Game analytics

faculteit industriËle

ingenieurswetenschappen

**Campus Groep t leuven**

Verzamelen en interpreteren van visuele data van Solitaire met behulp van openCV en C++

Bram HERMANS

Marnik ROOSEN

Masterproef ingediend tot het behalen van de graad van master of Science in de industriële wetenschappen: elektronica-ICT – optie internet computing

Promotor(en): Prof. V. Vanden Abeele

Co-promotor(en): K. Gielis, S. Puttemans

Academiejaar 2017-2018

© Copyright KU Leuven

Without written permission of the supervisor(s) and the author(s) it is forbidden to reproduce or adapt in any form or by any means any part of this publication. Requests for obtaining the right to reproduce or utilise parts of this publication should be addressed to KU Leuven, Campus GROUP T Leuven, Andreas Vesaliusstraat 13, 3000 Leuven, +32 16 30 10 30or via e-mail fet.groupt@kuleuven.be

A written permission of the supervisor(s) is also required to use the methods, products, schematics and programs described in this work for industrial or commercial use, and for submitting this publication in scientific contests.

Acknowledgements

Het voorwoord vul je persoonlijk in met een appreciatie of dankbetuiging aan de mensen die je hebben bijgestaan tijdens het verwezenlijken van je masterproef en je hebben gesteund tijdens je studie.

Abstract

De (korte) samenvatting, toegankelijk voor een breed publiek, wordt in het Nederlands geschreven en bevat **maximum 3500 tekens**. Deze samenvatting moet ook verplicht opgeladen worden in KU Loket.

The Abstract should be viewed as a miniature version of the paper. Since potential readers should be able to make their decision on the personal relevance based on the Abstract, the Abstract should clearly tell the reader what information he can expect to find in the paper. The Abstract should (1) state the principal objectives and scope of the investigation, (2) describe the methods employed, (3) summarize the results, and (4) state the principal conclusions. Most of the Abstract will be written in the past tense, because it refers to work done. The Abstract should never give any information or conclusion that is not stated in the paper. The authors should always keep in mind that the Abstract is the most frequently read part of a paper. This abstract is a short summary that also needs to be uploaded to KULoket. It is written in the main language of your thesis text. Due to technical limitations on KULoket, this part may not exceed 3500 characters, and should be printed in one paragraph. Do not cite references in the Abstract. Do not include or refer to tables and figures. Please respect the formatting of the template and use the styles provided by the template. Do not change fonts, font sizes, margins, etc.

Extended Abstract

Het extended abstract of de wetenschappelijke samenvatting wordt in het Engels geschreven en bevat **500 tot 1.500 woorden**. Dit abstract moet **niet** in KU Loket opgeladen worden (vanwege de beperkte beschikbare ruimte daar).

**Keywords**: Voeg een vijftal keywords in.

About four key words or phrases in alphabetical order, separated by commas. Given the importance of web-searching, it is essential that authors make every effort to ensure articles are found online, quickly and accurately. The ”key” to this is the appropriate use of keywords. Authors should know the key phrases for their subject area. Reference to an established common indexing standard in a particular discipline is a useful starting point (for instance ChemWeb).

Contents

[Acknowledgements ii](#_Toc513654356)

[Abstract iv](#_Toc513654357)

[Extended Abstract vi](#_Toc513654358)

[Contents viii](#_Toc513654359)

[List of symbols ix](#_Toc513654360)

[List of abbreviations xi](#_Toc513654361)

[1 Introduction 1](#_Toc513654362)

[1.1 Related work 1](#_Toc513654363)

[1.2 Requirements 1](#_Toc513654364)

[2 Design 1](#_Toc513654365)

[2.1 Card extraction 1](#_Toc513654366)

[2.1.1 Introduction 1](#_Toc513654367)

[2.1.2 Methods 2](#_Toc513654368)

[2.1.3 Results 4](#_Toc513654369)

[2.1.4 Discussion 4](#_Toc513654370)

[2.2 Card classification 4](#_Toc513654371)

[2.2.1 Introduction 4](#_Toc513654372)

[2.2.2 Methods 5](#_Toc513654373)

[2.2.3 Results 5](#_Toc513654374)

[2.2.4 Discussion 5](#_Toc513654375)

[2.3 Mouse clicks 5](#_Toc513654376)

[2.3.1 Introduction 5](#_Toc513654377)

[2.3.2 Methods 5](#_Toc513654378)

[2.3.3 Results 5](#_Toc513654379)

[2.3.4 Discussion 5](#_Toc513654380)

[2.4 General 5](#_Toc513654381)

[3 Development 6](#_Toc513654382)

[4 Discussion 6](#_Toc513654383)

[References 6](#_Toc513654384)

[Appendix 7](#_Toc513654385)

List of symbols

*Maak een lijst van de gebruikte symbolen. Geef het symbool, naam en eenheid. Gebruik steeds SI-eenheden en gebruik de symbolen en namen zoals deze voorkomen in de hedendaagse literatuur en normen.*

*De symbolen worden alfabetisch gerangschikt in opeenvolgende lijsten: kleine letters, hoofdletters, Griekse kleine letters, Griekse hoofdletters. Onderstaande tabel geeft het format dat kan ingevuld en uitgebreid worden.*

*Wanneer het symbool een eerste maal in de tekst of in een formule wordt gebruikt, moet het symbool verklaard worden.*

*FACULTATIEF*

List of abbreviations

**Druk <Ctrl + Alt + Shift + S> om het taakvenster met stijlen weer te geven**

# Introduction

Population aging is no uncommon topic. Due to the increasing life expectancy, the older generation has been growing faster than the younger population. Research papers project that this gap will continue to grow in the future. Many researchers investigate the role of health for this aging population.

At the moment, many tests exist for older adults to detect common illnesses such as dementia and frailty. However few of these tests can be carried out from home. Having to do these test at a hospital, away from the patients comfort zone can influence the results of the test. It could be useful to design applications which can capture valuable information for research purposes or as a supplement for a doctor visit.

In order to get information about the cognitive performance of older adults from within their home, game analytics comes to mind. Game analytics is an overarching term for the capture, analysis and interpretation of data from video games. By examining the interactions of a player, useful information about his or her cognitive performance could be discovered.

For our thesis, Solitaire (Klondike) will be used to capture, analyse and interpret data from the player. Solitaire, otherwise known as patience, is a well-known memory card game that is available on every computer. In the game cards can be stacked in alternating colours from king to ace on the lower playing board to get the right order for the 4 final stacks. By stacking the cards well, more cards get available in the playing field. The goal of the game is to stack the cards from the playing board to 4 stacks of cards sequenced from ace to king in their corresponding suit (clubs, hearts, diamonds and spades).

Currently, it isn’t possible to get information from the game because the source code isn’t accessible. We are trying to tackle this problem by analysing the state of the game using visual processing techniques. There are many open source libraries available for image processing. Examples being OpenCV, BoofCV, SimpleCV and many others. We decided on using OpenCV (Open Source Computer Vision Library) in combination with the C++ programming language. It’s one of the most popular computer vision libraries, it’s actively maintained and has extensive documentation.

## Related work

## Requirements

The goal of this thesis is to design a general shell over the memory card game Solitaire. This shell should be able to capture data while the patient can play the game from the comfort of his home. Although the game shell will be designed for Solitaire, it should also be adaptable to other card games.

There are four high-level requirements. The first requirement is the ability to map the board states of card games through image recognition. Each card in the board has to be extracted from the image and classified correctly. When the player makes a move, the application should register this move and process the new board state. A move is defined as the movement of a card to a new location.

The second requirement is the ability to add game logic such as game rules and states. An example being the undo button. When this button is pressed, the game should go to the previous game state instead of processing it as a new move. Another example is the hint button, which is accessed through clicking menu, followed by hint. The application should be able to handle the popup of the menu.

Another requirement is the ability to capture and interpret game data. In game analytics terms, this game data is called game metrics. These metrics are interpretable measures of things related to the game. For Solitaire, the following metrics should be captured: the total playing time, is the game solved, number of clicks on the pile/ talon/ build stack/ suit stack, number of undo’s, number of hints requested, number of colour errors (red on red, black on black), number errors, average time between moves, the score and beta errors. Beta errors are defined as the errors when a player clicks through the pile while there are still moves available on the board.

The final requirement is the ability to store all the data in a database for further analysis.

# Design

The design of the game shell consists of 3 major parts. The first part is the extracting of the cards from the image of the playing board. The second part is the classification of each card. The final part is the game logic. The design of each part will be discussed thoroughly below, followed by a general overview of the game shell.

## Card extraction

### Introduction

In order to map the state of the board, the cards of should first be extracted from an image of the playing board. Once the cards are extracted, they can be classified and processed later in the application. By implementing the game logic well, only the top cards have to be extracting from the board. We define top cards as the highest card on a location of stacked cards, as indicated in figure 2.1.



Figure 2.1 Card of interest in a stack of cards, namingly the top card

The game logic has to keep track of the cards that are underneath this top card. If for example the game starts with a visible eight of harts, and the players moves the seven on clubs on the eight of hearts, the game logic should keep a list that consists of both cards. The extraction should also keep track of the location of each card.

### Methods

To extract the cards from an image of the playing board, image processing techniques are required. OpenCV has a large module named “imgproc” which contains a lot of functions that can be used to extract the regions of interest. Before any processing takes place, the image should always be resized to a standard size to ensure consistent processing.

#### Extracting the cards

A useful function to extract cards is the openCV function “findContours”. The function requires a binary image and searches contours in the image, as white objects in a black background. By converting the image to grayscale and applying a threshold, a binary image can be created. Thresholding an image is the act of setting all pixels below a certain gray level to black and all pixels above that gray level to white, resulting in an image with only black and white colours.



Figure 2.2 Image of a basic playing board of Solitaire

Because cards are mostly white, or at least always have a white border, “findContours” should be able to detect the border of the cards. The cards aren’t the only white objects in the image. As visible in figure 2.2, there are also small white objects visible in the top bar and the bottom corner buttons. These can be removed by calculating the area of each contour and removing all contours that are too small. Another option is making sure that the unnecessary information isn’t processed in the first place. By extracting the general area where cards occur, before processing and finding contours, only useful information will remain. This second method is the better solution. Removing unnecessary information before doing extensive processing increases the efficiency of the application.

This general area can be calculated one time using the first method (finding all contours and keeping only the large ones). By calculating the rectangle that encloses all cards, the general area is found and can be used for all following extractions. Because stacked cards take more space towards the bottom of the screen, a large margin should be used at the bottom of the image.

#### Extracting the top card

After the contours have been found, the top card still needs to be extracted. As visible on figure 2.1, the three stacked cards are all white and will be seen as one contour. There are three methods to extract the top card from such an image. The first option is using the fixed size of a playing card. The height of a playing card is always 1.33 times bigger than the width. By using the width of a stacked card and calculating the card height from the bottom towards the top, the lowest card can be found. The same principle applies to the talon cards which are stacked sideways. Using the height and calculating the corresponding width from the right, the top card can be extracted.



Figure 2.3 Talon cards that are stacked sideways

Another method is the use of Sobel edge detection. There are small gray lines visible that indicate the contour of a single card. Efficient processing is required to find only the small gray lines and not other potential lines in an image. This is possible by inserting certain requirements for lines. The lines should always be perfectly horizontal (or vertical for the talon cards). They should also be almost as long as the width (or height) of the image.

The final method comes forth from thresholding. The line is small, but visibly gray. By applying a stronger threshold, the gray line should become black giving clear boundaries between cards. The largest remaining contour will be the card, the smaller contours will be the cut off part of cards.

#### Keeping track of the location of each extracted card

Klondike Solitaire always uses the same layout. The pile and talon are on the top left, the suit stack is on the top right and the build stack is at the bottom. The easiest way to keep track of the location of each extracted card, is by extracting each card region from the general region of cards that was described in 2.1.2.1. When the general region gets calculated using the contours of all cards, the height of the cards at the top can also be calculated. By using that height, the cards at the top can be separated from the cards at the bottom.

The cards at the bottom are all spaced equally over the width. The card regions at the bottom can be extracted by splitting the image in 7 equal parts. The cards at the top are spaced a little different as visible from the board image of figure 2.2. The figure can be divided again in 7 equal parts. The talon takes part two and three. The suit stack takes the last 4 parts.

The card regions are extracted one by one, next the cards are extracted, finally the top card is extracted.

#### Test method

To test the efficiency and robustness of the card extraction, the speed of implementations should be measured and the amount of faulty extractions. We do this by taking 10 screenshots of playing boards where a random amount of moves were played. There need to be some stacked cards in each playing board to test the correct extraction of the top card.

The test method will try to extract all top cards from each playing board, and repeat this action 100 times. The total duration of the card extraction of all cards in one board is the average duration of one loop of one playing board.

To test the correct extraction, the cards are manually extracted by us from the board. These manually extracted cards are compared to the cards extracted by the application. The difference between images gives the error rate of the card extraction.

### Results

TODO: manually extract the top cards from gameAnalysis\GameAnalytics\test\someMovesPlayed. In GameAnalytics::test, extract the cards using the application and compare them with the manually extracted cards (saved in a vector) and test the different methods:

* Standard size of cards
* Strong threshold
* Sobel edge (lower the threshold so that sobel edge is used)

Time each test function at least 3 times and calculate the error margin (average amount of nonzero pixels)

### Discussion

## Card classification

### Introduction

The classification of cards consist of two parts. First, the image of the rank and suit should be segmented from the image of the card. Next the images should be classified to the correct rank and suit. For easy of use, the ranks and suits can be represented by a single char. The ranks are represented as 2, 3, 4, 5, 6, 7, 8, 9, :, J, Q, K, A. The suits are represented as C (for clubs), S (for spades), D (for diamonds) and H (for hearts).

### Methods

### Results

### Discussion

## Mouse clicks

### Introduction

### Methods

### Results

### Discussion

## General

# Development

# Discussion

References

Appendix

**faculteit INDUSTRIËLE INGENIEURSWETENSCHAPPEN**

CAMPUS GROEP T LEUVEN

Andreas Vesaliusstraat 13

3000 LEUVEN, België

tel. + 32 16 30 10 30

iiw.groept@kuleuven.be

[www.iiw.kuleuven.be](http://www.iiw.kuleuven.be)