A FULLY-AUTOMATED SOLVER FOR MULTIPLE SQUARE JIGSAW PUZZLES USING HIERARCHICAL CLUSTERING

A Thesis

Presented to

The Faculty of the Department of Computer Science

San Jose Staté University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Zayd Hammoudeh

December 2016

The Designated Thesis Committee Approves the Thesis Titled

A FULLY-AUTOMATED SOLVER FOR MULTIPLE SQUARE JIGSAW PUZZLES USING HIERARCHICAL CLUSTERING

by

Zayd Hammoudeh

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

December 2016

Dr. Chris Pollett	Department of Computer Science
Dr. Thomas Austin	Department of Computer Science
Dr. Teng Moh	Department of Computer Science

ABSTRACT

A Fully-Automated Solver for Multiple Square Jigsaw Puzzles Using Hierarchical Clustering

by Zayd Hammoudeh

This paper is very abstract.

DEDICATION

To my mother.

TABLE OF CONTENTS

CHAPTER

CHAPTER 1

Introduction

Jigsaw puzzles were first introduced in the 1760s when they were made from wood; their name derives from the jigsaws that were used to carve the wooden pieces. The 1930s saw the introduction of the modern jigsaw puzzle where an image was printed on a cardboard sheet that was cut into a set of interlocking pieces [?, ?]. Although jigsaw puzzles had been solved by children for two centuries, it was not until 1964 that the first automated jigsaw puzzle solver was proposed by Freeman & Gardner [?]. While an automated jigsaw puzzle solver may seem trivial, the problem has been shown by Altman [?] and Demaine & Demaine [?] to be strongly NP-complete when pairwise compatibility between pieces is not a reliable metric for determining adjacency.

Jig swap puzzles are a specific type of jigsaw puzzle where all pieces are equally sized, non-overlapping squares. Jig swap puzzles are substantially more challenging to solve since piece shape cannot be considered when determining affinity between pieces. Rather, only the image information on each individual piece is used when solving the puzzle.

Solving a jigsaw puzzle simplifies to reconstructing an object from a set of component pieces. As such, techniques developed for jigsaw puzzles can be generalized to many practical problems. Examples where jigsaw puzzle solving strategies have been used include: reassembly of archaeological artifacts [?, ?], forensic analysis of deleted files [?], image editing [?], reconstruction of shredded documents [?], DNA fragment reassembly [?], and speech descrambling [?]. In most

of these practical applications, the original, also known as ‘‘ground-truth,’’ input is unknown. This significantly increases the difficulty of the problem as the structure of the complete solution must be determined solely from the bag of component pieces.

This thesis outlines the first fully-automated jigsaw puzzle solver algorithm for multiple puzzles; unlike all previous solvers, this thesis' implementation is provided no no makes no assumptions regarding the set of input pieces, including the number of ground-truth (i.e., original) puzzles. What is more, it defines a set of new metrics specifically tailored to quantify the quality of outputs of multi-puzzle solvers.

CHAPTER 2

Previous Work

Computational jigsaw puzzle solvers have been studied since the 1960s when Freeman & Gardner proposed a solver that relied only on piece shape and could puzzles with up to nine pieces [**?**]. Since then, the focus of research has gradually shifted from traditional jigsaw puzzles to jig swap puzzles.

Cho *et al.* [**?**] proposed in 2010 one of the first modern computational jig swap puzzle solvers; their approach relied on a graphical model built around a set of one or more ‘‘anchor piece(s),’’ which are pieces whose position is fixed in the correct location before the solver began. Cho *et al.*’s solver required that the user specify the puzzle’s actual dimensions. Future solvers would improve on Cho *et al.*’s results while simultaneously reducing the amount of information (beyond the set of pieces) passed to the solver.

A significant contribution of Cho *et al.* is that they were first to use the LAB (Lightness and the A/B opponent color dimensions) colorspace to encode image pixels. LAB was selected due to its property of normalizing the lightness and color variation across all three pixel dimensions. Cho *et al.* also proposed a measure for quantifying the pairwise distance between two puzzle pieces that became the basis of most of the future work (see Section **??**).

Pomeranz *et al.* [**?**] proposed an iterative, greedy jig swap puzzle solver in 2011. Their solver did not rely on anchor pieces, and the only information passed to the solver were the pieces, their orientation, and the size of the puzzle. Pomeranz *et al.* also generalized and improved on Cho *et al.*’s piece pairwise distance measure by

proposing a ‘‘predictive distance measure.’’ Finally, Pomeranz *et al.* introduced the concept of ‘‘best buddies,’’ which are any two pieces that are more similar to each other than they are to any other piece. Best buddies have served as both an estimation metric for the quality of solver result as well as the foundation of some solvers' placers [**?**].

An additional key contribution of Pomeranz *et al.* is the creation of three image benchmarks. The first benchmark is comprised of twenty 805 piece images; the sizes of the images in the second and third benchmarks are 2,360 and 3,300 pieces respectively.

In 2012, Gallagher [**?**] formally categorized jig swap puzzles into four primary types. The following is Gallagher's proposed terminology; his nomenclature is used throughout this thesis.

- **Type 1 Puzzle**: The dimensions of the puzzle (i.e., the width and height of the ground-truth image in number of pixels) is known. The orientation of each piece is also known, which means that there are exactly four pairwise relationships between any two pieces. A single anchor piece, with a known, correct, location is required with additional anchor pieces being optional. This type of puzzle is used by [**?**, **?**].

- **Type 2 Puzzle**: This is an extension of a Type 1 puzzle, where pieces may be rotated in 90° increments (e.g., 0°, 90°, 180°, or 270°); in comparison to a Type 1 puzzle, this change alone increases the number of possible solutions by a factor of 4^n, where n is the number of puzzle pieces. What is more, no piece locations are known in advance; this change eliminates the use of anchor piece(s). Lastly, the dimensions of the ground-truth image may be unknown.

- **Type 3 Puzzle**: All puzzle piece locations are known and only the rotation of the pieces is unknown. This is the least computationally complex of the puzzle variants and is generally considered the least interesting. Type 3 puzzles are not explored as part of this thesis.

- **Mixed-Bag Puzzles**: The input set of pieces are from multiple puzzles, or there are extra pieces in the input set that belong to no puzzle. The solver may output either a single, merged puzzle, or it may separate the input pieces into disjoint sets that ideally align the set of ground-truth puzzles. This type of puzzle is the primary focus of this thesis.

Sholomon *et al.* [**?**] in 2013 proposed a genetic algorithm based solver for Type 1 puzzles. By moving away from the greedy approach used by Pomeranz *et al.*, Sholomon *et al.*'s approach is more immune to suboptimal decisions early in the placement process. Sholomon *et al.*'s algorithm is able to solve puzzles of significantly larger size than previous techniques (e.g., greater than 23,000 pieces). What is more, Sholomon *et al.* defined three new large image (e.g., 5,015, 10,375, and 22,834 piece) benchmarks [**?**].

Paikin & Tal [**?**] published in 2015 a greedy solver that handles both Type 1 and Type 2 puzzles, even if those puzzles are missing pieces. What is more, their algorithm is one of the first to support solving Mixed-Bag Puzzles. Paikin & Tal's algorithm is used as the basis for much of this thesis and is discussed in significant depth in Section **??**.

CHAPTER 3

The Files

For this thesis format, you LaTeX the file `thesis.tex`. But first, you need to make some modifications to `thesis.tex`. The title of your report, committee members, etc., are specified in `thesis.tex`. All of the things that you need to modify are indicated by comments beginning with five consecutive asterisks, so search for "*****" in `thesis.tex` and make the necessary changes.

You put the actual content of your report in the following files:

- `abs.tex` --- abstract
- `ack.tex` --- acknowledgements
- `chap1.tex`, `chap2.tex`, and so on --- chapters
- `bib.tex` --- references
- `appA.tex`, `appB.tex`, and so on --- appendices (if any)