A Hybrid Approach to Domino Portrait Generation

Hadrien Cambazard and **John Horan** and **Eoin O'Mahony** and **Barry O'Sullivan**
Cork Constraint Computation Centre, Department of Computer Science, University College Cork, Ireland
{h.cambazard|j.horan|e.omahony|b.osullivan}@4c.ucc.ie

Abstract

A domino portrait is an approximation of an image using a given number of sets of dominoes. This problem was first stated in 1981. Domino portraits have been generated most often using integer linear programming techniques that provide optimal solutions, but these can be slow and do not scale well. We demonstrate a new approach that overcomes these limitations and provides high quality portraits. Our software combines techniques from operations research, artificial intelligence, and computer vision. Starting from a randomly generated template of blank domino shapes, a subsequent optimal placement of dominoes can be achieved in constant time when the problem is viewed as a minimum cost flow. The domino portraits one obtains are good, but not as visually attractive as optimal ones. Combining techniques from computer vision and local search we can improve our portraits to be visually indistinguishable from those generated optimally.

Introduction

In 1981 Kenneth Knowlton filed for a United States Patent entitled "Representation of Designs" in which he proposed the use of dominoes to render monochrome images (Knowlton 1981). A domino portrait is simply a rendering of an image using a given number of sets of dominoes. Generally he uses "double nine" domino sets, which contain all dominoes from the "double blank" to the "double nine", giving fifty five dominoes in all. The nice property of "double nine" domino sets is that they give a wide range of shades from complete black (the blank domino) to a bright white (the double nine domino). A set of dominoes gives us a constrained palette of monochrome shades, which we can use to produce images. We say that the palette is constrained for two reasons. Firstly, each set contains only one domino of each type. Secondly, we are not allowed to break dominoes into two parts, but rather use the entire domino. Several examples of domino portraits based on a well known portrait of George Boole are presented in Figure 1.

A problem with current approaches to generating domino portraits is that they do not scale very well. This is mostly due to the fact that researchers have focused on finding optimal portraits. We set out to develop a scalable approach that is not concerned with whether the portraits found are optimal or not, but is concerned with whether the portraits are sufficiently good so as to be visually indistinguishable from optimal ones (Cambazard et al. 2008).

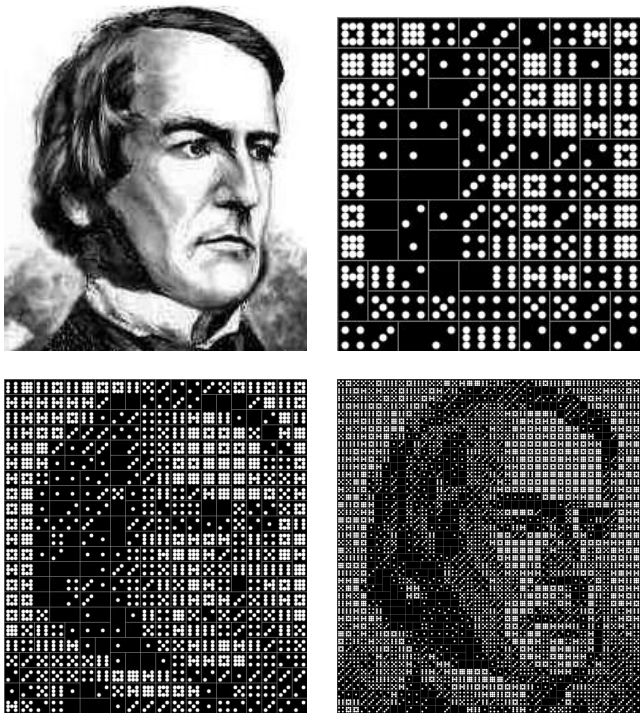


Figure 1: A well known portrait of George Boole is presented on the left, with a sequence of domino portraits generated from this image using 1, 4 and 16 sets of dominoes.

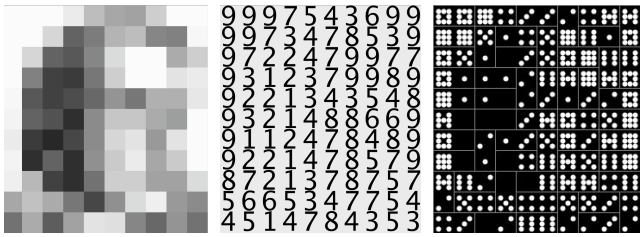
The Domino Portrait Generation Problem

A domino portrait can be generated for any target image. The first step in the process is to convert the target image into a grayscale graphic image using, for example, the UNIX pgm command. Each pixel in a grayscale image is given a grayscale value between 0 (black) and 255 (white).

We consider rendering images using sets of "double nine" dominoes. There are 55 dominoes in a complete set: 10

with equal face values in both halves, i.e. all dominoes with face valuations equal to $(0,0),\ldots,(9,9)$ along with an additional 45 non-equal face dominoes with face values in $\{(v_1,v_2)|v_1 \in \{0,\ldots,8\}, v_2 \in \{v_1+1,\ldots,9\}\}$. The surface covered by a single set of dominoes is 110 square units, since we have 55 dominoes each with 2 units. Therefore, given s^2 sets of dominoes, the grayscale image can be divided into $11s \times 10s$ cells and for each cell in row r_i and column c_i the mean grayscale value is computed and scaled to an integer between 0 and 9 called g_{ij}. The value in each cell specifies the perfect half domino to place in that cell.

Each domino with equal valued halves has two possible orientations, vertical and horizontal, whereas each non-equal valued domino has four orientations since such a domino can be flipped along its vertical and horizontal axes. While for $k = s^2$ sets we could use a canvas of size $11s \times 10s$ to be filled with the $55 \times k$ dominoes, in practice we can represent any canvas of size $110 \times k$ cells. The cost of positioning a half-domino p_l^q on a cell (r_i, c_j) is equal to $(p_l^q - g_{ij})^2$. Notice that it is quadratic so that the cost grows faster than the error and large errors are strongly penalised. The problem is to place the dominoes on the canvas so that the overall cost (the sum of the costs of each cell of the canvas) is minimised and every domino is used exactly once. A graphical representation of the process is presented in Figure 2.



(a) The image is divided into cells and the grayscale values averaged in each.

(b) The averaged grayscale values are scaled to the range $0 \ldots 9$.

(c) An example of a domino portrait.

Figure 2: A summary of the process of generating a domino portrait from an image.

Approach

Robert Bosch proposed an integer linear programming formulation of the domino portrait generation problem in (Bosch 2004), but the resulting integer programs are quite large, with more than one million decision variables and five thousand constraints for $k = 49$. We adopt an approach similar to Knowlton's (Knowlton 1981) and Knuth's (Knuth 1993), in which the image is divided up into blank domino outlines defining *a pattern*. Our approach relies on the observation that the problem becomes polynomial once the *pattern* is known, because it can be modeled as an assignment problem associating $55 \times k$ dominoes with $55 \times k$ empty outlines. This suggests that restricting ourselves to searching over alternative patterns is enough to generate optimal domino portraits. Rather than treating this problem as a traditional assignment problem, which can be solved using the Hungarian Method, we formulate it as a *minimum cost flow*. The flow formulation allows us to solve the assignment problem in constant time by breaking many symmetries. These symmetries arise from the fact that each domino is repeated k times, and there are at most 55 possible pairs of costs for an empty outline. The outlines of identical cost can, therefore, be gathered into *areas*. As a result, we only need to know how many dominoes of each kind are assigned in each area, and not where each individual domino is placed. The flow can be interpreted as the number of dominoes of each kind assigned to each area. The size of the graph supporting this flow is independent of k making the approach robust to increases in the number of domino sets we use.

However, because we predetermine the orientations of the dominoes by choosing a pattern, we are unlikely to find an optimal domino configuration. Therefore, we adopt a heuristic approach based on *large neighborhood search* (Shaw 1998) to identify regions of the domino portrait that, if redesigned, would improve its quality. The pattern only matters where the grey values are unbalanced, since uniform areas have almost no effect on the final cost. We use an algorithm from computer vision that performs *corner detection,* or *interest point detection* (Rosten, Reitmayr, and Drummond 2005), to extract certain kinds of features to infer the contents of an image. This approach seems suitable for portraits as it highlights the important characteristics of the face (eyes, mouth, hair, etc.) which matter in the final domino portrait, and suggests where it might be worthwhile improving the pattern.

Several orders-of-magnitude in runtime are obtained by this approach over current methods. It does not provide optimal solutions but produces high quality portraits within seconds, while remaining robust to increases in the size of the problem.

Acknowledgements

This work was supported by Science Foundation Ireland (Grant 05/IN/I886). We thank Robert Bosch for providing the inspiration for tackling this problem and useful feedback.

References

Bosch, R. 2004. Constructing domino portraits. *Tribute to a Mathemagician* 251–256.

Cambazard, H.; Horan, J.; O'Mahony, E.; and O'Sullivan, B. 2008. Fast and scalable domino portrait generation. In *Proceedings of CP-AI-OR 2008*, 51–65. LNCS 5015.

Knowlton, K. C. 1981. Representation of designs. *U.S. Patent # 4,398,890 (Awarded August 16th, 1983)*.

Knuth, D. E. 1993. *The Stanford GraphBase: A Platform for Combinatorial Computing*. Addison-Wesley.

Rosten, E.; Reitmayr, G.; and Drummond, T. 2005. Real-time video annotations for augmented reality. In *ISVC*, 294–302.

Shaw, P. 1998. Using constraint programming and local search methods to solve vehicle routing problems. In *CP*, 417–431.