The Design and Evolution of Fiducials for the reacTIVision System

Ross Bencina and Martin Kaltenbrunner
Music Technology Group, Audiovisual Institute
Universitat Pompeu Fabra, Barcelona, Spain
{rbencina,mkalten}@iua.upf.es

Abstract

The reacTIVision system is software for tracking specially designed fiducials (markers) in a real-time video stream. ReacTIVision was designed to enable expressive gestural control of musical sound, and can track many markers at a high frame rate. The development of reacTIVision involved not only computer vision algorithms, but also the design of a new marker system. We co-designed the computer vision system and markers, applying evolutionary computation to minimise marker size while meeting geometric constraints required to efficiently compute the location and 2D orientation of the markers. The computer vision techniques adopted would not have been practical had the genetic algorithms not been able to solve these constraints. This paper focuses on the design and evolution of the markers, the computer vision aspects of reacTIVision are documented elsewhere.

1. Introduction

The reacTable* (shown in figure 1) is a tangible user interface where physical objects represent the components of a software sound synthesizer [1, 2]. Fiducials are attached to the underside of objects placed on a translucent table. A camera beneath the table captures images which are processed to determine the location, orientation and identity of the fiducials. This information is sent to other components of the reacTable* software via a network socket using the TUIO protocol [3], a protocol layered on top of Open Sound Control [4].

ReacTIVision is the system we developed for tracking the location and orientation of fiducials (markers) in a real-time video stream. The system was developed for the reacTable* after initial prototyping with Costanza and Robinson's d-touch system [5]. The technical motivations and evaluation of the reacTIVision system are discussed elsewhere [6], this paper is intended to present the techniques which were applied to design the fiducial markers. Although it is necessary to understand our methods in order to reproduce, improve or build upon our results, it should be noted that the reacTIVision markers which we have produced are available in PDF format with the complete reacTIVision system, and as such it is not necessary to produce new markers in order to use the system.

The reacTIVision fiducials began their life as variations of the d-touch fiducials, where identification is performed using topological pattern matching on a region adjacency graph of a binarised input image. Once fiducials have been identified, d-touch uses geometric techniques including line detection and relative position of regions to determine the location, orientation and identity of each marker (marker geometry decodes to a unique id number). Our design evolved toward using only the region adjacency graph and the bounding rectangles of each region to determine all necessary information about each marker. This allowed us to remove a number of steps from the computer vision algorithm, however it also introduced increased complexity in laying out the fiducials. To solve the layout constraints while minimizing the size of the fiducials we successfully employed a genetic algorithm.

The remainder of this paper is structured as follows: First we describe how the fiducials are identified, and how orientation and location information is encoded in their visual structure. Next, we describe the steps which were taken to generate the marker images: the generation of the binary trees which represent their topological structure, the method used to encode the layout of the fiducials, and the multi-pass rendering scheme used to create the final marker images. Next we describe the genetic algorithm we used to optimise the layout of the fiducials to minimise size while meeting the requirements of the tracking method, after which our distributed computation environment is described. Then an evaluation of the output of the genetic algorithm is presented. Finally, we discuss some applications of our system.

Figure 1: Musicians playing the reacTable* at Ars Electronica 2005. Objects on the table's surface are tagged with fiducials which are tracked by a camera beneath the table. Some projected graphics represent the telepresence of collaborating performers in Barcelona.

2. ReacTIVision Topological Fiducial Tracking

This section introduces the method used by reacTIVision to track fiducials in binary images. The method combines pattern matching on binary topological graphs for recognition and identification, with simple geometric techniques for computing the location and orientation of fiducials. The information is presented here to provide the reader with sufficient background to motivate the application of evolutionary computation to the generation of the fiducials. The relation of this method to other marker tracking work is discussed elsewhere. First the topological region adjacency graph is introduced. We then describe our approach for determining the location and orientation of fiducials.

2.1. Topological Recognition

ReacTIVision employs the topological fiducial recognition approach introduced by Costanza and Robinson in the d-touch system [5]. In this approach, a region adjacency graph is derived from a binary image of the scene through the process of *segmentation*. The graph can be understood as a tree representing the containership hierarchy of the image, that is, which black regions are contained inside which white regions and vice-versa.

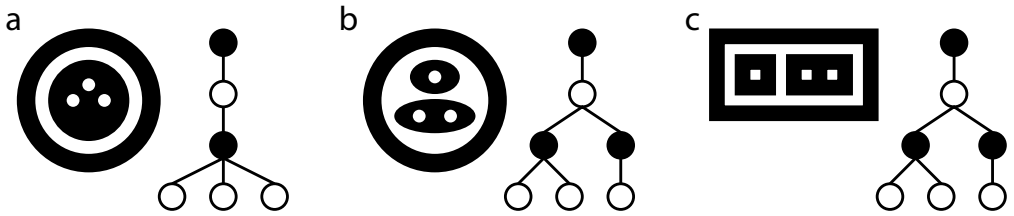

Figure 2: Some simple topologies and their corresponding region adjacency graphs.

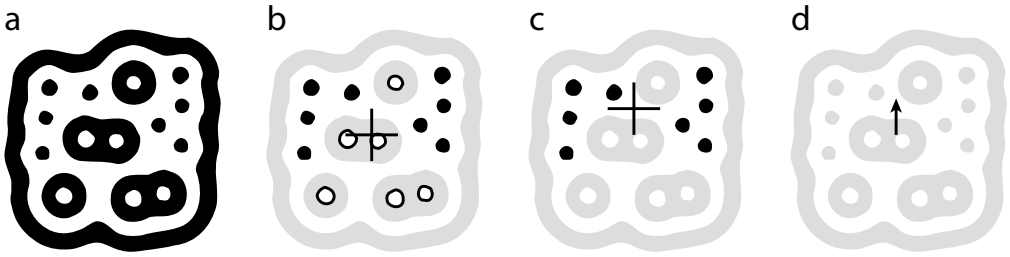

Figure 3: (a) a reacTIVision fiducial (b) black and white leafs and their average centroid (c) black leafs and their average centroid, and (d) the vector used to compute the orientation of the fiducial.

Regions of the image containing no other regions (unbroken blobs for example) appear as leaf nodes in the region adjacency graph. Figure 2 illustrates some simple images and their corresponding region adjacency graphs. Observe that figures 2b and 2c have identical region adjacency representations even though their geometries are quite different.

In contrast to the d-touch system which uses geometric properties to encode the identities of fiducials, reacTIVision fiducials are identified purely by their topological structure. Each fiducial in a set has a unique topology which can be efficiently matched against a dictionary of subtrees represented as strings [7, 6].

2.2. Fiducial Location and Orientation

Our method for computing fiducial location and orientation was influenced significantly by the design of our segmentation algorithm, which only retains axis aligned bounding boxes for each region. The center of a region's axis aligned bounding box provides a good approximation for the center of the region if the region is square, circular, and/or relatively small. We reasoned that as leaf regions will always be the smallest regions in a rendering of a tree, their centers are likely to be the most accurate spatial information we have about a fiducial. Consequently we choose to compute a fiducial's location and orientation as a combination of the bounding box centers of its leaf nodes.

As illustrated in figure 3, we compute the center point of the fiducial by taking a weighted average of all leaf centers. The vector from this centroid to a point given by the weighted average of all black (or white) leaf centers is used to compute the orientation of the fiducial. Each leaf is weighted by a function of its depth in the tree to account for the area consumed by its containing regions. We selected this method because it can be applied to any fiducial with at least one black and one white leaf region. Thus allowing us to vary the topological structure of fiducials without changing the method used to track them.

3. Generation

Generation of reacTIVision fiducials requires the selection of a set of unique topologies and graphical layout and rendering of these topologies. In the following subsection we discuss the generation of trees describing the topology of each fiducial; following which we give an overview of how we generate geometries which conform to the requirements of our location and orientation calculation method while minimising size. Finally we outline the method used to render the fiducials.

3.1. Fiducial Tree Generation

Before generating the geometry for a set of fiducials we generate a set of unique trees. Given a set size it is possible to calculate the number of tree nodes required to accommodate the set, however other constraints are also important such as ensuring a certain number of black and white leaf nodes. The number of nodes in a tree and its maximum depth also impacts the minimum size of the geometry which can be generated for a tree. Additionally, smaller trees are less unique, leading to a greater potential for encountering false detections when trying to recognise them in a scene.

Rather than enumerating all possible trees, we randomly generate trees with the desired number of nodes and select those which fulfill criteria including maximum depth and number of black and white leaves. The generation process slows when many of the candidates have been found in the search space. By observing this slowing we experimentally reduce the number of nodes to the minimum required to generate a set fulfilling our criteria.

3.2. Fiducial Geometry Generation

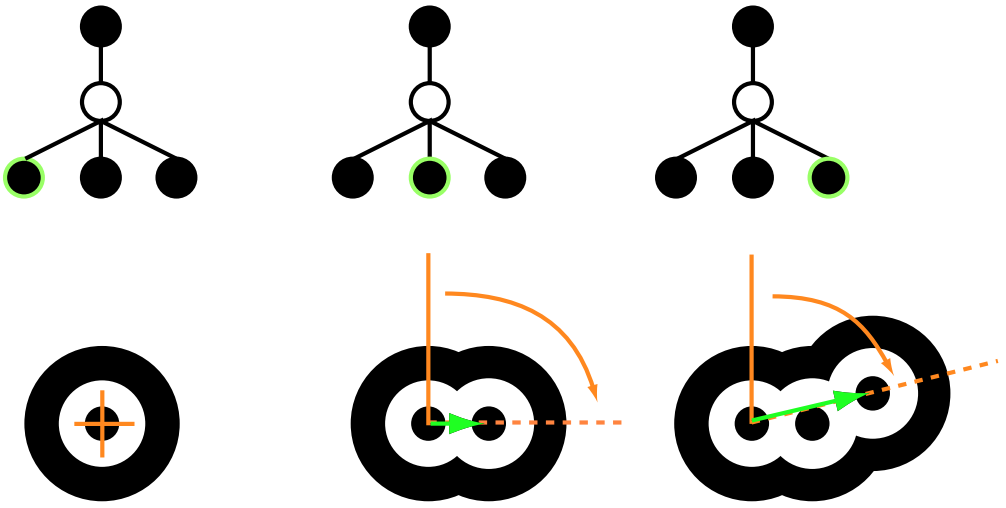

Figure 4: Leaf nodes are placed at different angles from a fixed center point, and moved outwards until they abut neighboring nodes.

Given a tree representing a fiducial's topology we create a compact geometry which conforms to the constraints implied by the location and orientation method described above: that the computed centroid of the fiducial is the same as, or lies very close to the real center of the fiducial and that the centroid of all black leaves is sufficiently distant from the centroid of all leaves to allow the fiducial's orientation to be computed with reasonable accuracy.

Each tree can be drawn in a huge number of ways making an exhaustive search for 'optimally' rendered fiducials impractical. We chose to employ a genetic algorithm (described later) to optimise parameters such as fiducial area, aspect ratio, symmetry and centroid locations for black and white leaves.

The method used to lay out a single fiducial involves aggregating circular leafs at angles relative to a fixed starting point while maintaining a fixed spacing between regions (see figure 4). A list of these angles forms the genotype for the genetic algorithm. Although earlier implementations used squares aligned to a grid, circles were chosen because they are easy to pack together at arbitrary angles, which we consider important to optimise the location of the centroids used to compute location and orientation. Circles also have the advantage that their bounding boxes are rotationally invariant, which simplified our machine vision implementation.

3.3. Rendering

The rendering process consists of drawing concentric circles of alternating colour for each leaf node, the number of concentric circles corresponds to the depth of each leaf node in the tree. All of the circles of the same depth are drawn at the same time, and built up in layers, with the leafs being drawn last. This method is not in itself sufficient to render fiducials with the correct topology because certain layouts can lead to miscoloured voids appearing within the fiducials. To avoid this problem we apply a median filter (erosion operator) to the rendered image after each depth layer is rendered. This also leads to a pleasing "rounded corners" aesthetic result.

4. Evolution

Following from the previous section, a reacTIVision marker is completely described by an angle-annotated tree and the colour of its root node. Since the structure of the trees are selected according to recognition criteria, the angles are the only parameters which are available for optimisation. We aimed to optimise the size and shape of the fiducials by using a genetic algorithm to evolve vectors of node angles meeting our constraints. The visual recognition mechanism used to determine the location and orientation of fiducials described in section 2.2 imposed additional constraints on our genetic algorithm, in that the leaf nodes needed to be placed so that the centroid of all leaf node locations was aligned with the center of the fiducial as a whole, and the centroid of all black leaf nodes was located some distance from the overall marker center. In the next subsection we describe the overall implementation of the genetic algorithm, which was a simple C++ program. Following that we describe the fitness functions used for two different classes of fiducials: square markers, and markers with a 5×9 aspect ratio. These functions were arrived at by trial and error using an interactive tool we developed.

4.1. Genetic algorithm

The genetic algorithm operates on pools of fixed length genotypes. Each genotype is a vector of signed 32 bit integers, each of which corresponds to a node angle used to generate the fiducial geometry. The full range of each integer is mapped to the angular range -π to π. Integers were used because modulo arithmetic comes for free (which is important for our angular computations), and integer random number generators are more common. For ease of implementation, a gene is allocated for each tree node, and non-leaf angles are ignored during rendering (i.e. "junk DNA"). Genomes are seeded with randomly distributed values. For each iteration the following steps are performed:

1. Remove duplicates from the pool.
2. Reduce the population to 40% of the original pool size retaining the best individuals, calling these "retained individuals".
3. Repeatedly apply genetic operators until the pool size returns to the original size.
4. Evaluate the fitness of each individual and sort by fitness.

The genetic operators and their associated probabilities of occurrence are: crossover (70%), major mutation (20%) and minor mutation (10%). Crossover is performed between any randomly selected individual and an individual randomly selected from the set of retained individuals. The crossover operation works by selecting a random gene index, splitting two genotypes at this index and splicing the first half of one genotype onto the second half of the other. Mutation is performed only on randomly selected new (non-retained) individuals. Major mutation selects a random number of genes (from one gene to 50% of the total genes in a genotype) and resets these genes to a random value. Minor mutation selects a random number of genes (from 2 to 50%) and increments or decrements their values by random numbers up to 50% of the total range, coresponding to the phenotype operation of rotating the angle of a node by a random amount up to 90^o in either direction.

In our computations we used pools of 500 individuals. We evaluated 20 pools for each fiducial and chose the best one. We ran approximately 3000 iterations of each pool, which from our observations appeared to produce convergence.

4.2. Fitness Functions

Although there were plans to produce fiducials in a number of different shapes, including circles and triangles, our efforts to date have only been successful in designing fitness functions for square and rectangular fiducials.

The function used for square fiducials is:

```
(black_symmetry + white_symmetry + (signed_black_centroid_distance_from_vertical_center /height))
* MAX(width,height) * width * height
+ (black_centroid_distance_x + 1)
+ (black_distances_below_y + 1)
```

And for rectangular fiducials:

```
(black_symmetry + white_symmetry + (signed_black_centroid_distance_from_vertical_center/height) * .5)
* MAX((width * (9. / 5.)),height) * (width * (9. / 5.)) * height
+ (black_centroid_distance_x + 1)
+ (black_distances_below_y + 1)
```

- In both cases lower fitness values denote “fitter” individuals.
- `Width` and `height` are the dimensions of the fiducial’s bounding box.
- The symmetry values `black_symmetry` and `white_symmetry` (computed separately for for black and white nodes respectively) are computed by calculating two centroids consisting of all leaf nodes on each side of a vertical center line, reflecting one centroid around this line and computing the distance between the two centroids. Nodes lying within a small margin around the line of symmetry are ignored.
- `Black_centroid_distance_x` is the absolute difference between the centroid of all leafs and the centroid of black leafs. `Black_distances_below_y`is the sum of the y coordinates of all black leaf nodes below the vertical center.

4.3. Computation environment

We implemented a simple but effective distributed computing system which allowed us to compute a number of different versions of each fiducial set within a reasonable period of time. The distribution system consisted of a pair of Python scripts. A client script bootstrapped itself from the server script by downloading an executable containing the genetic algorithm. The client then used this executable to run sets of iterations on pool files received from the server. This mechanism allowed us to distribute the workload across processors on our cluster and to desktop machines running both Linux and Windows.

When producing a set of 128 square fiducials with 19 tree nodes with a maximum depth of 3, we obtained usable results after 12 hours of computation time on a cluster of 12 dual processor 1Ghz Pentium 3 machines. In this time we computed 20 pools of 500 fiducials for each of 128 trees and selected the best result for each tree according to size and orientation vector length.

4.4. Final Selection

As noted above, multiple gene pools were evaluated for each fiducial. The selection of the final fiducial from among these pools was relatively ad-hoc. We found that the fitness functions which were necessary to drive the GA toward acceptable results were not necessarily the same as the evaluation criteria we used to select the

final fiducials. The final selection criteria were much simpler and were based solely on thresholding size and direction vector lengths.

From 128 square fiducials bred for the reacTable only 89 met our requirements for size and direction vector length. Anecdotal evidence suggests that this is due to inherent size limits for some fiducial topologies. For example, trees with many only-child leafs occupy more space in the rendering.

5. Fiducial Graphics

Although reacTIVision fiducials may have broader application, they have been designed for a specific musical instrument. The following subsections review the implications of our approach to the usability and aesthetics of the reacTable.

5.1. Usability Considerations

ReacTIVision fiducials have been designed to support fast and robust machine tracking. Although an expert may be able decode the tree sequences of reacTIVision fiducials just by looking at them, this process would consume too much time for recognizing objects in a concert situation. The ARtoolkit [8] offers a simple solution in that markers are equally readable for the machine and the human user, since it supports the use of any graphical symbol as a fiducial. In an earlier design phase of the reacTable* we developed a tactile coding scheme [9] to support rapid identification of the various synthesizer objects. This included the use of shapes, materials and surface structures allowing various physical artifacts to be recognised even in bad lighting conditions. In the current implementation we employ a simplified colour and shape code with plain plexi-glass artifacts, mostly for aesthetic reasons.

Out current coding scheme uses object shapes (square, circle, rounded square, pentagon and dome) to represent generic object classes (generator, effect, controller, mixer and synchronizer). An additional colour code allows the identification of individual object instances. The colour is printed into the white areas of the fiducial symbol and is only visible to the human player - it is actually invisible (white) to the infrared camera. Only one half of the white area is coloured in order to indicate the orientation of the symbol. This allows an exact angle to be chosen when an object is first placed on the table. Expert users could also determine a marker's orientation from the distribution of black and white leaf nodes, however the coloured marking proved to be much easier to recognize.

Most objects in the current reacTable* configuration have fiducial symbols affixed to both sides, which allows multiple uses of a single physical artifact. Flipping the object changes its behaviour. Cube objects for example provide multiple variations of simple oscillators. Each side is uniquely marked by filling a different number of white leaf nodes with colour. This coding has not proved to be very effective, especially in the bad lighting conditions of a concert situation.

5.2. Functional vs. Aesthetic Fiducials

In an earlier phase of reacTIVision's evolution the original d-touch markers were used. These had a very simple geometric structure with a somewhat technical aesthetic. Although the markers described in this paper still follow the same topological approach, they are generally considered more aesthetically appealing due to their organic appearance. Novice players usually consider the markers as a decorative element rather than a technically necessary component. Therefore the markers integrate very well into the overall visual design of the complete instrument. While our system did not initially allow the combination of aesthetics and function as for example the ARtoolkit approach provides, in the end it has defined an individual aesthetic domain for itself.

6. Applications

In the present reacTable* synthesizer configuration we use a total 74 fiducial symbols. Two players are each provided with a set of 20 artifacts, most of which are labeled on two or more sides. For the reacTable* installation at the International Computer Music Conference 2005, which took place in Barcelona, each conference attendee was provided with a unique fiducial symbol on the back of their conference badge. When placed onto the table the conference badge could be used to play back a personalised sample which was retrieved from the FreeSound project database [10]. For this task an additional set of fiducial symbols were created to support roughly 312 additional unique markers. The conference badge allowed the use of a slightly larger rectangular area of 5×9cm compared to the 6×6cm square of the core synthesizer set. In order to support the necessary number of markers at the required size we also used colour-inverted versions of each marker, which doubles the number of possible unique markers.

As already noted above, the reacTIVision was designed as the custom tailored sensor component for the reacTable* instrument. However, as a result of the well documented, standards-based TUIO protocol [3] which reacTIVision uses, it can be used for virtually any table based tangible user interface where objects need to be tracked on a 2D surface looking either from above or below. One example of such an alternative interface was developed at the Interface Culture Lab at the Art University of Linz in Austria: The recipe table [11] retrieves recipes for any ingredients which are placed onto a translucent surface. Conceptually the reacTIVision fiducials are proxies for bar codes used to identify and track the ingredients more easily.

7. Future Work

As noted above, although we had aimed to produce different shaped fiducials, we have thus far only been successful in crafting fitness functions for square and rectangular fiducials. Circular, pentagonal and triangular fiducials continue to be desirable for the reacTable project as we are already using plexi-glass objects of these shapes.

8 Conclusion

Overall we were pleased with the results achieved by applying evolutionary computation to the problem of laying out marker geometries. Our fitness functions were good for square and rectangular fiducials but we found it difficult to successfully craft fitness functions for other shapes. The GA was able to solve both the size, centroid, and distance-vector constraints simultaneously with reasonable although not perfect results. We didn't experiment with a broad range of breeding parameters to categorically state that the values used could not be improved upon although intuitively we feel that the results obtained were close to optimal. We were pleased with the aesthetic outcome which we feel is a better fit for the reacTable* musical instrument than more "engineered" marker design approaches.

References

[1] S. Jordà, "Sonigraphical instruments: From FMOL to the reacTable*," in *Proc. of the 3rd Conf. on New Interfaces for Musical Expression (NIME)*, 2003, pp. 70–76.

[2] M. Kaltenbrunner, G. Geiger, and S. Jordà, "Dynamic patches for live musical performance," in *Proc. of the 4th Conf. on New Interfaces for Musical Expression (NIME)*, 2004, pp. 19–22.

[3] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza, "TUIO: A protocol for table-top tangible user interfaces," in *Proc. of the The 6th Int'l Workshop on Gesture in Human-Computer Interaction and Simulation*, 2005 (to appear).

[4] M. Wright, A. Freed, and A. Momeni, "Open sound control: State of the art 2003," in *Proc. of the New Interfaces for Musical Expression Conf. (NIME)*, 2003, pp. 153–159.

[5] E. Costanza and J. A. Robinson, "A region adjacency tree approach to the detection and design of fiducials," in *Vision, Video and Graphics (VVG)*, 2003, pp. 63–70.

[6] R. Bencina, M. Kaltenbrunner, , and S. Jordà, "Improved topological fiducial tracking in the reactivision system," in *Proc. of the IEEE Int. Workshop on Projector-Camera Systems (PROCAMS 2005)*, 2005.

[7] T. Asai, H. Arimura, T. Uno, and S. ichi Nakano, "Discovering frequent substructures in large unordered trees," in *Proc. of the 6th Int'l Conf. on Discovery Science, DS'03*, vol. 2843. LNCS, 2003, pp. 47–61.

[8] H. Kato and M. Billinghurst, "Marker tracking and HMD calibration for a video-based augmented reality conferencing system," in *Proc. of the 2nd IEEE and ACM Int'l Workshop on Augmented Reality (IWAR)*, 1999, pp. 85–94.

[9] M. Kaltenbrunner, S. O'Modhrain, and E. Costanza, "Object design considerations for tangible musical interfaces," in *Proc. of the AISB 2004 COST287-ConGAS Symposium on Gesture Interfaces for Multimedia Systems*, 2004.

[10] the freesound project. [Online]. Available: http://freesound.iua.upf.edu/

[11] recipe-table. [Online]. Available: http://www.recipetable.net/

Appendix: Examples of reacTIVision fiducials

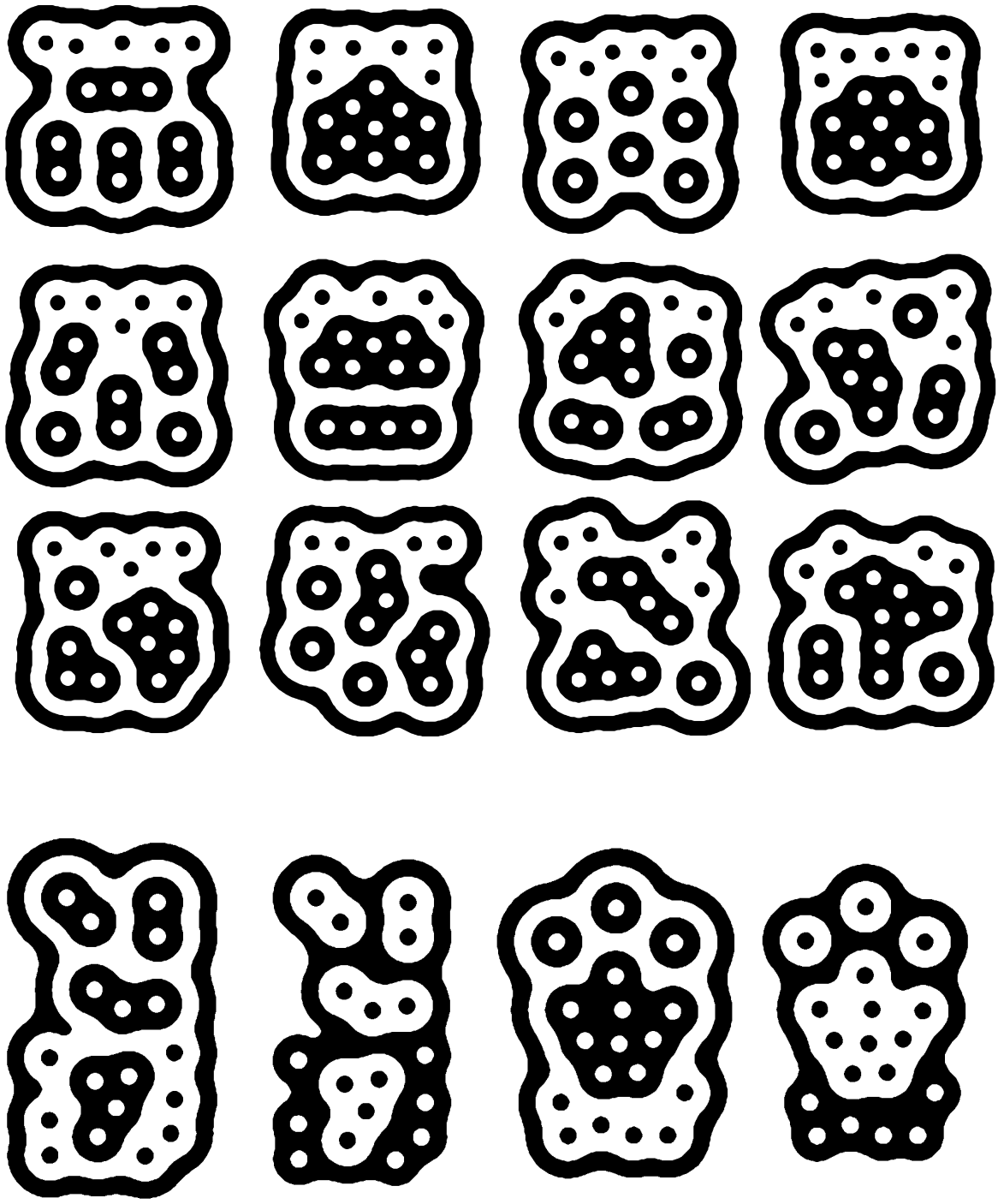

Figure 5: Example reacTable* fiducials. Top two rows: six symmetrical and six asymmetrical square fiducials. Bottom row: 5×9 (business card size) fiducials, positive and inverted versions.