The Interactive Visual Picross Solver (IVPS)

Ntsikelelo Sonjica

Thesis presented in fulfilment
of the requirements for the degree of
Honours in Computer Science
at the University of the Western Cape

Supervisor: Reg Dodds
Co-supervisor: Mehrdad Ghaziasgar

November 2016

Declaration

I, Ntsikelelo Sonjica, declare that this thesis "*The Interactive Visual Picross Solver (IVPS)*" is my own work, that it has not been submitted before for any degree or assessment at any other university, and that all the sources I have used or quoted have been indicated and acknowledged by means of complete references.

Signature: .... Date: ....

Ntsikelelo Sonjica.

Abstract

A picross puzzle is a game that takes the form of a R $\times$ C grid, with numbers placed on the left of its rows and on the top of its columns, which give the hints to solve the puzzle. These puzzles have been popular since the past years all over the world, and there are companies involved in the commercialization of products related to them, mainly magazines, newspapers, games for mobile phones and on-line puzzles that are found in the web(1). This project focuses mainly and only on the picross puzzles that are found in newspapers, magazines, i.e the picross puzzles that are hard-copied or drawn in papers. The aim is to use a visual and interactive approach to help the users(picross puzzle solvers) to better solve the picross puzzles that are found in papers. This project hypothesizes that the success of this project would be a solution to many problems that the users experience when they solve the puzzles, few may include: the minimization of the number of mistakes that one makes when solving the puzzles, to minimize the time that the user takes to solve the puzzles, a solution to the unavailability of the undo button when one has made a mistake, and the immediate availability of solutions of the puzzles to users. Also the technique can then be extended and applied to many puzzle games like Sudoku, tic-tac-toe and word-puzzle games that are found in papers. This would also eliminate the fact that it is only experienced players that solve puzzles in papers due to few hints given to players because it would also enable beginners/new players to easily learn how to play and solve the picross puzzles. In general this report hypothesizes that the picross solvers would learn and solve the picross puzzles found in papers better using the IVPS approach than using the traditional way to solve the puzzles. The remainder of this paper is divided into three chapters; the first chapter describes the User Requirements Document followed by the description of the Requirements Analysis Document and a concluding chapter.

Key words

Picross puzzles

IVPS

OpenCV

Computer Vision

Image processing

Machine learning

Magazines

Newspapers

Hard-copied or drawn in papers

Acknowledgment

This project is a compilation of the efforts of many people that helped me through the year. I would first like to thank my supervisor and co-supervisor, respectively Mr. Reg Dodds and Mr. Mehrad Gaziasgar, for encouraging me during my study. Without our weekly meetings, this work would not have been possible. At this time I would like to extend a very special thanks to God. Without his grace I would certainly not be where I am today. I would also like to thank TFG for their unwavering financial assistance without which my efforts would have been impossible.

Contents

Declaration iii
Abstract v
Key words vii
Acknowledgment ix
List of Tables xiii
List of Figures xv
Glossary xvii
1. User Requirements Document 1
1.1 Introduction 1
1.2 Users View of the Problem 1
1.3 Description of the Problem 2
1.4 Expectations from the Solution 3
1.5 Not Expected from the Solution 3
2. Requirement Anaylsis Document 5
2.1 Introduction 5
2.2 Designers Interpretation of the User Requirements 5
2.3 Related Work 6
2.3.1 Automatic Puzzle solving with Image Processing by (5) 6
2.3.2 Visual Chess Recognition by (6) 6
2.3.3 Sudoku Solver by (7) 8
2.4 Link these Solutions to the Problem 8
3. Conclusion 11
Bibliography 12

List of Tables

List of Figures

1.1 IVPS system 2

Glossary

URD User Requirements Document.

RAD Requirements Analysis Document.

IVPS Interactive Visual Picross Solver: is the interactive approach to solve picross puzzles that are hard-copied or drawn in papers.

OpenCV Open Source Computer Vision: is a library of programming functions mainly aimed at real-time computer vision.

Chapter 1

User Requirements Document

1.1 Introduction

The User Requirements Document (URD) is a document that is mostly used in software engineering, it describes the requirements that the user expects from the software which is to be developed. The URD gives a brief description of the problem from the user's point of view and desired results. The URD section is divided in to 5 subsections, namely, the Introduction, Users View of the Problem, Description of the Problem, Expectations from the Solution and finally, Not Expected from the Solution, the subsections are organised in that order.

1.2 Users View of the Problem

The user requires an interactive picross solving approach that gives him/her enough hints and instructions or warnings to play and solve a picross puzzle found in a paper.

The user requires an interactive picross solving approach that helps him/her to improve his/her solving skills and easily practice more solving strategies.

The user requires an interactive picross solving approach that minimizes the time it takes him/her to solve the picross puzzles.

The user requires to get a solution of the picross puzzle immediately after playing.

The user (inexperienced) requires to learn how to play and solve the picross puzzles easy without the help of a third party(another user).

The proposed or required building blocks for the success of this project are:

- user
- Web-camera
- Paper with the picross puzzle (hard-copied or drawn)
- Computer
- Screen

Below is a figure showing how the above proposed requirements for the success of this project are expected to work. The system is well explained in chapter 3.

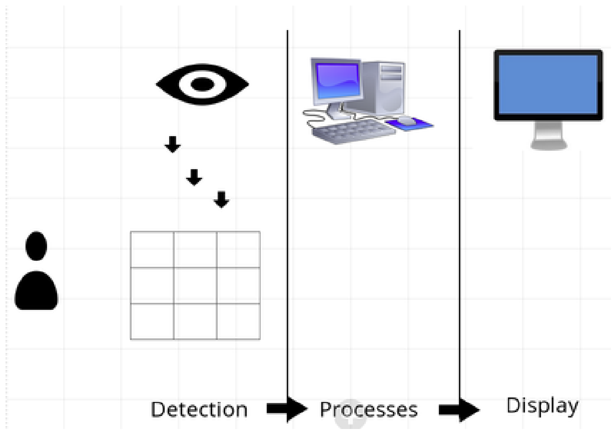

Figure 1.1: IVPS system

1.3 Description of the Problem

When playing or solving a picross puzzle that is found either in magazines, newspapers or picross puzzles that are hard-copied or drawn in papers, it becomes very slow and difficult to solve the puzzle due to the absence of special effects and due to limited hints, instructions or warnings given to the user, and sometimes users end-up giving up easily in solving these puzzles due to the number of mistakes they make and end-up being forced to draw the puzzles over and over again until they solve these puzzles. That process of drawing these puzzles over and over is very tiring and tedious. A system needs to be developed to help a user to solve the picross puzzles better, or to work to minimize or eliminate the number of mistakes that a user makes when he/she solves these picross puzzles.

1.4 Expectations from the Solution

The system needs to successfully help the user to play and solve a hard-copied or drawn-in-paper picross puzzle. Listed below is what is expected:

- A user has to be able to draw or at least find a paper that has a picross puzzle.
- A camera has to be pointed at the paper with a picross puzzle and has to be able to detect the picross grid/puzzle and stores it virtually in a computer
- A computer has to analyze the image and confirm that it is a picross grid and apply OpenCV techniques (image processing) to the image.
- A screen has to output hints to the user, warn the user when he makes mistakes, give the solution of the puzzle to the user and more guidelines to the user.

1.5 Not Expected from the Solution

Solving any puzzle games found in paper other than the picross puzzles is not part of the project scope. The user does not expect the system to be difficult to use and even though the will be time delay the process of solving the picross puzzles using IVPS should not be slower than the traditional way solving process. The system should not be user unfriendly.

Chapter 2

Requirement Anaylsis Document

2.1 Introduction

Requirement Analysis is the process of determining user expectations for a new or modified product. The Requirement Analysis Document (RAD) is then the document that takes the URD as a starting point and looks at the problem from a designer's point of view. This means that the requirements identified in the first chapter (URD) are analysed in this chapter and the problem is looked from the designer's point of view, hence the RAD focuses on the system and software requirements. The sections of this chapter are arranged as follows, first is the introductory section, followed by the designers interpretation of the user requirements then we look at the related work done i.e we identify existing solutions, we then link these solutions to this project problem and finally devise ways to test the solution. All these sections are integrated to complete the process of analysing the requirements stated in the first chapter.

2.2 Designers Interpretation of the User Requirements

The IVPS system is interpreted by the user to work following these steps: picross grid detection (using OpenCV techniques, i.e corner detection, edge detection or line detection), picross-grid numbers detection and recognition (using OpenCV image processing techniques, i.e hand-writing detection or digits detection), and determination of square occupancy. More OpenCV image processing techniques will be used, such involves conversion of a color image into gray-scale image; binarization ,that means converting the grayscale image into a binary image using a method known as adaptive thresholding; then the use of morphological operations to get rid of noise in the image.

2.3 Related Work

The following subsections are the different existing solutions that are related to the work needed to be done for the IVPS. Please note that these solutions are briefly described, however to get the full paper(article) corresponding to each solution, is found in the bibliography.

2.3.1 Automatic Puzzle solving with Image Processing by (5)

Their goal was to solve a puzzle using high quality reference image (or images) and digitally reassemble the puzzle, creating an image of the complete puzzle. They also hoped to go the reverse direction; if a user takes the picture of a piece, they hoped to be able to use the various image algorithms to detect where in the puzzle might go. Their methodology is given in the following points.

1) Take high quality reference image or images of puzzle pieces
2) Use an algorithm (maybe a combination of SURF, SIFT or other local feature detector) in order to align the images
3) Find different combinations to minimize the amount of error (space) between pieces.
4) Keep matching pieces until all the pieces are matching.
5) Hopefully the end image is rectangular and normal looking.

These are the resources they needed to complete their project, they needed puzzles, OpenCV, and Matlab. Their intentions were to do the initial part of their project on desktop computers with high quality images, and for the later parts of their project, they needed an android/iOS device upon which they do the testing.

2.3.2 Visual Chess Recognition by (6)

Their goal was to correctly detect and identify a chessboard and the configuration of its pieces through the application of image processing techniques. Their idea was that such an algorithm could be used to automatically record a game between two players without the need for a digital chess set, which can be costly. In addition, they proposed that image-based detection of chess pieces is a vital step in building chess-playing robots, as the playing strategy of the robot depends on its knowing the locations of its own chess pieces and the pieces of its opponent. Hence chess-playing robots can be used for fun and have furthermore been considered as an interactive toy that helps in developing the learning abilities of children.

At a high level, their algorithm performed the following steps: chessboard detection and segmentation, determination of square occupancy, and recognition of the chess pieces. The following list discuss these steps in further detail and were followed by a discussion of elements that they believed made the problem easier or more difficult. Implementation was performed in MATLAB and OpenCV using still images from a digital camera.

1) The first step was to detect the square pattern of the chessboard and identify the individual squares. The detection techniques was generally categorized as corner detection and edge detection.
2) After the individual squares was identified, each square was then evaluated to determine if it was occupied or not, and, for squares that were occupied, the color of the piece that is present. This was performed using a comparison to a reference image of an empty board in conjunction with gray level statistics for the potential occupancy region of each square.
3) Finally, they used feature recognition to classify the chess pieces into one of six types: pawn, knight, bishop, rook, queen, or king. They assumed no prior knowledge of the locations of chess pieces, and the robot had to detect it by itself from the captured image. Training images were captured from different angles for each of the six types of pieces, with an anticipated fewer number of images needed for pieces with azimuthal symmetry.

They encountered many complicating factors in the use of an image-based chess recognition product: viewing the board at an extreme angle, non-uniform lighting conditions, damage to the board or pieces, and extraneous objects in the scene. Some other inherent challenges were edges or corners being occluded by pieces and the standard use of the same two colors for squares and pieces, however with all that being said their project was done successfully and the supporting paper is found in (6).

2.3.3 Sudoku Solver by (7)

The goal of their project was to create an Android app that could automatically solve a Sudoku puzzle from images taken by the phone camera, either in real-time or close to real-time. Ideally, the user had simply needed to aim the camera at the Sudoku puzzle, and the app would automatically detect the puzzle, solve it, and overlay the solution on top of the camera image.

The steps of their algorithm were the following. First, the image is binarized using some method of locally adaptive thresholding for documents. Then, noise had to be reduced using morphological opening. The orientation of the Sudoku puzzle was estimated using the Hough transform. The existence of 10 vertical lines and 10 horizontal lines was used to help verify that a grid was indeed a Sudoku puzzle. The Sudoku puzzle was then perspective-corrected so that its grid lines were axis-aligned. The given digits in the grid cells were extracted and recognized using template matching, and methods that handled hand-written digits. Once the digits were recognized, the puzzle was solved using a Sudoku solving algorithm. Finally, the results were overlaid onto the screen image by adding perspective-transformed images of digits into the empty grid cells of the puzzle. The full paper to their project is found at (7).

2.4 Link these Solutions to the Problem

All the different techniques used for the above mentioned related work (projects) will be linked and applied in the IVPS. This means the projects are all together best solutions when integrated as they yield similar output, generally to detect a puzzle and give a solution to the user.

Chapter 3

Conclusion

The proposed system is an addition to the traditional approach of solving picross puzzles, i.e the idea is to stick with the ”pen/pencil and paper” process that the user is familiar with. The system aims to: minimize the time that a user takes and the number of mistakes that a user makes when she/he solves picross puzzles by giving the user hints and warnigs to solve picross puzzles. The provision of more hints and warnings also allows anyone to learn and play these puzzles. Another aim is to give a picross puzzle solution immediately to a user when she/he requires it. Here we introduce a camera, a computer and a screen. The camera captures the picross grid found on a paper, stores it on a computer that does many processes. The processes include computer vision computations using an open source computer vision library(openCV) to analyze the picross grid recorded as frames, computations that use python algorithm to solve the picross grid and computations to provide hints or warnings to the user. All those are then displayed on a screen to be visible to the user.

Bibliography

[1] E.G Ortiz-Garcia, S. Salcedo-Sanz, J.M Leiva-Murillo, A.M Perez-Bellido and J.A Portilla-Figueras, Automated generation and visualization of picture-logic puzzles, *Department of Signal Theory and Communications Universidad de Alcala and Carlos III de Madrid*, Madrid, 2007.

[2] H.S Hsiao and J.C Chen, Using a gesture interactive game-based learning approach to improve preschool children's learning performance and motor skills, *National Taiwan Normal University, No.162, Sec. 1, Heping E. Rd., Da'an Dist., Taipei City 106* ,Taiwan, 2016.

[3] P. Li and J. Connan, Numberplate Detection Using Double Segmentation, *Department of Computer Science, University of the Western Cape, Private Bag X17 Bellville, 7535*, South Africa, 2009.

[4] F. Dandurand, D. Cousineau and T.R Schultz, Solving nonogram puzzles by reinforcement learning, *Department of Psychology, Universite de Montreal, Ecole de psychologie, Pavillon Vanier, Universite d'Ottawa, Department of Psychology and School of Computer Science, McGill University*, Canada, 2013.

[5] A. Cousland, C. Ho and J. Nakamura, Automatic Silhouette-Based Puzzle Assembly, *How We Out-Puzzled Puzzling*, Electrical Engineering, Stanford University, Stanford, California, 2015.

[6] C. Danner and M. Kafafy, Visual Chess Recognition, *cdanner@stanford.edu, mkafafy@stanford.edu*, Stanford University, Stanford, 2015.

[7] Y. Wang, Sudoku Solver, *wangyix@stanford.edu*, Stanford University, Stanford, 2015.