being played on different hardware (CS:GO on computers and Halo 4 on home consoles) so basically they are not interfering with each other. Four of the six games had an implemented *spectator mode*. It is unclear if this is a vital property for a successful e-sport title but if taking into account earlier statements concerning video game spectators [14, 13, 7] it is clear that spectators play a great role in today's e-sport. At "My Game Idea", a website where Electronic Arts let people post ideas about their games for others to rate on, "Spectator Mode" is the fourth highest rated idea among all FIFA ideas[11], showing how highly coveted this property is in games lacking it.

The last property is the most dominating of the five. Five out of the six titles have *e-sport pro-active developers* with Halo 4 as the only exception. Some of the developers are investing huge amounts of money into various tournaments. Riot Games, the fathers of League of Legends, is perhaps the best example of a developer dedicated to see their game being successful in the e-sport scene. This year they invested in a prize pool of a total of five million dollars, the largest in e-sport history[12], in an attempt to establish League of Legends as the most popular e-sport title to date. Blizzard Entertainment have done the same by arranging what they call "The 2012 Battle.net World Championship", a worldwide StarCraft II tournament with a prize pool of 250 000 dollars. These kinds of tournaments are a way for the developers to put a game in the spotlight, attracting more players and media attention. If the game is newly released it also works as a great kick start into the e-sport scene, overshadowing other competitors.

In summary, it seems like symmetry and offering perfect information is of little importance when designing games intended for e-sport. The genre of the

[11] EA Sports. *My Game Idea*, URL: `http://www.easports.com/gamechangers/ideas?campaignId=27784` Accessed 2012-12-09.

[12] League of Legends. *League of Legends Season Two*, URL: `http://competitive.na.leagueoflegends.com/competitive/season-2/introduction` Accessed 2012-12-09.

game can be of importance if the genre already is occupied by a popular title. Other than that, the type of genre appears to be insignificant. An implemented spectator mode seems advantageously to have, considering the existing and growing numbers of e-sport spectators, but not vital. Perhaps the most important property of a multiplayer video game with the ambition to be a successful e-sport game is having an e-sport pro-active developer, willing to fund and/or organize e-sport events and other e-sport happenings.

5 Game Evaluation

Three out of the seven chosen properties have been found significant – *Spectator mode*, *e-sport pro-active developers* and *eye-hand coordination and reflex based* should all be positive to increase the chances for an e-sport game to be successful. Here are the tree properties tested according to three multiplayer video games *not* established within e-sport. They games tested are *Civilization V* (Civ V) (Firaxis Games, 2010), *Minecraft* (Mojang, 2011) and *Diablo III* (Blizzard Entertainment, 2012). The results are shown in Table 2.

	Civ V	**Minecraft**	**Diablo III**
Spectator mode	No	No	No
E-sport pro-active	No	No	No
Coordination and reflexes	No	Yes	Yes

Table 2. The three most important properties combined with three multiplayer video games not established in e-sports

As seen, only one property is found in the three games. The gameplay of Minecraft and Diablo III is based in eye-hand coordination and reflexes but the titles are lacking both spectator modes and an e-sport pro-active developer (even though Blizzard Entertainment as shown is very active with their other title, StarCraft II). Civilization V lacks all the three properties.

6 Discussion

One of the main problems during this research was to choose appropriate properties to compare with. As there are almost an infinite amount of properties to choose from, finding a small set of distinguishable and easy comparable ones was hard. There might be another set of properties that show no difference what so ever between the established e-sport titles and the three none established games. As there has been little research done in the field of e-sports compared to ordinary multiplayer video games, some statements are unfortunately lacking proper citations, but maybe this paper can serve as an introduction to the topic. It should also be noted that evaluating only six e-sport games probably gives a too small statistical basis for drawing trustworthy conclusions. However, achieving a satisfactory basis is hard as the amount of established e-sport games is limited – this winter the well known computer and e-sport festival *DreamHack* held nine official tournaments[13], which of five was compared in this paper.

The results are in a way surprising. It seems like the properties derived from game theory have little or no impact whether a game will make it as an e-sport game or not. What really matters is the dedication from the developer, and in the long run, money. An organized event with a huge prize pool can function as an injection to the e-sport scene and a way of buying your way ahead of the competitors so to speak. This may feel a bit unfair, as one might think a good e-sport game should rise to fame by its own. Although, if the game is not fit for e-sport, these kind of injections will probably only work as a form of artificial respiration.

The hypotheses of this paper – there are properties of multiplayer video games that all established e-sport games have in common – seems hard to validate, but matches in some cases.

References

1. Adamus, T.: Playing computer games as electronic sport: In search of a theoretical framework for a new research field. In: Fromme, J., Unger, A. (eds.) Computer Games and New Media Cultures, pp. 477–490. Springer Netherlands (2012)
2. Apperley, T.H.: Genre and game studies: Toward a critical approach to video game genres. Simulation and gaming 37(1), 6–23 (2006)
3. Arsenault, D.: Video game genre, evolution and innovation. Eludamos. Journal for Computer Game Culture 3(2), 149–176 (2009)
4. Fundenberg, D., Tirole, J.: Game Theory. Massachusetts Institute of Technology (1991)
5. Green, C.S., Bavelier, D.: The cognitive neuroscience of video games. In: Messaris, P., Humphreys, L. (eds.) Digital media: Transformations in human communication, pp. 211–224. Peter Lang (206)
6. Jonasson, K., Thiborg, J.: Electronic sport and its impact on future sport. Sport in Society 13(2), 287–299 (2010)
7. Lee, D., Schoenstedt, L.J.: Comparison of esports and traditional sports consumption motives. Journal of Research 6(2), 39–44 (2011)
8. Lowood, H.: Videogames in computer space: The complex history of pong. IEEE Annals of the History of Computing 31(3), 5–19 (2009)
9. Nyitray, K.J.: William alfred higinbotham: Scientist, activist, and computer game pioneer. IEEE Annals of the History of Computing 33(2), 96–101 (2011)
10. Osborne, M.J.: An Introduction to Game Theory. Oxford University Press (2009)
11. Rai, L., Yan, G.: Future perspectives on next generation e-sports infrastructure and exploring their benefits. International Journal of Sports Science and Engineering 3(1), 27–33 (2009)
12. Ratan, R., Kennedy, T., Williams, D.: League of gendered game play behaviors: Examining instrumental vs identity-relevant avatar choices. In: Meaningful Play 2012 (2012)

[13] DreamHack. *Official DreamHack Winter 2012 Tournaments*, URL: `http://www.dreamhack.se/dhw12/esport/` Accessed 2012-12-28.

13. Su, N.M., Shih, P.C.: Virtual spectating: Hearing beyond the video arcade. In: Proceedings of the 25th BCS Conference on Human-Computer Interaction. pp. 269–278. British Computer Society (2011)
14. Taylor, T.: Raising the Stakes: E-Sports and the Professionalization of Computer Gaming. MIT Press (MA) (2012)
15. Wagner, M.G.: On the scientific relevance of esport. In: Proceedings of the 2006 International Conference on Internet Computing and Conference on Computer Game Development. pp. 437–440. CSREA Press (2006)

The Genetic Algorithm vs Alpha-Beta Algorithm When Applied to Othello

Karen Farnes

Department of Computing Science
Umeå University, Sweden
mcs11kfs@cs.umu.se

Abstract. The Alpha-beta algorithm is a commonly used algorithm in two-player zero-sum games, such as Othello, for making AI-players. This article presents the application of the Alpha-beta algorithm and the genetic algorihtm for Othello with a set of commonly used strategies. The genetic algorithm is in many ways a contrast to the more structured Alpha-beta algorihtm, where the results often appear more as if by magic. The work presented herein has implemented and run teo AI-players for Othello using both algorithms. The result indicates that the genetic player outshines the Alpha-beta player when they play against each other.

1 Introduction

The Alpha-beta algorithm is a well-known and well used algorithm for making AI-players in games. While Alpha-beta algorithm is a straight forward and understandable algorithm, it is interesting to compare a more "natural" algorithm like the genetic algorithm to the Alpha-beta algorithm. The idea of the genetic algorithm is to mimic nature. Which of these two algorithms works best? The designed and structured Alpha-beta or the by nature inspired genetic algorithm that in many ways works as if by magic? In an attempt to answer this question and learn something about the relationship between the two algorithms we choose to apply the algorithm to the game of Othello, and present the following statement: *An AI-player implementing the genetic algorithm will in most cases win over a AI-player implemented with an Alpha-beta algorithm in the game of Othello.* A verification or lack of verification of this statement is a part of answering a bigger question, namely, are the many developers quest to create computers that behave as nature like as possible justified?

This section continues with a short presentation of similar work, before an introduction to the game of Othello, the Alpha-beta algorithm and the genetic algorithm is given. The next sections presents the implementation of Othello and the AI-players related to the two algorithms in Sect. 2, before Sect. 3 introduce a set of test cases and the result from running the test cases. In Sect. 4 the article ends with a discussion about the result and the algorithms. At last the conclusions are presented together with possible future work.

S. Bensch, F. Drewes (Eds.): Umeå's 16th Student Conference in Computing Science USCCS 2013, pp. 13–25, January 2013.

1.1 Similar Work

Most of the similar work done on this subject addresses the challenge of making a good AI-player by combining multiple algorithms. The work of [1, 2] concerns combining the Alpha-beta algorithm and the genetic algorithms in a single Othello-player, where the Alpha-beta algorithm is using the genetic algorithm for optimizing the evaluation function. Other works that deals with combination of algorithms are articles that use neural networks together with the genetic algorithm [3–5].

1.2 Othello

This section gives an introduction to Othello based on [6, 1]. The introduction contains information about the rules of the game, information about general strategies and desirable positions that can be used in an evaluation function.

Othello, also known as Reversi, is a fairly complex two-person zero-sum strategy board game, similar to Go, Checkers and Chess. Though the rules are simple to learn, the game still requires high level strategies to master.

Rules The game is played by two players (*Black* and *White*) on an 8x8 board. Initially the game starts with each player having two discs already on the board. White player has discs on position $(4, 4)$ and $(5, 5)$, while black's discs are at $(4, 5)$ and $(5, 4)$, as shown in Fig. 1.

```
   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
---|---|---|---|---|---|---|---|---|---
 1 |   |   |   |   |   |   |   |   | 1
---|---|---|---|---|---|---|---|---|---
 2 |   |   |   |   |   |   |   |   | 2
---|---|---|---|---|---|---|---|---|---
 3 |   |   |   |   |   |   |   |   | 3
---|---|---|---|---|---|---|---|---|---
 4 |   |   |   | 0 | X |   |   |   | 4
---|---|---|---|---|---|---|---|---|---
 5 |   |   |   | X | 0 |   |   |   | 5
---|---|---|---|---|---|---|---|---|---
 6 |   |   |   |   |   |   |   |   | 6
---|---|---|---|---|---|---|---|---|---
 7 |   |   |   |   |   |   |   |   | 7
---|---|---|---|---|---|---|---|---|---
 8 |   |   |   |   |   |   |   |   | 8
---|---|---|---|---|---|---|---|---|---
   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
```

Fig. 1. The initial board of Othello where 0 represents white discs and X black discs.

The players take turns making a move, starting with the black player. A move is defined by placing a disc on an empty square adjacent to at least one disc of opposite color. Further, there must exist a disc of the same color in a continuously line of opposite colored discs from one of those adjacent laying discs. If such a valid move is made, all discs on this line are flipped to the color of the player's disc color. Fig. 2 shows such a valid move.

A game is over when neither of the players can make a valid move. This usually happens when there are discs on all 64 squares of the board. The player with most discs on the board at the end of the game wins.

```
   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
---|---|---|---|---|---|---|---|---|---
 1 |   |   |   |   |   |   |   |   | 1
---|---|---|---|---|---|---|---|---|---
 2 |   |   |   | 0 |   |   |   |   | 2
---|---|---|---|---|---|---|---|---|---
 3 |   |   |   | 0 |   |   |   |   | 3
---|---|---|---|---|---|---|---|---|---
 4 |   |   |   | 0 | X |   |   |   | 4
---|---|---|---|---|---|---|---|---|---
 5 |   |   | 0 | 0 | X |   |   |   | 5
---|---|---|---|---|---|---|---|---|---
 6 |   |   |   |   | X |   |   |   | 6
---|---|---|---|---|---|---|---|---|---
 7 |   |   |   |   |   |   |   |   | 7
---|---|---|---|---|---|---|---|---|---
 8 |   |   |   |   |   |   |   |   | 8
---|---|---|---|---|---|---|---|---|---
   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
```

```
   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
---|---|---|---|---|---|---|---|---|---
 1 |   |   |   |   |   |   |   |   | 1
---|---|---|---|---|---|---|---|---|---
 2 |   |   |   | 0 |   |   |   |   | 2
---|---|---|---|---|---|---|---|---|---
 3 |   |   |   | 0 |   |   |   |   | 3
---|---|---|---|---|---|---|---|---|---
 4 |   |   | X | X | X |   |   |   | 4
---|---|---|---|---|---|---|---|---|---
 5 |   |   | 0 | X | X |   |   |   | 5
---|---|---|---|---|---|---|---|---|---
 6 |   |   |   |   | X |   |   |   | 6
---|---|---|---|---|---|---|---|---|---
 7 |   |   |   |   |   |   |   |   | 7
---|---|---|---|---|---|---|---|---|---
 8 |   |   |   |   |   |   |   |   | 8
---|---|---|---|---|---|---|---|---|---
   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
```

Fig. 2. Black making a valid move by laying a disc on $(4, 3)$

Strategy To be able to master the game of Othello, it is necessary to have a good strategy. A key element in strategy is to know which positions on the board are the most desirable to hold.

Number of Discs The strategy involves trying to have the most number of discs throughout the game. This is the strategy most commonly followed by beginners. The idea is that if the player has maximized its number of discs on the board, the opponent will have a minimal number of discs. The problem is that if a player has a lot less discs than its opponent it really means that the player has a lot more possibilities than the opponent. So, in a lot of cases, the opponent is able to turn the game around, and end up with the most number of discs at the end of the game.

Corners Corners($(1, 1)$,$(1, 8)$,$(8, 1)$ and $(8, 8)$) are examples of what we call stable squares. They are called stable because they cannot be flipped[1]. This makes these squares extremely desirable to hold.

Edges As with corners, edge squares are desirable because of the property of stability. Occupying an edge square does not automatically make the disc stable, but there is a high probability that it will remain stable. Edge squares are therefore desirable, but to a lesser extent than the corners.

1.3 Alpha-Beta Algorithm

This section gives an introduction to the Alpha-beta algorithm based on information collected from [7]. This section also contains pseudo code for the algorithm and a list of advantages and disadvantages.

Alpha-beta is a commonly used algorithm in Artificial Intelligence for games like Othello, for searching through the game tree. Alpha-beta is what we call a min-max algorithm, meaning that as the algorithm traverses through the tree, it will alternate between choosing the path with the maximum and the minimum value (See Sect.*Heuristic Function*). And in this way simulating both the players

[1] Change the color of a disc on the board.

Algorithm 1 AlphaBeta(state)

```
v = MaxValue(state,−∞,+∞)
return move in validMoves(state) with value v
```

Algorithm 2 MaxValue(state,α,β)

```
if depth=searchdepth then
  return Evaluate(state)
v= −∞
for each m of validMoves(state) do
  nextState=Result(state,m)
  v=Max(v,MinValue(nextState,α,β))
  if v≥ β then
    return v
  α =Max(α,v)
return v
```

Algorithm 3 MinValue(state,α,β)

```
if depth=searchdepth then
  return Evaluate(state)
v= +∞
for each m of validMoves(state) do
  nextState=Result(state,m)
  v=Min(v,MaxValue(nextState,α,β))
  if v≤ α then
    return v
  β =Min(β,v)
return v
```

playing perfectly (choosing the best move according to the heuristic). Alpha-beta optimizes the standard min-max by using branch cutting, so that the search is done more quickly. Pseudo code for Alpha-beta search can be found in Algorithms 1, 2 and 3 [8].

Heuristic Function When the algorithm reaches a given search depth, the algorithm does an evaluation of the state of the game. The function for evaluation is called the heuristics function. What this functions consists of depends on the game to which the algorithm is applied. It is the task of the persons that are using the Alpha-beta to ensure that they come up with a good heuristic. This means that the quality of the algorithm is not only depending on the algorithm itself, but also on the developers. One of software developments biggest challanges are the humans involved, and Alpha-beta algorithm is no exception.

Advantages and disadvantages of the Alpha-beta algorithm are [7]:

Advantages
- The algorithm is simple.
- "Clear separation of search and knowledge"
- Low complexity

Disadvantages
- Time limitations lead to limited search depth.
- Limited search depths may cause selection of the wrong paths, because important information may remain unexplored.
- Correctness of the heuristics.
- Knowledge compressed into a scalar number, leads to information loss.
- Assumes the opponent plays perfectly, while it may have be using a better heuristic which leads to other solutions.
- Confuse a blunder with a sacrifice.

1.4 Genetic Algorithm

As with the Alpha-beta algorithm, an introduction to the genetic algorithm is presented, this time based on information collected from [1, 9]. In addition to pseudo code for the algorithm, a list of advantages and disadvantages of the algorihtm are listed.

The genetic algorithm (from here on referred to as GA) is an evolutionary algorithm inspired by how a population develops and adapt over time in nature, through natural selection. Each individual in the population is represented by a string. When the algorithm starts a random set of individual strings are made and passed through a routine of selection, reproduction and mutation for a number of generations. Pseudo code can be seen in Algorithm 4 and 5 [8].

Algorithm 4 GA(population,*FitnessFunc*)

```
while some individuals are fit enough or enough time has elapsed do
  new_population=empty set
  for i = 0 to Size(population) do
    x=Selection(population,FitnessFunc)
    y=Selection(population,FitnessFunc)
    child=Reproduce(x,y)
    if small random probability then
      child=Mutation(child)
    add child to new_population
  population= new_population
return individual in population with best fitness
```

Algorithm 5 Reproduce(x,y)

```
n=Length(x)
c=random number between 1 to n
return Append(Substring(x,1,c),Substring(y,c+1,n))
```

Selection For each generation there are made a selection of strings based on a fitness value. How we measure the fitness of each individual will depend on the context to which the algorithm is applied. Further, there are several ways to select individuals based on their fitness. For example one can randomly set pairs of strings against each other and choose those strings that win their match. Another commonly used method is to directly choose strings according their fitness value. Of course it is natural to choose those strings with the best fitness value, but it can also be a good practice to select strings according to a probability related to the fitness value. By doing this, one ensures that the search does not narrow down its exploration to an unfavorable path.

Reproduction To produce the strings for the next generations, crossover is used between a given number of pairs of strings. The crossover may happen on one or more randomly picked places in the strings. The parents may produce one or two children. Fig. 3 shows an example of a one point crossover. The probability that a string applies crossover is typically set to around 60%.

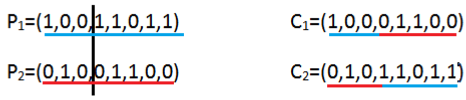

Fig. 3. Example of two parent string making two children with the help of one point crossover

Mutation To ensure that there is at least a small probability that all parts of the search space can be reached, mutation is used. Mutation is simply flipping a bit in a string. The probability of this is generally as low as 1%.

Advantages and disadvantages of the genetic algorithm are [1, 9]:

Advantages

- Quite easy to understand the idea behind the algorithm.
- Algorithm can be parallelized.
- Easily modifiable to solve different problems
- Tolerant to noise

Disadvantages

- Hard to understand why it works or not.
- Can become computational heavy.
- Can be difficult to find out how to represent a problem.
- Multiple parameters that is hard to set.

2 Implementation

The implementation is based on a program developed by the author in the course “Artificial Intelligence - Methods and Applications at Umeå University, where some of the skeleton code was made by course teacher Henrik Björklund. The implementation is written in JAVA, and all interaction is done by standard output and input.

2.1 Othello Program

To keep track of the state of the board, the Othello program stores board information in the class *OthelloPosition*. In addition to storing the current board, the class is able to print the board, say which players has the next turn, find all possible moves and conduct a given move. The class *OthelloAction* contains information about a move on the board. This leads us to the *Engine*. Each time a player has its turn, it calls the *Engine* to find the best move to do next. How this is done, is of course depending on the algorithm used. A general pseudo code for *Engine* can be found in Algorithm 6.

Algorithm 6 Engine(OthelloPosition *op*)

```
moves=op.getAllPossibleMoves()
if moves.SIZE=0 then
   return OthelloAction indicating a pass move
for all m in moves do
   make move m
   e = evaluate(m)
   reset board
return m with highest e
```

Game Play To play a game, two engines are made to represent each player and a *OthelloPosition* is initialized to represent the game board. The game is then played, with each player taking turns evaluating the board and making the move it finds to be the best. The game ends when both players make a pass move. The program then counts the number of discs each player has on the board and announces the winner.

2.2 Alpha-Beta Player

To evaluate which move is the best, the Alpha-beta player uses the Alpha-beta algorithm presented in Sect. 1.3. The algorithm can be given different heuristic functions to work with. The only rule is that the heuristic value should be between -100 and 100, where negative values indicate an advantage for the opposite player. The evaluator used for the remainder of the article is based on the desirability of those discs that each player holds. The values that the evaluator works with are these:

- Corner pieces are worth 12.5.
  - Make up 1/2 of the maximum value of 100
- Edge pieces are worth 1.39.
  - Make up 2/6 of the maximum value of 100
- Middle pieces are worth 0.46.
  - Make up 1/6 of the maximum value of 100

The evaluator adds up the total value for each player, and find the heuristic value by subtracting the current players value with the opponents value.

The values used are based on the knowledge of their importance for the game as presented in Sect. 1.3. Though there are a number of ways of making the evaluator more advanced, this method is far better than just using the most naive approach (simple counting the number of disc each player posess). This notion stems from previous (but not documented) work done by the author. Because this article looks at the basics of the algorithm when applied to Othello, we choose to settle with this quite simple heuristic.

2.3 Genetic Algorithm Player

The implementation of the genetic player is based on a paper written by Tucker Cunningham [10]. In contrast to the Alpha-beta player where we already have somewhat of an idea of which squares are desirable, we here try to find, without prior knowledge, which squares on the board that are the most desirable to hold. The idea is to assign weights to the different squares, and to find which set of weights works best with the use of a genetic algorithm. The weights lie between -2 and 2, and are supposed to represent the relative value to hold between the squares.

Each time it is the genetic player's turn to make a move, the genetic algorithm runs a new evaluation of the squares. The genetic player chooses next move by evaluating each valid move and picking the move with highest value. The evaluation counts the disc relation between the the player and its opponent. Each disc is worth as much as the weight for the square it lays on.

Weights In an effort to decrease the computational cost, we utilize the fact that the board has a rotational symmetry. This means that we only need 10 weights instead of 64 weights. Fig. 4 shows have these weights are placed on the board.

1	2	4	7	7	4	2	1
2	**3**	5	8	8	5	3	2
4	**5**	**6**	9	9	6	5	4
7	**8**	**9**	**10**	10	9	8	7
7	8	9	10	10	9	8	7
4	5	6	9	9	6	5	4
2	3	5	8	8	5	3	2
1	2	4	7	7	4	2	1

Fig. 4. Representation of 10 different positional weights on the board

Population A population consist of individuals, where each individual is represented by an array of size 10. Each element in this array is a weight for each position on the board (See Fig. 4). Each time the genetic algorithm is run, a new randomly made population is constructed. The algorithm then runs for a given number of generations.

Fitness To find the fitness for each individual in the population a full game to its end is played against an Alpha-beta player, starting at the current state of the board. This means that the same position is played multiple times against a Alpha-beta player with different weights. The fitness value represents how good or bad the individual did in the game. The value expresses the number of discs the individual has on the board in relation to the opponent at the end of the game. If the individual has 34 discs and the opponent 29, the fitness value will be 5.

Selection A selection of individuals from a population is done by finding the fitness of all the individuals. The algorithm selects an individual for the next generation based on a random draw with a given probability for each individual based on its fitness. Individuals with higher fitness, has a greater possibility of getting picked than those with lower fitness. Pseudo code for the selection can be seen in Algorithm 7.

Algorithm 7 Selection(Population *pop*)

```
Shift fitness values so all are positive.
while true do
  for all p in pop do
    if RandomNumber(0,100) < p.fitness() / (sum of all fitnesses) *100 then
      newPop.add(p);
      if newPop.SIZE=pop.SIZE then
        return newPop
```

Recombination The algorithm simply goes through pairs of individuals taken from the selection and with a probability does a crossover between these individuals. The implementation uses one point crossover at a randomly choosen point. The probability of producing two offsprings from two parents is set to 80%

Mutation The mutation is done by a probability of 10%. A mutation involves flipping a random bit in the weight-array.

3 Results

To find out which of the algorithm works best, we simply let the two types of players play a game against each other 10 times. A problem with doing this is that both algorithms are effected by a set of parameters that greatly impact how well the algorithm works. This is especially the case for Alpha-beta algorithm in relation to its search depth. For GA it is less clear to which extent the different parameters affect the result. Based on parameters suggested in [10], we choose the setup for the GA. [10] suggests a range of parameters, and for population and number of generation the minimum suggested values are chosen to minimize the computational cost.

- Population: 100
- Generations: 30
- Probability of crossover: 80%
- Probability of mutation: 10%

What we are left with are the search depth for Alpha-beta algorithm, both for the real player and the training player for the GA. Early tests of GA shows that the algorithm is very computationally demanding, and we therefore limit our self to testing three different cases of search depths. In the first test case we choose a low search depth of 4, for training and real play. In the second case we use a higher search depth and therefore we increase the search depths to 8. In the third and last case we set the real Alpha-beta players search depth to 8 while we let the Alpha-beta player that the GA trains against have a search depth of only 4. If the GA player is trained against a player with search depth 8, it knows what is coming. If it is allowed to only train against less for seeing player than what it is really playing against, we can check the robustness of the algorihtm and see how well it works in a more general case.

3.1 Test Results

Table 1 shows the results from running the test cases 10 times each.

Table 1. The result from the test cases

Test case	Search Depth AB		AB win	GA win
	Real	Training		
1	4	4	1	9
2	8	8	2	8
3	8	4	3	7

For all test cases the results show that the GA is superior to the Alpha-beta algorithm. As we increase the search depth (which favors Alpha-beta) we see that GA become slightly less superior, but is still wins 70% of the games.

4 Discussion

In this section we will present a discussion over several aspects of the results, ending with a conclusion and suggestions of future work.

4.1 Search Depth

As mentioned, the results favor the GA over the Alpha-beta algorithm. The result does however indicate that the GAs advantage decrease as the search depth of the Alpha-beta algorithm increases. When working with an Alpha-beta algorithm where we, for the sake of simplicity, refrain from changing the heuristic, the only parameter that strengthens the algorithm is the search depth. Hence, it is not surprising that with a bigger search depth, the algorithm performs better compared to the GA, but as test case 2 indicates, as long as the GA is allowed to train against an Alpha-beta player with the same search depth, it will have little problem defeating it later on.

4.2 The Genetic Algorithm

The real strength of the GA is shown in test case 3. In this case the GA is restricted to train its values against an Alpha-beta opponent with much lower search depth than the one it is really playing against, and still it is winning 70% of the games. Through generations of recombination and mutations, the genetic algorithm still works its magic with training against a less for seeing opponent than in the real game.

4.3 Gamers Luck

One does not need a lot of experience with playing games to realize that even a novice player can get lucky and occasionally win against a more experienced player. Based on that, it can easily be assumed that the single win of the Alpha-beta in test case 1 is a result of sheer luck, but in this case where we are working with the genetic algorithm, the notion of luck is more complicated. Even though the randomness element in the genetic algorithm is somewhat of its strength, there is always a small chance that the randomly chosen values continues to be bad for all generation in an evaluation and result in a bad move for the GA that affect the rest of the game. A bad move can be an opportunity for the opponent to place a disc in a corner, this does not have to be the case. In the test cases the Alpha-beta looks at least 4 moves ahead. Intuitively that is still so far ahead that it can take advantage of small mistakes made by the other player.

4.4 Time Consumption

Both algorithm struggles with computational cost. In the case of the Alpha-beta algorithm we can set the search depth so that it does not use too much time, and still give a result worth talking about. For the genetic algorithm the problem is more complex. The first problem is that there are a lot more parameters affecting the result and efficiency than in the Alpha-beta algorihtm, and the second problem is that we with the genetic algorithm have lesser understanding of how changing these parameters affect the result as a whole. In the final configuration, Alpha-beta algorithm evaluates its move in a time so small that a human does not notice it. For the genetic algorithm however, the evaluation can take up to 15 second before producing an answer. Of course, this is not an unreasonable amount of time if this was a human player, but 15 second for a computer is a lot of time. Though the whole process of a genetic algorithm in all cases takes a lot of time, in this particular implementation, it is the calculation of the fitness that takes most of the time.

4.5 Over-Fitting

The Alpha-beta algorithm is deterministic, leading to the Alpha-beta player always choosing the same move from the same position. Training against such an opponent can lead to over-fitting, where the genetic player only learns how to play against a particular Alpha-beta player. If the goal of the article was to train the genetic player to beat all types of opponent, we obviously would have a problem. Since the article focuses on the comparison between these two algorithms, we can accept that the genetic algorithm does not really learn to play Othello, but only how to beat this particular Alpha-beta algorithm. A solution to this problem would be to replacs the player used as opponent in training to the algorithm it would play against in other cases.

4.6 Conclusion

With the implementation presented in this article, the test results indicate that the genetic algorithm is most likely to win over an Alpha-beta algorithm for all test cases. This applies both to cases where the GA trains against an Alpha-beta player with the same search depth as its opponent and when the genetic algorithm is restricted to train against a player with a lower search depth. The latter case demonstrates an element of robustness of the genetic algorithm. Because it is able to select does individuals that are better than others and develop its values in the further generations. Unfortunately the test cases and test runs can be seen as a little sparse, but never the less it gives us some good indications and a better understanding of the relationship between the two algorithms.

4.7 Future Work

A natural continuation of this work is extending the test cases. This does not only apply to testing a greater range of different search depths, but also experimenting with the other parameters. Because exploring the different parameters in a genetic algorithm is a science of its own (not to say an art), the focus may in first hand be related to the game of Othello. This includes, but not restricted to, the heuristic of the Alpha-beta algorithm and the representation of Othello in the genetic algorithm.

Developing the implementation so it runs faster, can be seen as an extension of the work on its own, it will also greatly benefit the experimentation of the test cases as the current time consumption quite fast become frustrating.

References

1. Faybish, I.: Applying the genetic algorithm to the game of othello. Master's thesis, VRIJE UNIVERSITEIT BRUSSEL (1999)
2. Nicolas, J.M.A., Dur, N.: A genetic algorithm to improve an othello program. In: Artificial Evolution, Springer Verlag (1995) 307–319
3. Singer, J.A.: Co-evolving a neural-net evaluation function for othello by combining genetic algorithms and reinforcement learning. In: Proceedings of the International Conference on Computational Science-Part II. ICCS '01, London, UK, UK, Springer-Verlag (2001) 377–389
4. Moriarty, D., Miikkulainen, R.: Evolving complex othello strategies using marker-based genetic encoding of neural networks. Technical report (1993)
5. Moriarty, D.E., Miikkulainen, R.: Discovering complex othello strategies through evolutionary neural networks. Connection Science **7** (1995) 195–209

6. Emmanuel Lazard, F.T.: Strategy guide. `http://radagast.se/othello/Help/strategy.html` (March 1993) [Online; accessed 14-November-2012].
7. Junghanns, A.: Are there practical alternatives to alpha beta? ICCA Journal (1998)
8. Stuart Russel, P.N.: Problem-solving. In: Artificial Intelligence - A modern approach. Third edn. Pearson Education (2010)
9. Whitley, D.: A genetic algorithm tutorial. Statistics and Computing **4** (1993) 65–85
10. Cunningham, T.: Using the genetic algorithm to evolve a winning strategy for othello. In Koza, J.R., ed.: Genetic Algorithms and Genetic Programming at Stanford 2003. Stanford Bookstore, Stanford, California, 94305-3079 USA (4 December 2003) 31–37

Tree Identification and Trunk Diameter Estimation with a 2D Laser Scanner

Peter Hohnloser

Department of Computing Science
Umeå University, Sweden
peterh@cs.umu.se

Abstract. This paper presents an algorithm for identifying trees in a 2D laser scan and six different tree trunk diameter estimation methods. The tree identification algorithm turned out to be very sensitive and hence not always as reliable as one could wish. Of the tree trunk estimation methods three were developed during this work. All methods were tested and compared in an experimental way to find out which is most appropriate to use in forestry. The experiment was conducted on three positions where the tree trunks visibility differed. The result shows that one of the existing methods is not as reliable as the other ones and the other two existing ones give a similar result. Noticeable is that the existing methods overestimates the tree trunk diameter and the newly developed methods mostly underestimates it, but as the experiment shows gives the most accurate result in some positions.

1 Introduction

Today forestry companies normally use forestry machines, for example harvesters that cut, delimb, and buck trees and quite often they are combined with another forestry machine; a forwarder, that moves the logs that the harvester created to a roadside landing.

The forestry machines collect large amount of data from their current work, which is used e.g. in forest companies' own monitoring and continuous improvement of planning procedures.

Continuous reports of production data improves the control of timber supply, and GPS-technology simplifies the work by supplying information about where the forestry machine is relative to, for example, property boundaries and in demanding environments. The geographical information is also used in later parts. The forwarder receives information from the harvester about where the woodpiles are located.

A trend in the development for future forestry machines are autonomous functions for crane control and navigation. A central problem is the localization of the forestry machine and the crane tip. Methods based on GPS has limitations and do often need to be complemented with other methods to work in forest environments. This article focuses on the complementing use of a 2D Laser Scanner and the importance of estimating a tree's diameter. A Laser Scanner measures

S. Bensch, F. Drewes (Eds.): Umeå's 16th Student Conference in Computing Science USCCS 2013, pp. 27–38, January 2013.

the distance to an objects in the surrounding, in a fan-shaped area by sending laser beams measuring the time of flight. Using such methods will for example contribute to three important aspects:

1.1 Calibration of Airborne Laser Scanning of Trees

During the last years the development of new sensor technology has changed the possibilities to improve the gathering of geographic information for larger areas with high precision and high resolution. E.g. the Swedish Land Survey (Lantmäteriet) laser scans large parts of Sweden from an airplane during a four year project. Several forestry companies have also gathered data from Airborne Laser Scanning (ALS) for their own forestry holdings. The method simplifies collecting information for the owner of a forest because manual analysis of images from aerial photography can now be replaced by a computerized and automated analysis of ALS data, providing improved information of the forest area. The laser scanning from an airplane only measures the size and the shape of the crown of a tree, something that is currently calibrated with manual measurements, for example diameter and volume for trees in a test area whose position is measured by GPS. The manual measurements are quite time consuming and improving the measurement would greatly improve the calibration of ALS for forest application.

Tree position and tree trunk diameter can be determined with the help of data from a ground based laser scanner. Automatic calibration can be made by comparing this information with the ALS information.

1.2 Localization of the Forestry Machine

In the forest environment a GPS gives quite poor accuracy, normally around ± 10 m. One way of improving the forestry machine position is by using a laser scanner that detects trees and matches them with an ALS to get the exact position. This could be one step in the method for determining the tree position when harvesting described in Section 1.3. Another use of localization of the forestry machine could be for navigation of (semi-) autonomous forestry machines where the exact position of the machine is needed [1].

1.3 Determining the Tree Position When Harvesting

When locating the forestry machine using a laser scanner, as described in Section 1.2, it is known that a tree is located within a 10 meter radius , which also is the range of the crane. If the crane's exact position is known when a tree is harvested it is possible to combine the collected data on the trees type, height, diameter, quality, etc with the known position. This combination means a number of opportunities and benefits:

– Data collected when harvesting trees can be used to calibrate data from the airborne laser scanning (ALS) and thus obtain continuous calibration data (an improvement over manual calibration).

- It is possible to get a total traceability of the wood production. For example, a furniture manufacturer can show a costumer the exact position for each tree that is used for the product.
- It is also possible to improve the details when selecting which tree to harvest. If the harvester comes close to a protected area the driver will be informed.
- Automation and semi-automation of the harvester work is possible. For example the harvester automatically position itself or the crane grabs the tree the driver pointed out on a tree map automatically [2].

All these tasks can be approached using a laser scanner that can identify trees and determine the positions and diameters of them. This paper describes an algorithm for identifying trees [2, 3] and evaluates six different trunk diameter estimations; three existing [3–5] and three methods that we have developed [6]. To evaluate these six different methods and the tree identification algorithms, a experiments was conducted in a forest environment.

2 Experiment

For the experiment a SICK LMS 221 laser scanner was used. The angular resolution is 0.25° and the field of view is 100°. Each scan consists of 400 beams. The laser scanner has a measurement range up to 80 m with a measurement accuracy of $\pm 3,5$ cm. A problem is that only a very few laser beams hit each tree (at 10 m range the distance between two laser beams is ca. 4.5 cm, which means that a tree with 20 cm diameter is hit by only four beams). This seriously reduces the accuracy for both tree detection and diameter estimation. The laser scanner could also hit leaves, needles or branches, which makes it even more difficult to identify trees. Each laser beam has a width of around 0.8° (ca. 14 cm at 10 m range).

For the experiment the laser scanner and a GPS were mounted on top of a wood harvester cabin (see Figure 1) ca 4m above ground. The roll and pitch angel of the laser scanner were approximately level to the cabin but the yaw angel was unknown since we could not measure this.

The laser scanner measurements took place on three different positions in the same forest, along a slope (around 8°, with the top towards northeast). Position A had many trees leaves, needles or branches and the measuring were made from the bottom of a small hill up to the hills top. The Second Position B1 was higher up then Position A and the measurements where measured up to the hill top again. This position had the most visible tree trunks without branches, etc. Position B2 is the last position and were taken at the same place as Position B1. The only difference between the two positions are that the harvests cabin was rotated so that the laser scanner point down towards Position A. The last position had trees with and without leaves, needles and branches. The three positions and their respective viewing angle can be seen in Figure 2.

The diameter of, and distance between, the trees were measured. As was the distance between the laser scanner and the trees. Each tree was to have a minimum of two adjacent trees with known distance. This information was

Fig. 1. The Laser Scanner (yellow arrow) and GPS (red arrow) mounted on top of the cabin of a wood harvester. Image by Ola Lindroos used with kind permission.

collected manually from the small forest with 53 trees and a manual draft tree map was created. From the beginning, the trees that the tree identification algorithm found were matched with the drafted tree map. This turned out difficult and a rematch had to be done when tree identification algorithm variables were changed. The solution, when the forest was harvested, was then to create a tree map from GPS-coordinates (see Figure 2).

3 Tree Identification

For identifying trees, we used an algorithm from Jutila et al. (2007) [3] with slight modifications. To identify trees in a laser scan, the first thing to do is to cluster laser points. A cluster is a collection of points with something in common. The tree clustering algorithm has two different parts. The first part creates a cluster of laser points close to each other, i.e. $\|r_i - r_{i-1}\| < \Delta R_{max}$ [3, 2], where r_i is the range of the i:th measurement and ΔR_{max} is the threshold for the maximum allowed distance between two points. The second part of the algorithm validates each cluster with the following five rules:

1. The number points in the cluster has to be greater than two.
2. The curvature has to be greater than or equal to zero ($curv \geq 0$, see equation 2).
3. The diameter, calculated by "Circle Fit" algorithm (see Section 4.4), from the cluster has to be between 15 and 80 cm.

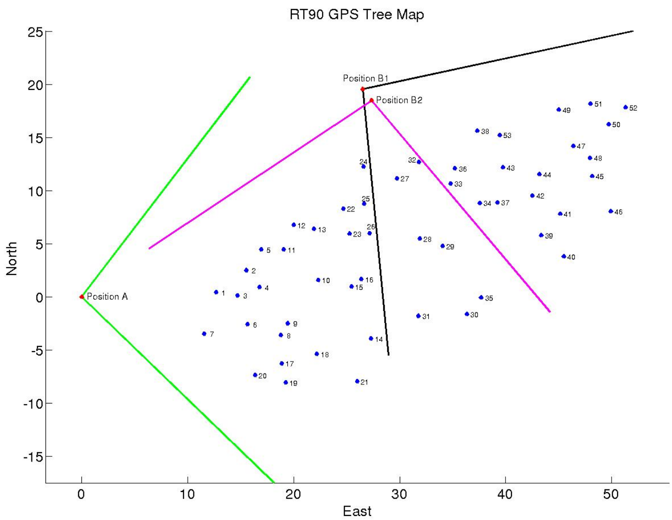

Fig. 2. RT90 GPS tree map solution where the blue dots are 53 trees and the three laser scanner positions are red dots. At each position, two lines shows the viewing angel of the laser scanner.

4. The “Circle Fit” max error has to be less than 0.15.
5. The range (in this case the height) of the cluster has to be greater than 0.08.

The curvature for a point is calculated with this equation [3] :

$$c_i = r_{i-1} - 2r_i + r_{i+1} \tag{1}$$

The curvature for the whole cluster is calculated using the number of points n of the cluster as follows :

$$curv = \left(\sum_{i=1}^{n-1} c_i\right) / \left((n-1) * r_{middle}^2\right) \tag{2}$$

Where r_{middle} is the middle point of the cluster.

To find the trees with branches and leaves, different techniques were used, for example standard deviation and shaking the laser scanner to find nearby items. Standard Deviation was used after several scans to see if there was a difference between the scans. When shaking the laser beam changes target and previous hidden items (the tree trunk) can be found. In addition to using just one scan, using the median and mean respectively of several scans was also tested. As described above, the tree identification algorithm depends, among other things, on tree diameter estimation methods. The better the diameter estimation is, the better the size of the tree cluster can be determined. Section 4 describes different methods for estimating diameter.

4 Trunk Diameter Estimation Methods

This Section describes six different methods for the trunk diameter estimation. "Two triangle diameter estimation" (TDE) [3], "Diameter estimation with resolution of the laser scanner and the two outer points" (DER) [4], and "Circle Fit" (CF) [5] are existing tree diameter estimation methods. "Two triangle trunk estimation with two outer points adjusted" (TDEA), "Circle Fit with two outer points adjusted" (CFTA) and "Circle Fit with all points adjusted" (CFAA) are methods modified by us.

4.1 Two Triangle Diameter Estimation

This approach for calculating the tree trunk diameter uses the outer points and the shortest range (r_{middle}) of the tree cluster. The angle θ is calculated by the two outer points of the tree cluster [3], see Figure 3 The diameter (d) is calculated as follows:

$$d = 2r_{middle} * sin(\frac{\theta}{2})/(1 - sin(\frac{\theta}{2})) \tag{3}$$

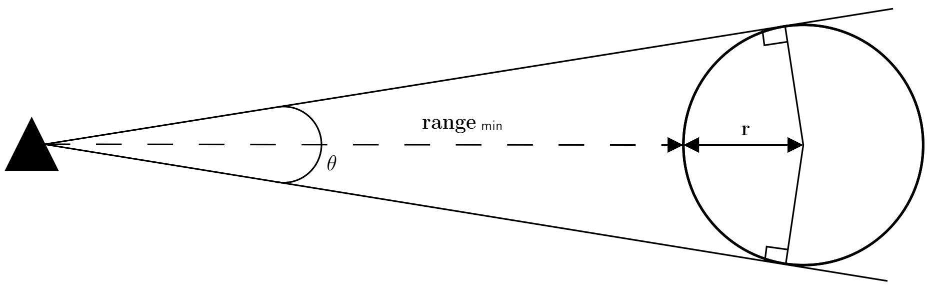

Fig. 3. Tree trunk diameter calculation using two right-angled triangles and the shortest range point ($range_{min}$) of the tree cluster [3].

4.2 Diameter Estimation with Resolution of the Laser Scanner and the Two Outer Points

This method uses the first (r_1) and the last (r_n) cluster points, the number of cluster points, and the resolution of the laser scanner to estimate the trunk diameter d [4]. The resolution is the distance between two laser beams from the same laser scanner.

$$d = (n - 1) * \triangle\beta * (r_1 + r_n)/2 \tag{4}$$

where:
n is the number of cluster points
$\Delta\beta$ is the resolution of the laser scanner

4.3 Two Triangle Trunk Estimation With Two Outer Points Adjusted

In this approach the trunk diameter is calculated with the TDE described in Section 4.1. The only difference is that the two outer points of the tree cluster is adjusted with ca. 0.27°. The adjustments were made to simplify the estimation of the tree diameter.

4.4 Circle Fit

Another approach for estimating a trunk diameter is a Circle Fit algorithm in a x,y plane. To use the this, the first thing to do is to transform from Polar to Cartesian coordinate system. Circle fit tries to find the circle that best fits a given set of measured (x,y) pairs, but with some added noise, see Figure 4. The algorithm outcome is a circle with known radius (R) and center point (xc,yc) (i.e. finding xc, yc, R such that $(x - xc)^2 + (y - yc)^2 = R^2$).

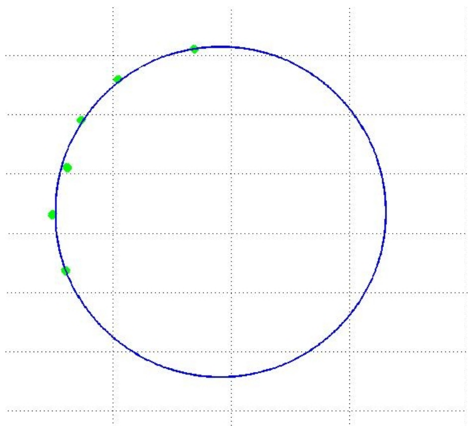

Fig. 4. The green points are a tree cluster and the blue circle is calculated with the CF algorithm.

4.5 Circle Fit With Two Outer Points Adjusted

This diameter estimation method uses CF for calculating the trunk diameter where the two outermost points of the tree cluster are adjusted with ca. 0.27°. The adjustments were made to simplify the estimation of the tree diameter. Figure 5 shows a tree cluster with green dots and the adjusted points with red crosses. The blue circle is created with the adjusted points with Circle Fit (see Section 4.4).

4.6 Circle Fit With All Points Adjusted

The last diameter estimation uses CF for calculating trunk diameter after all tree cluster points have been adjusted with ca. 0.27°. The adjustments were made

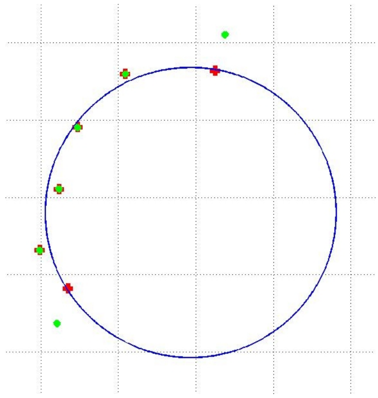

Fig. 5. The green points is a tree cluster, the red points are the tree cluster where the two outer points are adjusted and the big blue circle is calculated with CF algorithm from the cluster where the two outer points are adjusted.

to simplify the estimation of the tree diameter. Figure 6 shows a tree cluster represented by green dots and the adjusted points represented by black dots. The blue circle is a circle created with the adjusted points with (see Section 4.4).

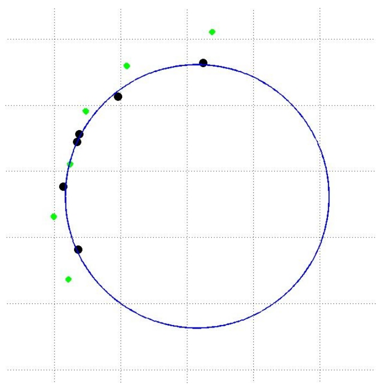

Fig. 6. The green points is a tree cluster, the black dots are the tree cluster where all points are adjusted and the big blue circle is calculated with CF algorithm from the cluster where all points are adjusted.

5 Results

Table 1 presents the mean absolute error value in percent for all six tree trunk estimation methods for all three measured positions. If the error value is positive the trunk estimation method overestimates the diameter otherwise it underestimates it. The ideal error value would be zero.

	Position A	Position B1	Position B2
TDE	150.27	28.77	15.38
DER	150.65	28.03	14.94
TDEA	100.74	18.85	28.82
CF	185.15	59.89	33.92
CFTA	127.57	33.01	20.93
CFAA	119.89	30.39	27.94

Table 1. The mean absolute error value in percent for all six estimation methods for all three measured positions.

5.1 Position A

At position A (see Figure 7), the tree identification algorithm identified five trees but one was a false positive (i.e the cluster did not match a real tree). At this position the clusters varied, the better the cluster was (i.e if no branches were disturbing) the better the methods work. Tree number 2 is problematic for all diameter estimation methods. This is because of that the tree cluster contains not only the tree trunk but also parts of branches and leaves.

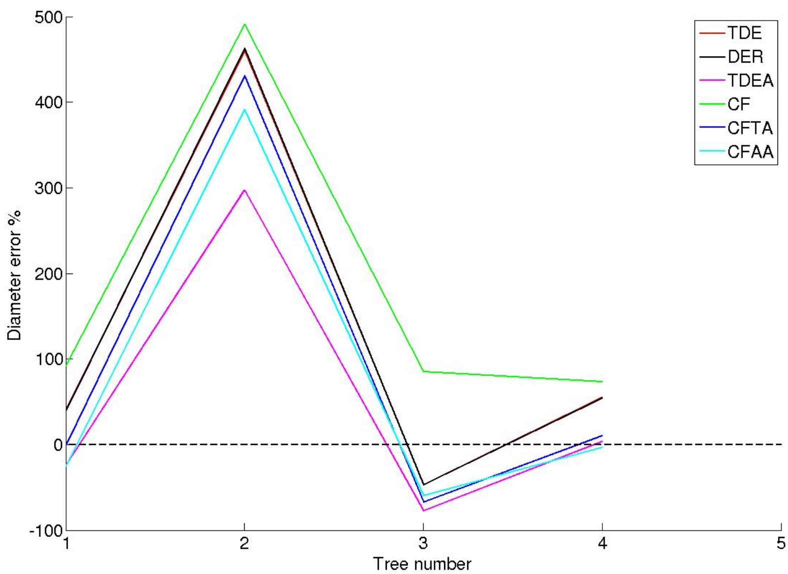

Fig. 7. Diameter error in percent from Position A for all six diameter estimation methods. Tree number 5 is a false positive.

5.2 Position B1

At position B1 (see Figure 8) the tree identification algorithm identified eight trees, all of them matching real trees. In general it seems as if the CF, TDE, and

DER overestimated the tree trunk diameter. The methods where the spot diameter is compensated, for underestimated the tree trunk diameter. Tree number 2 and 7 are problematic for all diameter estimation methods. The reason for that is that the tree cluster consists not only of the tree trunk, but also part of a branch. For this position the most stable tree trunk estimation method is the TDEA.

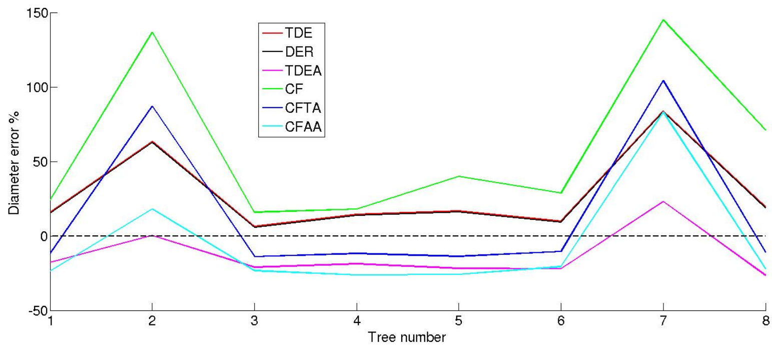

Fig. 8. Diameter Error results from Position B1 for all six diameter estimation methods.

5.3 Position B2

The tree identification algorithm identified 4 trees, all of them matching real trees, at position B2 (see Figure 9). In general, the tree trunk estimation method CF overestimates the tree trunk diameters at this position. The estimation method CFAA underestimates the tree trunk diameters.

6 Summary and Conclusions

The tree identification algorithm works reasonable well even in a forest with many branches. However, the algorithm is very sensitive and the outcome could differ depending on the variables used for identifying the trees. To be able to identify trees in a 2D laser scan even if the tree has branches, needles, and leaves, the variables has to be set differently depending on the conditions. It could result in that some of the identified trees being false positives and some false negatives (i.e. some trees are missing and some that are detected are not trees).

After testing standard deviation and shaking the laser no real change in the way a tree with branches was found could be seen. Shaking the scanner was not a good option for the type of laser used in this experiment, since it has so wide laser beams that a minor shake does not change the scanned area enough. The best way to detect trees with branches/leaves was to use the median value of several scans as input to the diameter calculation algorithms. Using that method did not affect the trees without branches/leaves either.

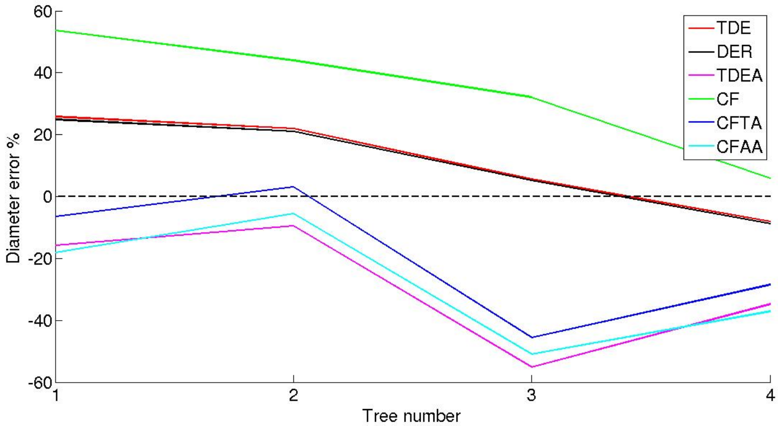

Fig. 9. Diameter Error results from Position B2 for all six diameter estimation methods.

The result from the different estimation methods compared in this article are all different and it is not possible to argue that one of the methods for diameter estimation is the best at all times. TDEA was the best method for position A where the tree trunks were hidden by branches, and for position B1 where the tree trunks were easy to find. DER was the best method for position B2 where some of the trunks were hidden and some not. Regarding the other methods CF, CFTA and CFAA are not recommended for this kind of work, because they can not handle the noise in the tree cluser. The two methods TDE and DER give rather similar results.

7 Future Work

In this area much research is yet to be done. Using a Neural Network to get a better trunk diameter estimation could be one way forward. However, for that much more data must be collected. Another improvement could be to add a camera to improve both the algorithm for detection of trees (the clustering algorithms) and the algorithms for estimation of tree diameter. Better results would probably also be achieved with another type of laser, with thinner and distance independent beams.

8 Acknowledgements

Thanks to Sveaskog who assisted with a forest machine and an operator for our tests. Thanks also to Ola Lindroos at SLU, for his substantial help with the field experiments.

References

1. Ringdahl, O.: Automation in Forestry - Development of Unmanned Forwarders. Phd thesis, Department of Computing Science, Umeå University (2011) ISBN 978-91-7459-213-9.
2. Zheng, Y., Liu, J., Wang, D., Yang, R.: Laser scanning measurements on trees for logging harvesting operations. Sensors **12**(7) (2012) 9273–9285
3. Jutila, J., Kannas, K., Visala, A.: Tree measurement in forest by 2d laser scanning. In: Computational Intelligence in Robotics and Automation, 2007. CIRA 2007. International Symposium on. (june 2007) 491 –496
4. Selkäinaho, J.: Adaptive autonomous navigation of mobile robots in unknown environments. PhD thesis, Helsinki University of Technology, Espoo, Finland (2002)
5. Umbach, D., Jones, K.N.: A few methods for fitting circles to data. IEEE Transactions on Instrumentation and Measurement **52**(6) (December 2003) 1881–1885
6. Thomas Hellström, Peter Hohnloser, O.R.: Tree diameter estimation using laser scanner. Technical report, Department of Computing Science, Umeå University (2012)

Automatic Determination of Major Vocal Types Using Support Vector Machines

Saeed Partonia

Department of Computing Science
Umeå University, Sweden
mcs10spa@cs.umu.se

Abstract. This paper discusses the classification of basic vocal types for northern Swedish voices. The classification is carried out by use of support vector machines. The paper aims to identify the important features of voice signal for classification. The paper also presents the results of applying the selected features in classification performance. Finally, those features that outperform others in classification are introduced.

1 Introduction

Voice type plays a major role in music composition. An expert, e.g., a composer, can usually assign a human voice to its corresponding vocal category. Basically, there are four types of voices; soprano, alto, tenor and bass [1]. The motivation for this work is to automatically detect the voice type without help of human. For this task, we use Support Vector Machines (SVM) [2, 3] to perform classification. While there are other classification methods such as Artificial Neural Networks (ANN) which also might be used, SVM has several advantages over them. First, unlike ANN, which is subject to fall in the local optima, SVM always converges to the global optimum [4]. Furthermore, SVM has a straightforward model representation unlike ANN, in which selecting a proper model e.g. number of hidden layers, is challenging task. Before applying any classifier, the voices must be feature extracted. An audio file usually has several characteristics including low level features (such as pitch, zero-crossing rate etc), high-level features (like MFCCs acquired from low-level features) and psychoacoustic features (roughness, sharpness, etc) [5]. The aim of this work is to identify suitable features for vocal type classification for northern Swedish voices. A version of this work is implemented by the author as a project course at Umeå University. In this paper we will examine additional features in order to improve the accuracy of the system. The structure of manuscript is as follows. In the next section the idea behind SVM is briefly described. Then, we continue with experiments and model building for the SVM. Finally some conclusion are drawn based on the experiments and then best features are selected.

S. Bensch, F. Drewes (Eds.): Umeå's 16th Student Conference in Computing Science USCCS 2013, pp. 39–48, January 2013.

2 Support Vector Machine

Support Vector Machines (SVM), introduced by Cortes and Vapnik [6], are class of supervised learning methods used for classification. Basically, SVM is a binary classifier, that maps an input variable x to a binary output $y \in \{-1, 1\}$. In this section we introduce the basic concepts of SVM and then extend the subject to cover multi class classification and non-linear classification.

2.1 Basics of SVM

Let x_i denote an input consisting of D attributes (i.e D dimensions). Suppose x_i belongs to one of two classes, either $y_i = +1$ or $y_i = -1$. Hence, collection of data has the form

$$\{(x_1, y_1), (x_2, y_2), ...(x_n, y_n)\} \tag{1}$$

where n is the total number of samples, $x_i \in \mathbb{R}^D$ is the input vector and $y_i \in \{-1, 1\}$ is the class of corresponding x_i. for the sake of simplicity, we assume that the data is linearly separable. In other words, we one can distinguish between two classes by drawing a line between two sets of points. This line is known as hyperplane, considering dimensionality higher than two. The Hyperplane is described as

$$w.x + b = 0 \tag{2}$$

where w is normal to the hyperplane and b is a scalar showing the offset from origin. Notably, absence of b forces hyperplane to pass through the origin which restricts the solution (Figure 1).

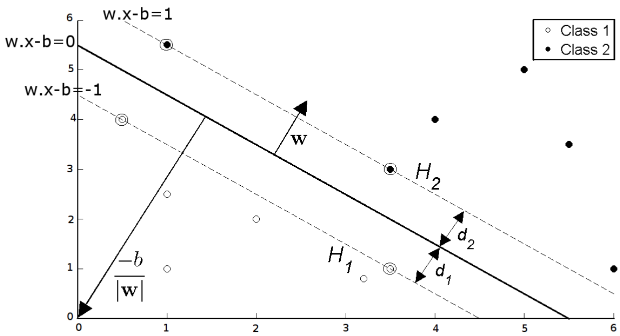

Fig. 1. Hyperplane separating two distinct classes.

Since we assumed that our data is linearly separable, we can imagine two hyperplanes separating two classes in a way that, no points exist between planes and their distance is maximized. Hyperplanes are described by

$$\begin{aligned} w.x + b &= 1 \\ w.x + b &= -1 \end{aligned} \tag{3}$$

By a geometric argument, it could be found that the distance between two hyperplanes is $\frac{2}{\|W\|}$. Considering the dataset, we can modify the equation 3 into:

$$\begin{cases} w.x + b \geq 1, & y_i = +1 \\ w.x + b \leq -1, & y_i = -1 \end{cases} \tag{4}$$

which also could be written in compact form,

$$y_i(w.x_i - b) \geq 0, \quad i = 1, ..., n \tag{5}$$

those samples that lie on the separating hyperplanes are called Support Vectors (SV). We define d as being the distance between support vectors and the hyperplane. The hyperplane must be equidistant from both classes. This quantity is known as *"margin"*. SVM aims to find a hyperplane called Optimal Canonical Hyperplane (OCH) such that the margin is maximized. Recall that the distance between marginal hyperplanes obtained equal to $\frac{2}{\|w\|}$, so the margin should be $\frac{1}{\|w\|}$. Hence, maximizing the margin is equal to the minimizing $\| w \|$. In order to simplify the calculation, we minimize $\frac{1}{2} \| w \|^2$ which is cheaper computationally. This turn out that we need to find

$$min(\frac{1}{2} \| w \|^2) \tag{6}$$

subject to

$$y_i(w.x_i - b) - 1 \geq 0, \quad i = 1, ..., n \tag{7}$$

which is a quadratic programming optimization problem. Using the Lagrange Multipliers [7], we can convert the equation 6 into an unconstrained minimization problem by constructing the cost function,

$$J(w, b, \alpha) = \frac{1}{2} \| w \|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(x_i.w + b) - 1 \right] \tag{8}$$

where $\alpha_i \geq 0$ is called the Lagrange multiplier. Now, we are interested in minimizing the cost function $J(.)$ with respect to w, b and maximizing with respect to $\alpha_i \geq 0$.

$$\max_{\alpha} \; (\min_{w,b} \; J(w, b, \alpha)) \tag{9}$$

Minimum amount of $J(.)$ with respect to w, b is given by,

$$\begin{aligned} \frac{\partial J}{\partial b} &= 0 \Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0 \\ \frac{\partial J}{\partial w} &= 0 \Rightarrow \sum_{i=1}^{n} \alpha_i y_i . x_i = 0 \end{aligned} \tag{10}$$

putting equations 8, 9 and 10 together, we obtain objective function $W(\alpha)$,

$$\max_{\alpha} \; W(\alpha) = \max_{\alpha} - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_{k=1}^{n} \alpha_k \tag{11}$$

This turns out that we need to minimize,

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j - \sum_{k=1}^{n}\alpha_k \tag{12}$$

with respect to constraints,

$$\alpha_i \geq 0, \quad i = 1, ..., n \sum_{i=1}^{n}\alpha_i y_j = 0 \tag{13}$$

By solving the equation 12 and considering the constraints in 13, Lagrange multipliers are obtained. So, the optimal hyperplane is defined as

$$w = \sum_{i=1}^{n}\alpha_i y_i x_i b = -\frac{1}{2}(w(x_r + x_s)) \tag{14}$$

where x_r and x_s are any support vectors from either of the classes that satisfy,

$$\alpha_r, \alpha_s > 0, \quad y_r.y_s = -1 \tag{15}$$

Now, we can classify each new input $x^{'}$ with

$$y = sgn(w.x^{'} + b) \tag{16}$$

the sign of y determines the class which input $x^{'}$ belongs to.

2.2 Soft Margin Classifiers

Consider the case that the data is not fully linearly separable. In other words, there is no such a hyperplane that separates two classes in the data. In that case, we may allow some samples to be misclassified in order to be able to draw a hyperplane (Figure 2).

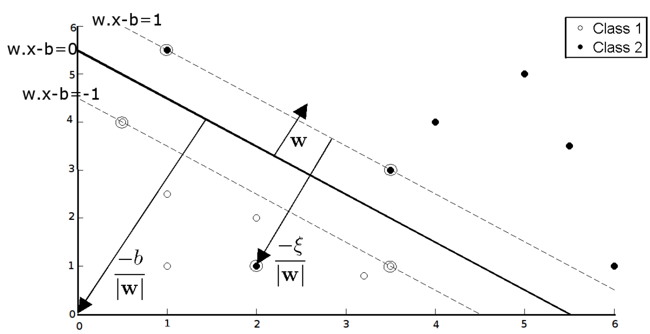

Fig. 2. Hyperplane through two nonlinearly separable classes.

Cortes and Vapnik [6] suggested *Soft Margin* method which allows some misclassified points by relaxing constraint in the equation 4. This is done by introducing slack variable $\xi_i, i = 1, ..., n$ which measures the degree of misclassification for x_i,

$$\begin{aligned} x_i.w + b &\geq +1 - \xi_i \quad , y_i = +1 \\ x_i.w + b &\leq -1 + \xi_i \quad , y_i = -1 \end{aligned} \tag{17}$$

where $\xi_i, i = 1, ..., n$. Now the objective is, dealing with a trade off between making the margin as large as possible, while keeping the misclassification penalty small which is proportional to the distance from the margin,

$$min \frac{1}{2} \parallel w \parallel^2 + C \sum_{i=1}^{n} \xi_i \quad s.t \quad y_i(x_i.w + b) - 1 + \xi_i \geq 0, i = 1, ..., n \tag{18}$$

where parameter C controls the trade off between slack variable penalty and the size of margin.

Similar to the equation 11, solving equation 18 with Lagrange multipliers gives the objective function,

$$\max_{\alpha} \left[\sum_{j=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \right] \tag{19}$$

with respect to

$$0 \geq \alpha_i \geq C \quad , i = 1, ..., n \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{20}$$

Note that α_i is now limited to C.

2.3 Nonlinear SVM

it is sometimes the case that the data is not linearly separable and SVM fails to perform classification correctly (Figure 3). Bernhard et al. [8] suggested using a *"kernel trick"* to create a nonlinear classifier.

The idea behind the kernel trick is to transform input vector x into a higher dimensional feature space such that the SVM is able to construct an optimal hyperplane in the new space (see Figure 3). Resulting algorithm is identical to the linear SVM except that instead of dot product of the input vectors, $k(x_i, x_j) = x_i^T x_j$ which is known as linear kernel, a nonlinear kernel is replaced.

There are four popular kernel functions according to [9]:

- Linear: $k(x_i, x_j) = x_i^T x_j$
- Polynomial: $k(x_i, x_j) = (\gamma x_i^T x_j + r)^d \quad , \gamma > 0$
- Radial Basis Function (RBF): $k(x_i, x_j) = \exp(-\gamma \parallel x_i - x_j \parallel^2) \quad , \gamma > 0$
- Sigmoid: $k(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)$

where γ, r, d are kernel parameters. The choice of kernel function is largely depends on the training data and and its distribution.

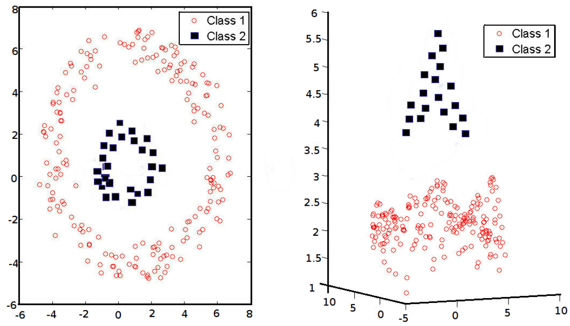

Fig. 3. Nonlinearly separable data (left), is transformed to be linearly separable (right).

2.4 Multi Class Classification

Support Vector Machines were originally designed to solve the binary classification problems. However, many real world problems need to be formulated as multi class problems. Several techniques proposed to extend the SVM concept to the multi class classification [10, 3]. "One-versus-one", "one-versus-all" and "directed acyclic graph SVM (DAGSVM)" are amongst the most popular techniques. Choosing the proper technique among those methods come with a trade off between the accuracy and computation cost. While one-versus-all is efficient computationally, one-versus-one and DAGSVM achieve more accurate results [3]. Following section, briefly describes these methods.

One-Versus-All In this method, M binary classifiers are trained where M is the number of classes. The ith SVM, M_i, is trained with taking the elements of the ith class as positive labels and elements of all other $M-1$ class as negative. Thus, $M-1$ classifiers are trained. Each new input x, is assigned to the class, whose output is maximized among others.

One-Versus-One In this method one classifier is constructed for each pair of classes. So, for a M class problem, $M(M-1)/2$ classifiers are trained. In other words, a classifier C_{ij} is trained, such that, it takes the elements of class i as positive, and the elements of class j as negative. The class of each new input x, is determined as follows,

1. for all classifiers, if classifier C_{ij} assigns x to class i, vote for class i is added by one. Otherwise, increase j's vote by one.
2. assign x to the class with highest vote. If several classes exist, choose one by random

Directed Acyclic Graph SVM (DAGSVM) Similar to the one-versus-one method, for a M class problem, $M(M-1)/2$ classifiers are trained. To choose the

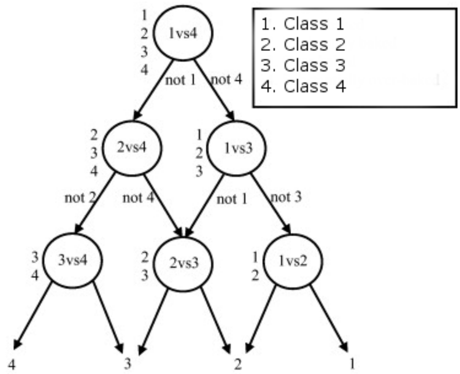

Fig. 4. Determining the winner in DAGSVM method.

winner class, DAGSVM uses a routed binary directed acyclic graph. The graph consists of $M(M-1)/2$ internal nodes, representing the various classifiers, and M leaves, denoting the number of classes. For each new input x, it starts with the root node, considering the output of the classifier placed in the node, it moves either to the left or right until reaching to a leaf node. The class of leaf node is the class of input x (see Figure 4). It has been shown [3], that DAGSVM usually performs faster than one-versus-one.

3 Experiments

The classification experiments are conducted on a database of recorded Swedish voices. This database includes three major types in which 47 of them are altos, 95 are bass and 101 are sopranos. The voices are recorded with sampling rate of 48 KHz. These voices are labeled manually by a music expert. In order to use SVM for classification, these files must be feature extracted.

3.1 Feature Extraction

In order to be able to classify the voices using SVM, first, they must be feature extracted. An audio file has several characteristics like low level, high level and psychoacoustic features [5, 1, 11]. Low level features refer to the characteristics of the voice that could be acquired directly from signal. Examples of such features are voice pitch, zero crossing rate and so on. High level features are those which could be calculated from the relationships between low level features but not directly through the signal itself. Mel Frequency Cepstral Coefficients (MFCCs) are instances of such features. Finally, the third group is psychoacoustic features like roughness or sharpness of sound. Choosing the suitable features in different voice applications are totally objective and varies from case to case. For instance, in music genre classification, both low level and high level features have significant importance [12, 5, 11]. On the other hand, for gender identification, high level features showed better performance [13, 14]. However, it does not mean that, the low level features are not used at all. In this paper, we used both low level and high level features separately and also in combination with together. Table 1 illustrarates the features we used, with a brief description.

Feature #	Description
1	min pitch
2	max pitch
3	mean pitch
4	std pitch
5	zero-crossing rate
6	brightness
7,...,19	MFCCs

Table 1. Features used and their descriptions.

3.2 Model Selection

In implementing SVM, some issues must be considered which have great effect on the SVM's performance. First, we need to scale the parameters to the scale of $[-1, 1]$. Scaling has two advantage [9]. First, it confronts the attributes with greater range to dominate those with smaller range. furthermore, it makes the calculation easier when the values have smaller range. Other issue which must be considered is choosing the penalty parameter C and kernel parameters. Here we use RBF kernel for implementation which needs selection of value for the γ. One reason for using the RBF kernel is that, according to [9], number of the features are not very large compared to the training samples or vice versa. A technique called Cross-validation and Grid-search is introduced to deal with the selection of best values for (C, γ) pair. The strategy is as follows. We first divide the data into sets of 60%, 20%, 20% respectively as training, cross-validation and the test sets. Then, we try different values for C and γ (grid search) and validate the model on the cross-validation set. The combination of (C, γ) which produces the best accuracy for the SVM is selected. The final SVM is then trained with achieved (C, γ) pair and will be tested on the test set. In our experiments we used values in the range $[0.01, 0.03, 0.1, ..., 10, 30]$ for both C and γ. For multi class classification, one-versus-one method is used.

3.3 Results and Discussion

The parameters C and γ are automatically selected from 64 different combinations of $\gamma = [0.01, 0.03, 0.1, ..., 10, 30]$ and $C = [0.01, 0.03, 0.1, ..., 10, 30]$ (Table 2).

Furthermore, a total number of 19 high level and low level features are examined (Table 1). The low level features, include six types; min, max, mean and standard deviation of pitch, zero crossing rate and brightness. High level features include 13 point MFCCs. With the low level features only, achieved accuracy is

Exp. #	(C, γ)		
	alt vs. bass	alt vs. sop	bass vs. sop
1	(1.0, 0.3)	(0.01, 0.01)	(10, 0.3)
2	(0.01, 0.3)	(30, 3.0)	(0.01, 0.03)
3	(0.3, 3.0)	(0.01, 3.0)	(0.01, 0.03)
4	(0.01, 0.03)	(10, 1.0)	(0.01, 0.03)
5	(0.01, 0.03)	(10, 1.0)	(0.01, 0.03)
6	(30, 1.0)	(3.0, 0.3)	(10, 1.0)
7	(1.0, 0.3)	(10, 0.3)	(1.0, 0.3)
8	(0.01, 0.03)	(10, 1.0)	(0.01, 0.03)
9	(0.01, 0.03)	(30, 3.0)	(0.01, 0.03)
10	(0.01, 0.03)	(10, 03)	(0.01, 0.03)
11	(0.1, 0.1)	(0.01, 0.3)	(0.01, 0.03)
12	(0.01, 0.3)	(0.01, 0.3)	(0.01, 0.03)

Table 2. Choice of C and γ for different iterations.

about 65%. However, high level features achieved 90% accuracy. The main weakness of using the high level features is that the SVM fails to distinguish between altos and sopranos. For this reason, we add some low level features to compensate it. Table 3 illustrates the outcome of combining multiple features both from the high and low level features. From Table 3 it is evident that the combination of features 5 and 6 from low level features with the high level features from 7 to 11 outperforms the other combinations with 95% accuracy.

Exp. #	Features	Accuracy (%)			
		overal	alt vs. bass	alt vs.sop	bass vs. sop
1	low level	65	53	78	69
2	high level	90	93	87	92
3	1,..,4,7,..,19	89	92	86	100
4	5,6,..,19	91	96	86	100
5	3,4,..,19	91	100	85	100
6	1,2,..,13	87	100	79	97
7	1,2,..,11	87	80	92	97
8	1,..,6,11,..,19	87	96	81	100
9	5,6,11,..,19	83	93	74	100
10	5,..,11	95	100	94	97
11	3,..,6,11,..,19	81	93	75	100
12	1,..,6,13,..,19	85	96	76	100

Table 3. Accuracy of model with selected features.

4 Conclusion

In this paper we identified the features for vocal type classification. Results showed that the high level features (MFCCs) outperform low level ones, when used separately. However, it still suffers from some misclassification between altos and sopranos. For this reason, a combination of the low level and high level features are tested. Results showed that, the combination of zero crossing rate and brightness from low level features, along with 5 first MFCCs from high level features achieved the best results.

References

1. Cotton, S.: Voice classification and Fach— Recent, historical and conflicting systems of voice categorization. PhD thesis, The University of North Carolina at Greensboro (2007)
2. Fletcher, T.: Support vector machines explained. Technical report, UCL (2009)
3. Duan, K., Keerthi, S.: Which is the best multiclass svm method? An empirical study. Multiple Classifier Systems (2005) 732–760
4. Burges, C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery **2** (1998) 121–167
5. Mckinney, M., Breebaart, J.: Features for audio and music classification. In: Proceedings of the International Symposium on Music Information Retrieval. (2003) 151–158
6. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3) (1995) 273–297
7. Bertsekas, D.: Nonlinear programming. Athena Scientific (1999)
8. Boser, B., Guyon, I., Vapnik, V.: A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on Computational learning theory, ACM (1992) 144–152
9. Hsu, C., Chang, C., Lin, C., et al.: A practical guide to support vector classification (2003)
10. Hsu, C., Lin, C.: A comparison of methods for multiclass support vector machines. Neural Networks, IEEE Transactions on **13**(2) (2002) 415–425
11. Pope, S., Holm, F., Kouznetsov, R.: Feature extraction and database design for music software. In: Proceedings of the International Computer Music Conference. (2004) 596–603
12. Dhanalakshmi, P., Palanivel, S., Ramalingam, V.: Classification of audio signals using svm and rbfnn. Expert Systems with Applications **36**(3) (2009) 6069–6075
13. Harb, H., Chen, L.: Gender identification using a general audio classifier. In: Proceedings of the International Conference on Multimedia and Expo (ICME). (2003)
14. Martins, R., Trancoso, I., Abad, A., Meinedo, H.: Detection of children's voices. In: Proceedings of the I Iberian SLTech 2009. (2009)

Towards an Assessment of Gamification: Comparing Game Design Elements with Android Design Principles

Lisa Ring

Department of Computing Science
Umeå University, Sweden
id08lrg@cs.umu.se

Abstract. Gamification is the use of game design elements in non-game contexts. In this article the context is non-game applications developed for Android 4.2, Jelly Bean, and the aim is to identify possible advantages and disadvantages with this approach. Six game design elements that derive from psychological motivational needs are identified and explained. These game design elements are compared with Android's design principles to determine whether the game design elements are applicable on Android applications or not. The result shows that it is possible to develop gamified Android applications without infringing the Android design principles, even if not all game design elements clearly correlate with the Android design principles.

1 Introduction

The fact that people play games is nothing new. It can be a sport, a board game, a video game or any other kind of game. The relevant observation however, is that nowadays also other applications often are inspired by the characteristics of traditional games. For example, there are web platforms for learning languages while competing with other people[1] and internet forums where the users earn points by asking and answering computer science related questions[2]. This phenomenon has become more and more common within the area of interaction design and is widely recognized as *gamification*.

1.1 Gamification

The word gamification has occurred in documentation since 2008[3] but did not become universally used until 2010 [1]. The phenomenon is defined as "the use of game design elements in non-game contexts" [1] which also describes the main difference between an actual game and a gamified application. The relevant

[1] Duolingo, language education platform, `http://duolingo.com/`
[2] Stackoverflow, internet forum, `http://stackoverflow.com/`
[3] Paharia, R. Who coined the term "gamification"? `http://www.quora.com/Who-coined-the-term-gamification`. Accessed 2012-10-28.

S. Bensch, F. Drewes (Eds.): Umeå's 16th Student Conference in Computing Science USCCS 2013, pp. 49–60, January 2013.

difference is that the main intention of a gamified application is not a game itself but rather some kind of everyday life chore, while a game usually is played for entertainment.

1.2 Aim

Since gamification can be used on different platforms it is of interest to investigate how this phenomenon can be applied to different kinds of applications. One big platform for application development is Android, an operating system for smartphones and tablets. The aim of this article is to compare game design elements used within gamification with Android design guidelines. This comparison will be done to see if there are any advantages or disadvantages of applying gamification to applications developed for Android smartphones.

To do this investigation some game design elements need to be identified. These elements will then be compared to the Android design guidelines to see if there are any contradictions or if they are completely applicable. Another aim will also be to exemplify the usage of gamification with a real life gamified application. This will be done in order to further clarify the use of gamification in an Android application context.

1.3 Related Work

The idea of gamification originally derives from the desire to develop non-game computer systems that can motivate users to engagement in the same way as video games do [1]. In parallel with the research about gamification there has also been other terms used for similar purposes. Some examples of such terms are "productivity games" [2], "funware"[4] and "behavioral games" [3].

One application area where the phenomenon gamification has gained ground is learning and education. Several studies have tried to find an answer to how to develop educational systems for example for teaching children math [4, **?**].

1.4 Organization of Article

In Chapter 2 both guidelines for gamification and guidelines for Android development are identified and described. Chapter 3 contains a comparison between the identified guidelines from Chapter 2 and states which guidelines that are correlating and which are not. Chapter 4 gives an example of a gamified Android application and Chapter 5 gives a summarization of the result in Chapter 3 and a discussion of the conclusions that can be drawn.

---

[4] Dean Takahashi, Funware's threat to the traditional video game industry. Venturebeat, 2008. `http://venturebeat.com/2008/05/09/funwares-threat-to-the-traditional-video-game-industry/`. Accessed 2012-12-09

2 Guidelines

Guidelines are attempts to streamline a particular process. They are used to simplify the work for the developer and to increase the possibility to succeed with the process and reach a satisfying product. For this research it is the guidelines for gamification and for Android design that is relevant to identify.

2.1 Design for Gamification

Within the field of gamification the guidelines are expressed as game design elements. A game design element is a characteristic that is contributing to the game feeling and is frequently occurring in games. Individual game design elements are however neither necessary nor sufficient to create a game by their own [1].

Some of the most common game design elements that are used for gamifying applications have been identified. Since there are elements that are similar and different from each other, it is useful to structure them by dividing them into categories. One way to do this is to categorize them by human motivational sources and needs [6] (see Table 1).

Motivation can be divided into *intrinsic* or *extrinsic* motivation based on the different reasons or goals that give rise to an action. Intrinsic motivation refers to doing something because it is interesting or enjoyable, while extrinsic motivation refers to doing something because it leads to an advantageous outcome [7]. For example, reading a book because the topic is entertaining compared to reading a book because there will be an exam based on its content.

Within the field of psychology the intrinsic motivation has been perceived as more desirable than extrinsic motivation [7]. For example, studies have shown that students who are intrinsically motivated are more likely to achieve better learning results [8]. Autonomy, competence and relatedness are all examples of basic psychological needs [7].

Autonomy and the Self Autonomy is the psychological need to experience a choice of if and how to perform an action [6]. This experience of independence has been shown to increase the intrinsic motivation [9]. To take advantage of this motivation, applications should support the users desire to decide how they want to express themselves and how they want to do things [6].

Competence and Achievement The intrinsic motivation can be increased with usage of positive feedback since this tends to satisfy the psychological need of competence for performing a task [9]. Also a desire to perform achievements can be used to enhance the need of competence by offering an optimal level of challenge [6]. The feel of earning the positive feedback is required to increase the intrinsic motivation while negative feedback or failure tends to decrease the intrinsic motivation [9] and that is why it is important that the level of challenge is neither too low nor too high.

Relatedness The psychological need of belongingness is called relatedness and can be fulfilled by the perception of a social bond [6]. To satisfy the need for relatedness an application can enable the possibility to have a sense of communion or closeness with other users [10]. Relatedness is however not mandatory to perceive intrinsic motivation.

Motivational source and need	**Design element**	**Example of use**
Psychological: Autonomy and the Self	1. Support autonomy. 2. Promote creation and representation of self-identity.	Desktop skins, cell phone ring signal, online avatars, application toolbar customization.
Cognitive: Competence and Achievement	3. Design for optimal challenge. 4. Provide timely and positive feedback.	Games and learning systems with various challenge levels and immediate performance feedback.
Social and psychological: Relatedness	5. Facilitate human-human interaction. 6. Represent human social bond.	Group based games with a chat section, visualization of email exchanges over a period of time to show both tasks and social related messages.

Table 1. Game design elements divided into categories [6].

2.2 Design for Android Applications

Android is an operating systems primarily developed for mobile touchscreen devices such as smartphones and tablets. Since Android is part of the open source community anyone can develop their own applications and provide these to the rest of the world through Google play[5]. To simplify the design and development of applications consistent with Android's core values, there are a lot of guidelines and examples available at the official Android developer website[6]. These guidelines are regularly updated as Android releases new upgrades but for this research the guidelines for Android 4.2, Jelly Bean, will be used.

The most basic guidelines for designing applications for Android devices is the design principles. These are expressed in general terms and are intended to describe what an Android application should express without specifying exactly how this should be achieved. The design principles are divided into three main creative visions that are called *Enchant me*, *Simplify my life* and *Make me amaz-*

[5] Google play, official store for Android applications, `https://play.google.com/store`

[6] Android's official developer website, `http://developer.android.com/index.html`. Accessed 2012-12-09

ing[7]. These creative visions are expressed from the user's point of view, where *me* therefore refers to the user.

Enchant me The creative vision *Enchant me* mainly focus on the visible surface of the application. The usage of animations, transitions and sound effects can be used to enhance the user experience, provided they are used on the right places. The developers are also invited to offer suitable abilities for customization, to learn users' preferences over time and to manipulate objects rather than using ordinary lists, buttons and menus (see Table 2)[8].

Design principle	Description
A. Delight me in surprising ways	A beautiful surface, a carefully-placed animation, or a well-timed sound effect is a joy to experience. Subtle effects contribute to a feeling of effortlessness and a sense that a powerful force is at hand.
B. Real objects are more fun than buttons and menus	Allow people to directly touch and manipulate objects in your app. It reduces the cognitive effort needed to perform a task while making it more emotionally satisfying.
C. Let me make it mine	People love to add personal touches because it helps them feel at home and in control. Provide sensible, beautiful defaults, but also consider fun, optional customizations that don't hinder primary tasks.
D. Get to know me	Learn peoples' preferences over time. Rather than asking them to make the same choices over and over, place previous choices within easy reach.

Table 2. Android design principles from Enchant me.

Simplify my life The creative vision *Simplify my life* is about the work flow and the actual content of the application. Keeping texts short increases the probability they will be read and the user only need to see what is relevant for the moment. Pictures can be used to convey information quickly and the user should only be interrupted if it is necessary. This vision also clarifies the importance of feedback and simple navigation throughout the application to ensure that the user always knows where he or she is in the hierarchy of views and which tasks are in progress (see Table 3)[8].

Make me amazing The creative vision *Make me amazing* focus on letting users feel competent and to lower the threshold for testing new things. By highlighting the main functions, dividing complex tasks into smaller steps and offering

[7] Android's creative vision, `http://developer.android.com/design/get-started/creative-vision.html`. Accessed 2012-12-09

[8] Android's official design principles, `http://developer.android.com/design/get-started/principles.html`. Accessed 2012-12-09

Design principle	Description
E. Keep it brief	Use short phrases with simple words. People are likely to skip sentences if they're long.
F. Pictures are faster than words	Consider using pictures to explain ideas. They get people's attention and can be much more efficient than words.
G. Decide for me but let me have the final say	Take your best guess and act rather than asking first. Too many choices and decisions make people unhappy. Just in case you get it wrong, allow for 'undo'.
H. Only show what I need when I need it	People get overwhelmed when they see too much at once. Break tasks and information into small, digestible chunks. Hide options that aren't essential at the moment, and teach people as they go.
I. I should always know where I am	Give people confidence that they know their way around. Make places in your app look distinct and use transitions to show relationships among screens. Provide feedback on tasks in progress.
J. Never lose my stuff	Save what people took time to create and let them access it from anywhere. Remember settings, personal touches, and creations across phones, tablets, and computers. It makes upgrading the easiest thing in the world.
K. If it looks the same, it should act the same	Help people discern functional differences by making them visually distinct rather than subtle. Avoid modes, which are places that look similar but act differently on the same input.
L. Only interrupt me if it's important	Like a good personal assistant, shield people from unimportant minutiae. People want to stay focused, and unless it's critical and time-sensitive, an interruption can be taxing and frustrating.

Table 3. Android design principles from Simplify my life.

great defaults even novice users can feel like amazingly competent experts (see Table 4)[8].

3 Result

Some of the identified game design elements have immediate counterparts in Androids' design principles, while some of them have not. The game design elements with the most obvious correlation with the design principles are design element 3 - *design for optimal challenge* and 4 - *provide timely and positive feedback*. However, game design element 1 - *support autonomy* and 2 - *promote creation and representation of self-identity* are also correlating. All of these correlations will be described more thoroughly below.

3.1 Correlation between Autonomy and the Self and the Android Design Principles

The main focus of design principle C - *Let me make it mine* is to enable the availability to customize the application to make it more personal. For example

Design principle	Description
M. Give me tricks that work everywhere	People feel great when they figure things out for themselves. Make your app easier to learn by leveraging visual patterns and muscle memory from other Android apps. For example, the swipe gesture may be a good navigational shortcut.
N. It's not my fault	Be gentle in how you prompt people to make corrections. They want to feel smart when they use your app. If something goes wrong, give clear recovery instructions but spare them the technical details. If you can fix it behind the scenes, even better.
O. Sprinkle encouragement	Break complex tasks into smaller steps that can be easily accomplished. Give feedback on actions, even if it's just a subtle glow.
P. Do the heavy lifting for me	Make novices feel like experts by enabling them to do things they never thought they could. For example, shortcuts that combine multiple photo effects can make amateur photographs look amazing in only a few steps.
Q. Make important things fast	Not all actions are equal. Decide what's most important in your app and make it easy to find and fast to use, like the shutter button in a camera, or the pause button in a music player.

Table 4. Android design principles from Make me amazing.

by offering the possibility to change the ring tone or the background image. This kind of customizations should be optional and are therefore correlating with game design element 1 - *Support autonomy*, which states the importance of letting the user decide if and how to take action within the application. This design principle is also correlating with game design element 2 - *Promote creation and representation of self-identity* since the settings makes the application more personal and thereby represents the individual user.

Design principle G - *Decide for me but let me have the final say* states that applications should be designed to get to know the user and take action based on prior use or a substantiated default. However, the user should still have the possibility to undo the action or change the settings. This possibility to control the application's behavior correlates with game design element 1 - *Support autonomy* since it supports the possibility to choose if and how actions are performed.

3.2 Correlation between Competence and Achievement and the Android Design Principles

Game design element 3 and 4 relates to the importance of the right level of challenge and to get feedback from the application. These design principles mainly concern facilitating and being consistent and can be mapped to a number of design principles from the Android design guidelines.

Design principle E - *keep it brief*, F - *pictures are faster than words*, H - *only show what I need when I need it* and P - *do the heavy lifting for me* all focus on keeping information brief and easy to understand. These correlates with game design element 3 - *Design for optimal challenge* by making it possible to perform actions faster with the feeling of mastering the necessary knowledge.

Design principle K - *if it looks the same, it should act the same* and M - *give me tricks that work everywhere* concerns the importance of being consistent. This gives a feeling of recognition and increases the probability for the user to succeed. These design principles correlates with game design element 3 - *Design for optimal challenge* because a user that succeeds is not challenged with a degree of difficulty that is too high.

However, it is hard to satisfy all users and eliminate all possible mistakes. But if an error occurs there are ways to treat the user with respect and give a positive feeling. Design principle N - *it's not my fault* states the importance of the right degree of difficulty of error messages. A carefully designed error message only gives the information necessary to correct the error and is expressed with non-technical words understandable by the user. This design principle is correlating with game design element 3 - *Design for optimal challenge* because even if there are mistakes occurring due to a too high degree of difficulty there are still possibilities to lower the challenge by providing clear error messages.

Design principle A - *delight me in surprising ways*, I - *I should always know where I am*, and O - *sprinkle encouragement* all concerns the importance of giving the user feedback. For example, a subtle glow, a well time sound and transitions when switching views can be used to tell the user what is going on. This is directly correlating with game design element 4 - *Provide timely and positive feedback*.

3.3 Correlation between Relatedness and the Android Design Principles

Game design element 5 - *facilitate human-human interaction* and 6 - *represent human social bond* originates from the motivational source relatedness. These game design element are the only ones that cannot be directly mapped to the Android design principles.

4 Example of a Gamified Android Application

One example of a gamified Android application is an exercise application called *Zombies, Run!*. The main focus for Zombies, Run! is to motivate the user to run, but instead of an ordinary exercise application tracking distance and time this application tries to take advantage of game design elements.

The background story of Zombies, Run! is that the human race has been infected with a contagion transforming the victims into zombies and the remaining people have to run, hide and defend themselves against these living dead. To use the application the user puts on the headphones and listens to his or her ordinary exercise music. The application then mixes the music with recorded conversations and other sounds that make the story more realistic. Now and then there are also sounds from zombies that are close which are signs for the user to run faster.

During the ongoing exercise the user can get missions from the characters in the story and automatically pic up supplies from the ground. These supplies are then stored in the application and can be used to level up the home village in the game.

4.1 How Zombies, Run! Takes Advantage of the Game Design Elements

Game design element 1 - *support autonomy* is accomplished by the application by letting the user change the settings. In the settings the user can for example chose the length of each mission and which distance unit to use for the distance tracking (see Figure 1). The settings make the application more suitable for people with different exercise habits and from different countries.

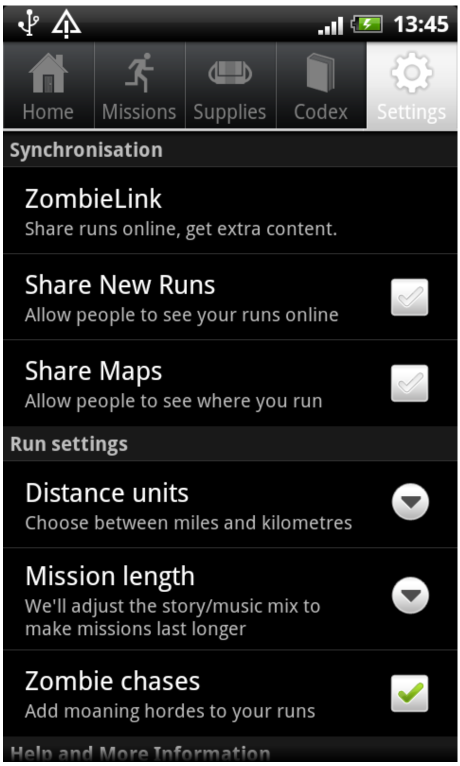

Fig. 1. Settings view in *Zombies, Run!*.

Even if the main purpose for the application is to motivate the user to run, the applications does not really care about if this is the case or if the user actually is riding a bicycle, walking or anything else. This unencumbered attitude correlates with game design element 3 - *designing for optimal challenge* since each user only need to run as fast as he or she is able to. The story however aims to motivate the user to try a little bit harder than usual and thereby gain a harder exercise. In addition to the story itself the application also uses the supplies as motivation. The more often the user runs the more supplies are collected. The user gets feedback in the headphones when a supply is collected. When the user finishes the exercise these supplies can be used to level up the buildings in the home village by dragging the supply onto the chosen building (see Figure 2). The more the user runs, the higher level will the village reach and the more missions will be available to play.

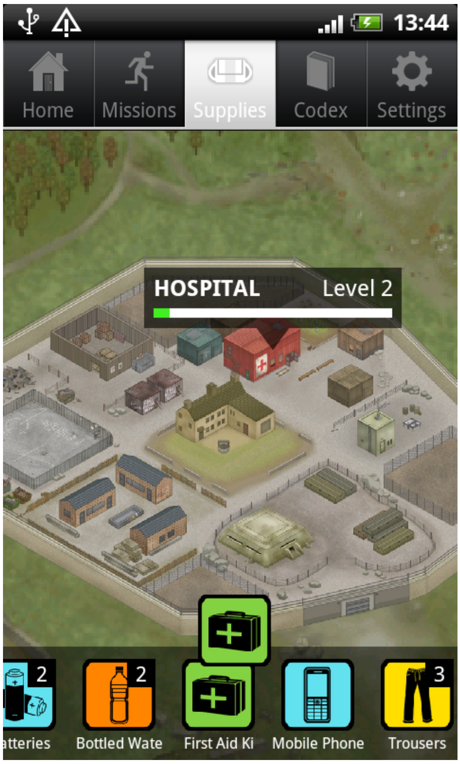

Fig. 2. Leveling up the home village in *Zombies, Run!*.

Except for these in play design elements there is also the availability to share maps and runs with other users. This function correlates with the need for relatedness and thereby game design element 5 - *facilitate human-human interaction*.

5 Discussion

Overall the identified game design elements match with the Android design principles. Game design element 1 - *support autonomy*, 2 - *promote creation and representation of self-identity*, 3 - *design for optimal challenge* and 4 - *provide timely and positive feedback* are all directly applicable on Android's design principles. Game design element 5 - *Facilitate human-human interaction* and 6 - *represent human social bond* are however exceptions. These are not directly correlating with the Android design principles. However, they do not have any contradictions which means that even if Android do not aim to necessarily fulfill them, Android still do not try to avoid them. This makes it possible to create good gamified applications for Android devices without infringing the Android design principles.

An explanation for game design element 5's - *Facilitate human-human interaction* and 6's - *represent human social bond* lack of correlation with the Android design principles is that the presence of these design elements in an application is strongly coupled to the aim of the application. One way to implement these game design elements, and thereby fulfill the psychological need for relatedness, is to connect the application to some kind of social network. This solution is however not suitable for all kinds of applications which explain the lack of correlation between these game design elements and the Android design principles.

5.1 Identified Advantages and Disadvantages

The aim of this article is to compare some game design elements used within gamification with the Android design principles. As already discussed there are no Android design principles that oppose the guidelines for gamification. Therefore no disadvantages with gamification in Android applications have been identified. However, most of the game design elements are more or less correlating with the Android design principles which can be considered prove the use of gamification. An advantage with the game design elements is that they are measurable in the way that it is possible to determine whether they are fulfilled or not by a simple 'yes' or 'no'. Another advantage is that there is no specification of how they should be accomplished, which makes it possible to apply gamification on a lot of different kinds of applications.

5.2 Future Work

The result shows that gamification is completely applicable on Android applications without infringing the android design principles. However, this is only a first attempt of an investigation and a more thorough analysis is required. This investigation is also completely theoretical and would need a more practical follow-up. One proposal of a more practical approach is to create two implementations of an Android application where one of these implementations respects the game design elements and the other does not. These implementations could be used to analyze users behavior and further investigate the topic.

References

1. Deterding, S., Dixon, D., Khaled, R., Nacke, L.: From game design elements to gamefulness: defining “gamification”. In: Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments. MindTrek ’11, New York, NY, USA, ACM (2011) 9–15
2. McDonald, M., Musson, R., Smith, R.: The Practical Guide to Defect Prevention. Microsoft Press (2008)
3. Dignan, A.: Game Frame: Using Games as a Strategy for Success. Free Press (2011)
4. Malone, T.W.: Heuristics for designing enjoyable user interfaces: Lessons from computer games. In: Proceedings of the 1982 Conference on Human Factors in Computing Systems. CHI ’82, New York, NY, USA, ACM (1982) 63–68
5. Churchill, D.: Conceptual model learning objects and design recommendations for small screens. Educational Technology & Society **14** (2011) 203–216
6. Zhang, P.: Technical opinion: Motivational affordances: reasons for ict design and use. Commun. ACM **51**(11) (nov 2008) 145–147
7. Ryan, R.M., Deci, E.L.: Intrinsic and extrinsic motivations: Classic definitions and new directions. Contemporary Educational Psychology **25** (2000) 54–67
8. Grolnick, W.S., Ryan, R.M.: Autonomy in childrens learning: An experimental and individual difference investigation. Journal of Personality and Social Psychology **52**(5) (1987) 890–898
9. Deci, E.L., Ryan, R.M.: The “what” and “why” of goal pursuits: Human needs and the self-determination of behavior. Psychological Inquiry **11**(4) (2000) 227–268
10. Sheldon, M.K., Bettencourt, B.A.: Psychological need-satisfaction and subjective well-being within social groups. British Journal of Social Psychology **41** (2002) 25–38

Interaction without Taking Action – Implicit Interaction Design in a Continuous Facial Recognition Identification System

Sabina Sonning

Department of Computing Science
Umeå University, Sweden
saso0014@cs.umu.se

Abstract. This paper examines a gate passage system with continuos facial recognition as identification method. We look at the system from the perspective of three frameworks and attempt to identify weaknesses and potential in the design solution. It is one part analysis of the system and one part application of frameworks, putting them to use. In theory and practice we explore the design issues related to implicit interaction systems using sensors with movement and image as input.

1 Introduction

In this paper we will examine the ramifications of removing traditional physical input actions in a gate system that requires identification.

In most available gate systems without human operators the user presses a PIN code, swipes a card, carries a RFID tag and/or allows biometric recognition by placing a thumb on a pad, an eye in front of a lens or speech by input. All these methods offer a pairing of identification artifact or personal imprint with a system sensor that requires user decision and action to take place.

The studied system is a gate for members and personnel of IKSU, a large training facility situated in Umeå, Sweden. Normally, close to 20 000 [1] users swipe a member card or punch a sequence of numbers as identification method when standing at the gate (the latter being much less common and mostly for IKSU personnel), which then opens for a short time, allowing passage. The solution is recognized and well used, but has the following weaknesses in maintenance cost, security and in interactional quality.

- The wear on cards and card readers cause passage break downs, hinderingen members to enter, and this in turn put high pressure on reception and maintenance personnel.
- The user is required to remember a code or carry and produce a card before entrance. Users frequently forget their cards, seeking assistance from the reception that has to be fully manned at most opening hours for this and the reason above, at great cost. Need of assistance is so common that some non-member users have started pretending to forget their card or have card troubles, to be allowed in without paying the fee.

S. Bensch, F. Drewes (Eds.): Umeå's 16th Student Conference in Computing Science USCCS 2013, pp. 61–76, January 2013.

– The procedure requires the users to stop in front of the gate to be identified. This pause before entering adds to the queue forming during rush hours and is a small annoyance for any user carrying a trunk and layers of clothing being in a rush.
– Finally, from a security perspective, the card and code system allows for multiple users to share identification tokens, making it easy to cheat the system and avoid paying full price for facility access.

We will study a pilot project of a gate using continuous facial recognition as identification method and the ramifications of removing the expected decision and action phase from the interaction paradigm. As this system is built to identify and verify members before they reach the gate, it requires no learned behavior associated with an identification process and operates on the premise that users should just be able walk to the gate and enter. The face recognition sensor is not necessarily visible or comprehensibly a sensor from the user perspective and no action is required or means offered to activate or deactivate it. This poses an interesting challenge in how to communicate the desired behavior to the member, avoid hesitation based on expectations derived from previous experiences with gates, and how to give feedback of success, failure or suggested corrections without interrupting the flow made possible by this approach.

The system involves several technologies relatively new to the consumer field, namely continual facial recognition on a large member base together with a depth sensor, motivating an analysis of the interaction design and user experience of such a system.

We will use interaction design guidelines and themes as proposed by Benford et al, Ju et al. and Belotti et al.[2–4] to identify key issues in relation to the user studies, and describe how the studied system handles these problems as well as suggest how to address them in the future. Ju and Leifer [3] uses a doorman and an automatic door example to explore the design space of implicit interactions. This makes their framework highly interesting for this paper, both in reference to the similarity of the use case to the gate passage system and in the direct addressing of the implicit interaction design challenge. Belotti et al. [4] address how to translate the *cognitive* aspects of interaction, as proposed by Norman's by seven stages of execution [5], to emphasize on the *communicative* aspects more relevant in an implicit interaction system. Benford [2] focus on the activities in and around the system, categorizing them by groups of expected, sensed and desired to explore the design solution.

2 Continuos Facial Recognition as Identification Method

To address the issues with the old system described in the introduction, the new pilot system running at IKSU on one of the gates, replaces the card and code identification method with continuos facial recognition paired with a depth sensor. The first instance of the pilot in 2011 ran successfully on a test group of about 100 members and the second is being set up with a larger test group as this paper is written in the fall 2012. This time with a stand-alone registration unit, improved algorithms and system stability. The existing gate and member records are used, with an addition of identification data extracted from each members facial image connected to the records. This aims to reduce cost for replacing worn out or lost cards and the maintenance of the system card readers. There is also no way to forget your face at home and potentially much harder to borrow or share identities with other users, "by using biometrics it is possible to confirm or establish an individual's identity based on *who she is*, rather than by *what she possesses* (e.g., an ID card) or *what she remembers* (e.g., a password)" [6]. Adding a depth sensor technique enables tracking of identified users in the gate area and limits the scope for the facial recognition modules facial detection, speeding up the process. Continuos identification allows for less queue forming as users never have to stop for identification, tested in the first pilot by multiple users approaching the gate simultaneously. More complex human interactions are still to be tested, as well as a larger database effect on system speed, potentially reducing the positive effect on queue reduction. A simplified overview of the system with its modules and activity flow can be seen in Figure 1. The figure shows the four main modules (Facial recognition and RGB-camera, Depth sensor with Kinect camera, Member records and Gate) and the activity flow in a successful sequence of interaction.

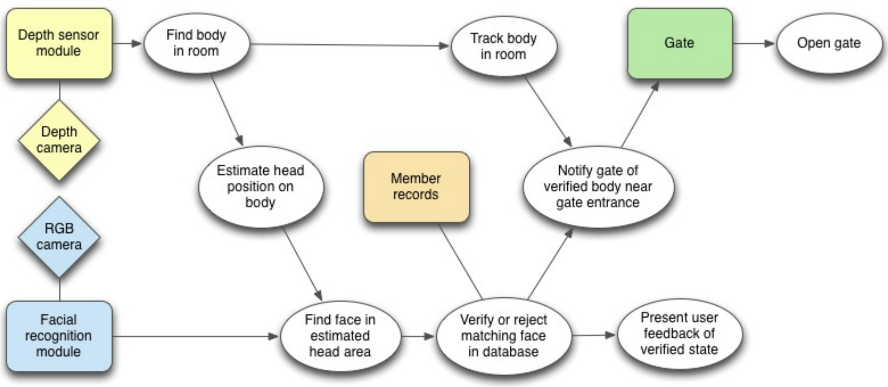

Fig. 1. Simplified pilot system overview.

2.1 Depth Sensor Module

The depth sensor module is built around a Microsoft Kinect-camera that projects a grid of IR-light dots and calculates depth based on distortions of that grid [7]. The hardware was designed for the gaming system Xbox and offers a cheap and efficient depth sensor with an open source interface. Grouping each depth reading in segments based on vicinity and depth variation, and discarding segments representing reading smaller than humans, identifies readings of bodies. Overlapping bodies, from the sensor perspective, is separated by depth thresholds (see Figure 3), enabling several bodies to be observed at the same time.

Each segment is tagged and tracked and a naive position of the head is calculated based on segment shape (see Figure 2). This head position is sent to the facial recognition module to limit the area of search for face detection and face recognition.

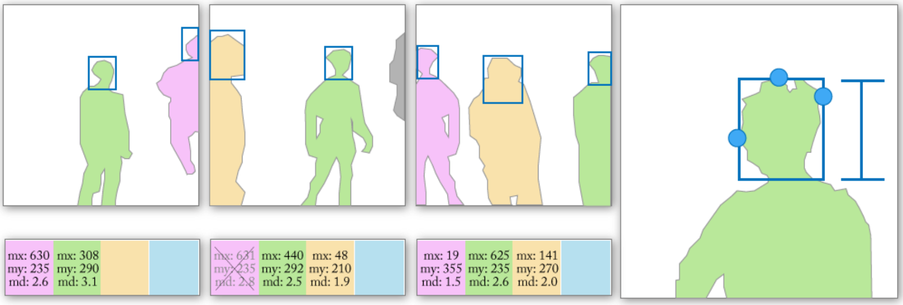

Fig. 2. Segment tracking and calculation of head position.

As soon as a user steps within range of the Kinect camera an overhead screen by the gate displays feedback of this body silhouette mirroring the users movement in the room in a bright orange color. When a user is identified the silhouette turns green (see Figure 3).

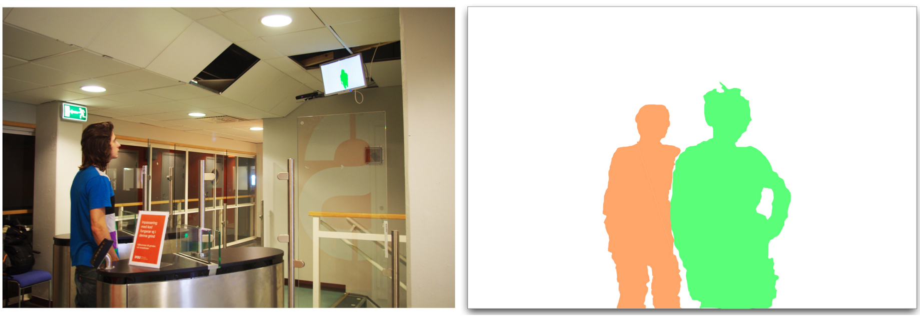

Fig. 3. User feedback of tracking and identification.

2.2 Facial Recognition Module

Based on the position data from the depth sensor module, calibrated to fit the scope of an RGB-camera, the facial recognition module researched and developed by Hung-Son Le and later WAWO Technology research company [8], first uses an open source component from OpenCV [9] with face detection to find a face. It is set to accept very broad definitions of a face, including strange angles and other imperfections, a compromise until a more reliable face detection algorithm can replace the OpenCV one. If a probable face has been found, as a combination of the body segments calculated head position and the OpenCV face detection on that position, the image data is compared with those in the member records using a Hidden Markov Model [10], constituting the face recognition. Several attempts might be needed before recognition is complete and this is generally the case. Preliminary test data suggest that users walking towards the gate with their face fully visible and turned towards the camera require 1-30 attempts before identification, about 10 in average, or approximately 1-3 seconds. Taking into account the actual distance of about 7 meters from the corner turned where the gate is visible, the perceived time of identification is 3-5 seconds on the database of 100 members. These numbers are all estimated based on system tests conducted by us during development but must be viewed as complementary information for now, here to give a better sense of the system capacity, as they are preliminary.

3 System Interaction Analysis

To narrow the scope of this paper we will focus on interaction with the gate, leaving out the registrations process that is necessary before using the facial recognition system, deserving full attention at another time. The gate still offers the option to use magnetic card or pin code as identification method for users not in the pilot and both groups intermix while the pilot study is running. We will also leave the topic of integrity concerns related to surveillance and storing of personal information aside, since this is such a large topic and only partially related to the interaction experience. First we present the old and new procedures:

- **Old system:** The user walk towards the closed gate, stop by the gate with a visible card reader and a small red lamp, produce a magnetic card to match the input device, slide the card through the reader, system verifies identity of card, the lamp turns green and the gate opens.
- **New system:** The user walk towards the closed gate and see movement of an orange silhouette that is an abstraction of the users body and movements on a display, while walking the system finds the users face, identifies it in the member records and turns the silhouette green, as the user reaches the gate it opens automatically.

3.1 User Tests

Enrolled users have been asked to enter the gates, either alone or as a group, while being observed. We have also studied normal traffic through the gates and later asked the pilot group to fill out a questionnaire, and some to participate in more thorough interviews. The interviews and questionnaire was mostly focused on comparisons to the old system, frequency of use and opinions about using facial recognition, making the observations and user tests the most relevant part for this paper. All registered users are given cues about how to behave and how to improve recognition while going through the registration process, in the form of images and instructions. That is still not a guarantee that they have integrated that information and should not be the sole way to understand how to interact with the gate.

3.2 Expected, Sensed and Desired

The framework by Benford et al. [2],*Expected, Sensed and Desired: A Framework for Designing Sensing-Based Interaction*, assists in the design of sensing-based interfaces and puts a special focus on the comparison of physical form and actions to what can be sensed by a computer system. It looks at the relationship between those in terms of what is expected, sensed and desired. An important attribute of this framework is that the categories of movements only partially overlap and that the parts where they do not can help visualize problems but also unexpected possibilities, potentially useful in the design solution.

- **Expected** movements are those movements that users might be expected to carry out in the specific context of the user(s), interface and environment. For the gate passage system, a closed gate with a display screen, a depth sensor and a RGB-camera, expected movements include everyday movements in the area inside and outside the gate, like walking, engaging in conversations with other people in the room or on the phone, sitting, standing and looking at the display.
- **Sensed** movements are those that can be measured by a computer based on input from the sensing technologies in that system. For the gate passage system there are two different sensors with different sensory areas (not counting the built in gate safety sensors that control the closing of the gate after one person has passed): The depth sensor (Kinect) represented in yellow and the RGB-camera in blue in Figure 4, where the unity (green) of the areas result in the full system frame of view. The Kinect is mounted from the ceiling (labeled 2 in the figure), limiting the sensory capacity very close to its position because of the limited vertical view angle and has an effective range of about 6 meters. The grid of IR-light projection actually gives stable readings on as far as 8 meters but the OpenCV library does not support more than 6 meters. The RGB-camera range is only limited by the angles of the room and furniture, but the facial recognition module working on the images require the face captured to be of approximately 100x100 px to work properly. Because of the way the application for a permit to place a camera in a public environment was formulate, the RGB-camera had to be fastened on the wall behind the gate (label 1 in the figure), reducing the identification range noticeably. The consequence is the orange and yellow areas, where the user feedback from the display (also mounted from the ceiling, labeled 3 in the figure) suggest that the user is within system sensor range, but in fact, the identification process only works satisfactory after passing the dotted line labeled 5 in the figure.

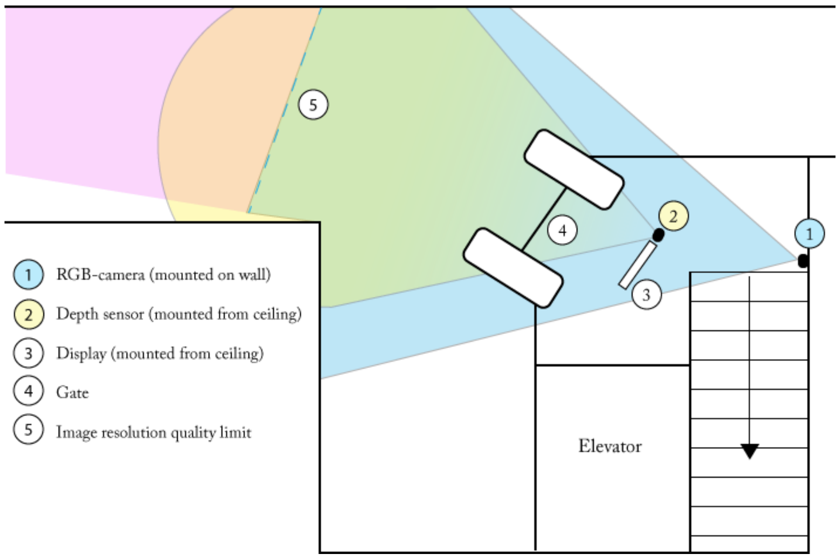

Fig. 4. Model of room, gate, sensors and display of the gate passage system.

– **Desired** movements are those that are required for the system to work in the intended, or at least some satisfying way. To pass the gate the user has to have their face turned towards the RGB-camera (long enough and within image resolution quality range) for the RGB-camera to catch an image of high enough quality to use for identification. They also have to move close enough to the gate at some point before leaving the Kinect sensory view for the gate to open.

We applied the Benford et al. framework to an analysis of the gate passage system, regarding the user movements and placing the different types of movements in the categories. Even without decision and action, the user movements are in fact input if perceived by the system. The groups of movements and behaviors in relation to the framework can be seen in Figure 5. Some movements and situations were hard to categorize since the "sensed" aspect has a fuzzy definition due to the overlap of two sensors and the limitations of the calculations done on the input. We limit the "sensed" category to be the green area in Figure 4 where the Kinect and the RGB-camera scope overlap and where the captured images have high enough quality, even though users might experience being sensed because of the feedback on the display being activated over a larger area.

– **A: Expected, sensed and desired** movements, where users walked towards the gate with their face faced towards the display and with a relaxed expression. They continued to the gate, which opened at arrival. This group represents the intended flow, where the users never stopped in their stride and included those users arriving in smaller groups, sequentially identified and let in in order of arrival.

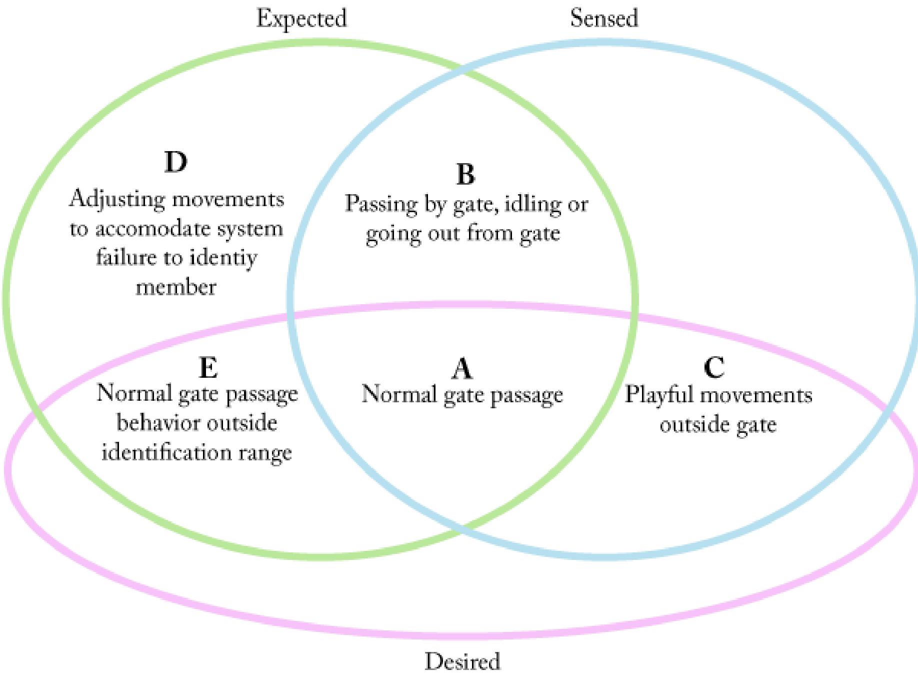

Fig. 5. Venn diagram of movements grouped according to the Benford et al. framework.

- **B: Expected and sensed, but not desired** movements around the gate where users did not intend to be identified. Most users were not identified during these movements, keeping their face turned more towards their goal than on the display, or passing through the green area too quickly for identification. But some that looked up at the display were identified, their silhouette turned green. The gate never opened for those passing by or standing near the gate idling, since they never moved closed enough to the gate, but this is certainly a possibility.
- **C: Sensed and desired, but not expected** movements outside the expected domain but with interesting behaviors. Users moved outside the gate, interacting with the display feedback, without entering on identification. Some waved, sidestepped, went out of the room and than back again, exploring the silhouette mirroring of their own movements and testing the system limits alone or with friends, often in what could be interpreted as a playful manner.
- **D: Expected, not sensed or desired** movements, where users tried to get in, acting as they perceived appropriate but failing to be identified. Thus actually moving in the green area of Figure 4 but not beings sensed. They would first walk towards the gate and when being rejected, go back and forth on a line 1-4 meters directly in front of the gate, angle their face, change expression, adjust accessories and so on until either let in or giving

up. The failed attempt resulted in the users either producing the magnetic card or pin code as a replacement identification method or them going to the help desk to be let in manually. This was observed at the gate even when the system was inactive, the display showing a desktop instead of the silhouette feedback. Also when users (identified or not) moved too close to each other, their silhouettes melting in to one, and failed to enter because of the tracking algorithms limitation to handle such incidents. This group had bigger problems adjusting since moving out of the depth sensor view and re-entering is the only solution today.

– **E: Expected and desired, not sensed** movements in the orange and yellow zone of Figure 4, where users started to look at the display, walking with a natural expressions, unknowing of the systems limits to where identification could take place. Also people moving out from the member gym from the other direction, their head turned away from the sensors.

The framework relates to other existing taxonomies that explore sensed and expected movements [11, 12] but is differentiated by its focus on overlaps and the not sensed and not expected aspects. For the gate passage system it reveals that the design mostly communicates *where* users can be sensed, except for the orange and yellow area in Figure 4, through the display feedback, accommodating group A and B in Figure 5. The small area of discrepancy involving group E, where users think they are sensed but really are not, did not pose a problem since those experiencing problems only perceived it so when much closer to the gate and tried to adjust this inside the green zone as group D, at closer range. The yellow and orange area could be viewed as an area of preparation for the user, without demanding resources from the identification module.

It also reveals that users enjoy interaction with the system for other purposes than entering the gate and this is how we interpret group C. The playful movements observed outside the gate indicate design success at least in aspects of enjoyability and indicates that this sensed and desired overlap could be developed further.

On the other hand, the system does a poor job at communicating *how* to be sensed. That is, what behavior is suitable and even more so, what type of behavior, if any, could remedy identification problems. This goes for the system state in general, as some users tried to pass through the gate using their face when the system was turned of during updates. Even though the display showed only a computer desktop interface instead of the silhouette feedback, these users were expecting the system to work because they had no alert to the opposite.

It is interesting, that even though the framework is built on theories for input and input based devices [11, 12], it seem to suit the analysis of systems no matter if the interaction style is "implicit" or "explicit". We will look further in to the distinction between these interaction styles in the next section, with the support of Wendy Ju and Larry Leifers framework.

3.3 Implicit Interactions

The framework by Ju and Leifer [3], *The Design of Implicit Interactions: Making Interactive Systems Less Obnoxious*, gives interaction researchers and designers a tool to understand the difference between "implicit" and "explicit" interaction and to give guidance on when implicit interactions are useful. They describe implicit interaction as exchange either initiated by the computer system rather than the user, or as an exchange that occurs outside the focus of user attention. This is a relevant distinction from the interaction that considers explicit input and output sequences, that according to the authors has been the main domain for traditional HCI, with a history of exploring devices and systems with such procedures. The framework is limited to interactions between one user and one system and has a quadrant layout that covers the two types of implicit interactions and their counterpart, the explicit, by exploring interaction design in concepts of attentional demand and initiative. Ju and Leifer tie their attentional demand concept (vertical axis in Figure 6) to Bill Buxton's attentional ground. They argue however, that his definition only considers user-initiated interactions and that this results in a conflation of *attention* and *intention*. By decoupling these, a separate dimension emerges: *initiative* (horizontal axis in Figure 6).

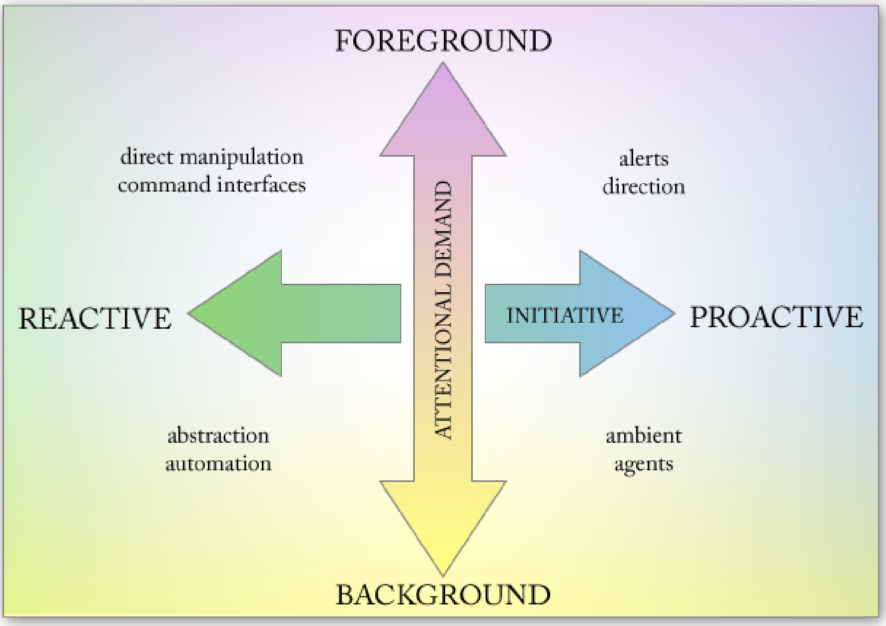

Fig. 6. Attentional demand and initiative quadrant from Ju and Leifer's framework.

We applied Ju and Leifer's framework to the new and old gate passage system by classifying the user and system activities level of initiative and attentional demand and placing them in the quadrants. By comparing the two approaches we hope to make the differences more tangible. Recall the two interaction procedures:

- **Old system:** The user walk towards the closed gate, stop by the gate with a visible card reader and a small red lamp, produce a magnetic card to match the input device, slide the card through the reader, system verifies identity of card, the lamp turns green and the gate opens.
- **New system:** The user walk towards the closed gate and see movement of an orange silhouette that is an abstraction of the users body and movements on a display, while walking the system finds the users face, identifies it in the member records and turns the silhouette green, as the user reaches the gate it opens automatically.

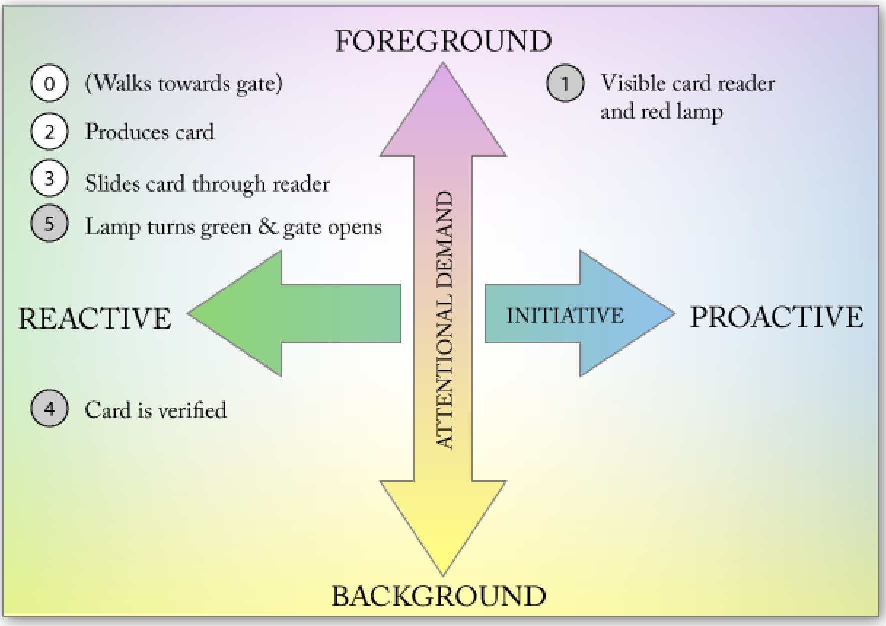

Fig. 7. The old gate passage system in the Ju and Feifer quadrants. Activities are labeled 0-5, where white represent the user and gray the system.

For the old system shown in Figure 7, we see that most activity take place in the upper left quadrant representing a HC-conversation of human input and machine response as output. It requires engagement and physical activity, as well as a bit of the users time. The new system, designed for implicit interaction, reduce the user input to "walking" and proactively identifies users before reaching the gate without user initiation. But, as the authors of the framework also mention, a problem with proactive systems is that they can be experienced as presumptuous when they make decisions without user consent, an issue even more prominent

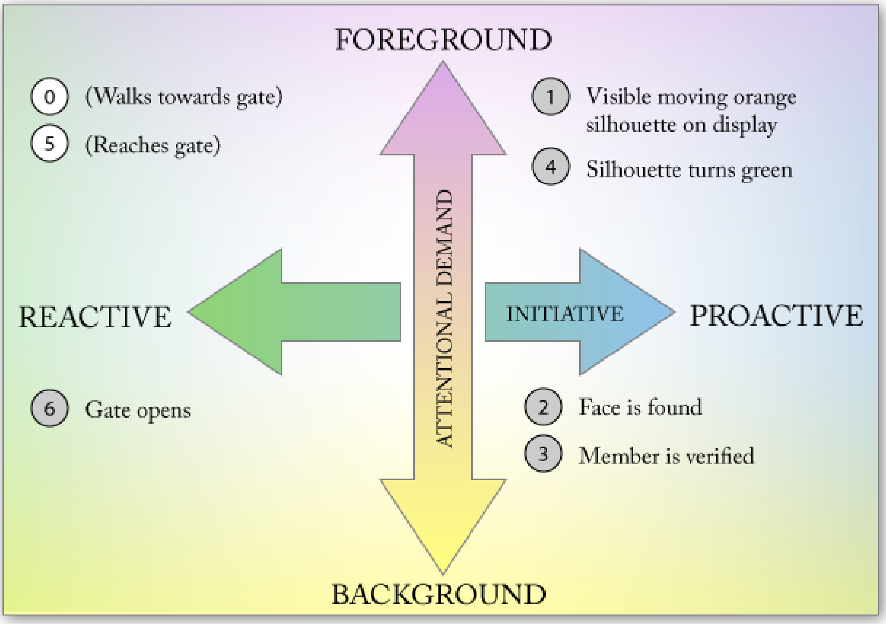

Fig. 8. The new gate passage system in the Ju and Feifer quadrants. Activities are labeled 0-6, where white represent the user and gray the system.

if integrity concerns are included in the analysis. The users (and non-users for that matter) of the gate passage system can not decide *not* to let the system try identification in any other way than hiding their face while passing through the room. The question of users lacking insight in the cause and remedy to system failure, mentioned in the earlier section of the Benford framework, becomes very visible with the Ju and Feifer quadrants (see Figure 9).

There is no way for the user to know *when* the problem occurred, making it presumably harder to know *why*, and allowing user diagnosis of system failure or insufficient compliance to some invisible limitation. Compare to the old system where sliding the card either lights the lamp green or not. If it turns green and the gate still does not open, the user can deduce that the gate mechanism is broken and report to the help desk. If sliding the card does not light the lamp, the problem lies in either the card being incorrectly swiped, the card reader itself, the magnetic card or the validity of the membership. The user might try sliding the cards a couple of times more, trying different ways of insertions, and if eliminating that possible source of error visit the help desk. This ties back to one of the initial incentives of IKSU, to reduce pressure on the help desk by removing the many issues with card and card reader reliability. But is still a preferable situation for the user in regard to self confidence and self efficacy, to be able to diagnose the reason for failure, at least to make sure it is not related to his or her own oversight or “mistake”.

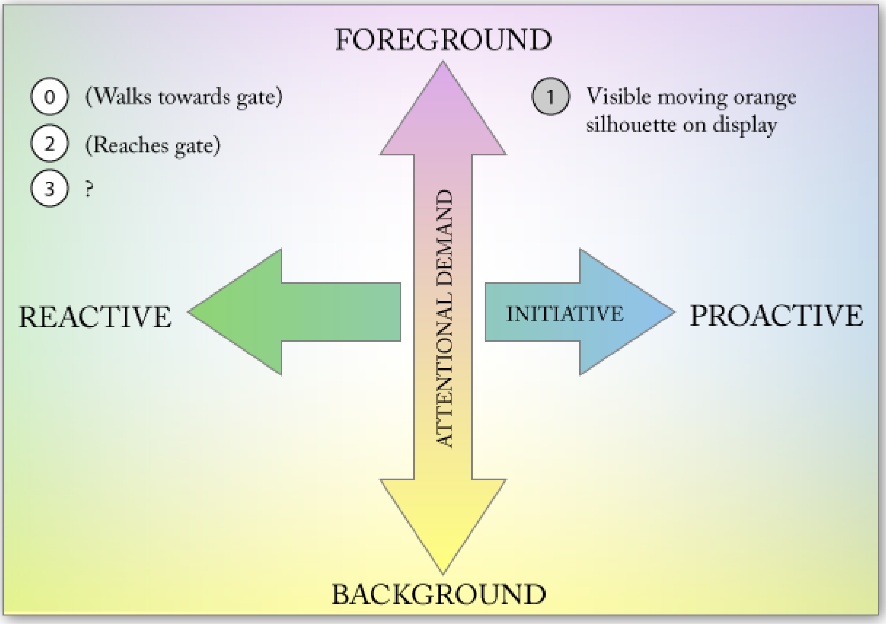

Fig. 9. The new gate passage system in the Ju and Feifer quadrants. Activities are labeled 0-3, where white represent the user and gray the system.

Another example is when multiple users are standing in front of the gate. The user closest to the gate is the first one on the systems list to identify. If this user is not a member or if she or he are standing with their face not visible, engaged in another activity, like speaking in the phone for example, the other approaching user will never be identified. That user has to be closer to the gate to move up in queue, a rule completely invisible for the users.

3.4 Summary and Making Sense of Sensing Systems

We look at the system using Belotti's use perspectives from *Making Sense of Sensing Systems: Five Questions for Designers and Researchers*[4], based on Norman's seven stages of execution [5] to summarize our finds.

– **Address:** The direction of communication with a system. In our case, facing the display and by doing so facing the camera, and the movement towards the gate. Wearing a neutral expression and fully visible face is necessary, but this accommodation comes naturally as the users know that their face is their key. Compare to a passport control at a foreign airport, where users give a human controller the photo for comparison. They do not hide their face behind a hat or scrunch their face, they try to help the officer by looking like the photo, because their objective is to be let in. It can of course be hard to communicate that the facial recognition has a harder time pairing a face with a big smile to a neutral photo than a passport controller. A way to give feedback of appropriate facial expressions could be the solution.

- **Attention:** Establishing that the system is attending. In our case, the feedback of the silhouettes. In most cases this is enough for the users in terms of understanding that the system is attending. We have identified some question marks however: The orange zone in Figure 4 where users get the attention feedback but actually are outside range of identification, and the small group of users that did not connect the feedback at all with system attention. At least for the last group, that could be remedied with a simple “system off” state presented on the display as a fallback.
- **Action:** Defining what is to be done with the system. There is only on purpose with the system from a member perspective, to be recognized and let in. The addressing is part of the action as this is what the system needs to succeed, and the movement to the gate is the other component.
- **Alignment:** Monitoring system response. The response being the user silhouette turning green and finally the door opening on approach. Taking in to consideration the weaknesses exposed in diagnosing accidents, where users do not know where in the chain of system activities the mistake happened, this is an area with room for improvement. By adding feedback of *when* the system is trying to identify the user, the problem of identification order mentioned in the section before, where an idling user near the gate can block the identification of another user trying to get in, could be addressed. This does not help the non-members attempt to enter. The system will obviously try to identify these users to and there is no way to discern if the reason for rejection is because the user has an expression that is too different from the image or if the user just is not in the member records.
- **Accident:** Avoiding or recovering from errors and misunderstandings. For the gate passage system, this is where we find the most issues. Improving alignment could reduce accidents but the question remains in how to succeed in doing this without robbing the system of its implicitness, and with that the benefits of a quick, low attention solution.

3.5 Conclusions

When setting out to write this paper we assumed that one of the key issues with the smooth gate passage system was that user would *expect* to behave in one way and be confused by the missing “action” required to enter the gate. This does not seem to be the most prominent reason for interaction problems in this system, a conclusion supported by Chalmers et al. in that “studies of use consistently point out that accommodation and appropriation are key of the adoption of new technologies: users design their activity to fit ’our’ technologies...” [13]. Enrolled users know or learn that they only need to make their face visible to enter and accommodate by looking up at the display, drawn in by the reflected movement. During the gate observations we noticed a couple of users that were not part of the study who mimicked the behavior of the pilot users. They obviously did not know that they needed to have their picture taken before entering, still, they saw people entering the gate without a card and wanted to do the same. This could be seen as a form of situated learning, where users can learn how to interact with the system by participating in that community of practice [14]. The learning is limited, since these users know no more after a failed attempt due to the lack of feedback, but shows promise for the allure of the system and the simplicity of the interaction for enrolled users and learning between these. The more playful interaction with the silhouettes not planning to enter, could even be interpreted as a form of *epistemic action*, where users manipulate their movements to understand the context and how their actions relate to the system, like Tetris-players rotating pieces to understand how different options would work [15]. It is rather that feedback of system state and response is lacking, especially in error handling.

The frameworks have been a useful tool to highlight the problems with the system and, to some extent, find where improvements could be made. Benford et al. show the unexpected but desired behaviour for further exploit and visualize that discrepancy between the sensed and user perceived sensed areas of the system. It is a great tool in asking questions, but have little to say on how to remedy the problem areas. Ju et al. helps us formulate that the deeper problems with the system interaction lies in the lack of feedback when something goes wrong but has little help to give in how to actually handle the transition from implicit to explicit interaction when this would be needed. The fine line between making an elegant, low attention system and one that handles error well is still not fully drawn. As Belotti et al. points out "There has been little discussion in the Ubicomp literature so far concerning failure modes and errors. [...] As things stand in sensing systems, our accident-avoidance challenges, though serious, are largely unaddressed. [..] Future, more ambitious systems will most likely need to provide a wide range of mechanisms for dealing with the common problem of error." Their framework focus on a more or less balanced communication between system and user, not really taking into consideration the implicit and therefore give little new to the analysis in our opinion. For simple systems they mention to design for no errors, a slightly discouraging note. This together with improved feedback of system states is at least a working angle for how to address the systems future development.

On the positive side, as mentioned before, the system seem to encourage a playful behavior, a potential ground for learning and recovering from failure in the overlapping space between implicit and explicit interaction. If changing expression or slowing down to help the system is not perceived as annoying but part of communication that is satisfying on other levels than just getting a point through, tolerance for error could perhaps be extended.

References

1. IKSU: Iksu history (in swedish) (2009) `http://www.iksu.se/om-iksu/idrottsforeningen-iksu/iksus-historia/`.
2. Benford, S., Schnædelbach, H., Koleva, B., Anatasi, R., Greenhalgh, C., Rodden, T., Green, J., Ghali, A., Pridmore, T., Gaver, B., Boucher, A., Walker, B., Pen-

nington, S., Schmidt, A., Gellersen, H.: Expected, sensed and desired: A framework for designing sensing-based interaction. ACM Transactions on Computer-Human Interaction **12**(1) (2005) 3–30

3. Ju, W., Leifer, L.: The design of implicit interactions: Making interactive systems less obnoxious. Design Issues **24**(3) (2008) 72–84
4. Belotti, V., Back, M., Edwards, W.K., Grinter, R.E., Henderson, A., Lopes, C.: Making sense of sensing systems: Five questions for designers and researchers. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. (2002) 415–422
5. Norman, D.A.: The Design of Everyday Things. Doubleday: New York, New York (1990)
6. Jain, A.K., Ross, A., Prabhakar, S.: An introduction to biometric recognition. IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Image- and Video-Based Biometrics **14**(1) (2004)
7. Villaroman, N., Rowe, D., Swan, B.: Teaching natural user interaction using openni and the microsoft kinect sensor. In: Proceedings of the 2011 conference on Information technology education. (2011) 227–232
8. Le, H.S.: Face recognition - a single view based hmm approach. Technical Report SE-901, Dept. of Applied Physics and Electronics, Umeå University, Umeå, Sweden (2008)
9. OpenCV: Opencv wiki (2012) `http://opencv.willowgarage.com/wiki/`.
10. Rabiner, L.R., Juang, B.H.: An introduction to hidden markov models. ASSP Magazine, IEEE **3** (1986) 4–16
11. Buxton, W.: There's more to interaction than meets the eye: Some issues in manual input. In: User Centered System Design: New Perspectives on Human-Computer Interaction. (1986) 319–337
12. Jacob, R.J.K., Sibert, L.E., McFarlane, D.C., M. Preston Mullen, J.: Integrality and separability of input devices. acm transactions on computer-human interaction. ACM Transactions on Computer-Human Interaction (TOCHI) **1**(1) (1994) 3–26
13. Chalmers, M., Galani, A.: Seamful interweaving: Heterogenity in the theory and design of interactive systems. In: Symposium on Designing Interactive Systems. (2004) 243–252
14. Klemmer, S.R., Hartmann, B., Takayama, L.: How bodies matter: Five themes for interaction design. In: DIS '06 Proceedings of the 6th conference on Designing Interactive systems. (2006) 140–149
15. Maglio, P.P., Kirsh, D.: Epistemic action increases with skill. In: Proceedings of Cognitive Science Society. (1996) 391–394

Author Index