Formalising Harmony Seeking Rules of Morphogenesis

Tim Hoverd and Susan Stepney

Department of Computer Science, University of York, UK, YO10 5DD
tim.hoverd@cs.york.ac.uk susan@cs.york.ac.uk

Abstract

The 15 generative patterns of Alexander's "Nature of Order" are descriptions of architectural structures that are seen in both buildings and in the natural world. We are investigating various aspects of complex systems, including those relating to structural patterns that may underlie those systems. Here we describe some experiments to generate 2D structures that incorporate those patterns that Alexander describes as *Positive Space* the voids that contribute to the overall pattern, and *Levels of Scale* a gradation in the size of the pattern's components. We show some of the results, illustrating that these patterns can be achieved as emergent properties of simple placement algorithms with a generative component.

Introduction

Studies of morphogenesis in ALife are typically inspired by biological growth and development process. However, there are other systems that grow and develop, influencing and influenced by their environment: buildings and towns. Here we investigate using these processes as an alternative source of inspiration.

Alexander's *Generative Patterns* [Alexander, 2004] are a vision of the way that successful architectural forms can be seen as the product of the generative application of a small number of properties that are seen in those forms. They attempt to describe the way that an architectural whole, be it a house or a city, evolves as a consequence of its environment and use. For example, [Alexander, 2004] shows a diagram of ancient Rome and discusses how that particular configuration emerged from the human use and development of the city.

We are examining these *Generative Patterns* to investigate the way that such approaches work. Our long term goal is to apply these properties, or similar ones, to the generative development of the architecture of *complex systems*: systems whose complex behaviour *emerges* from the simple behaviour of a large number of elements. But first it is necessary to explore Alexander's patterns in more detail, and to be able to synthesise structures that satisfy his criteria.

Here we discuss Alexander's patterns, and show the results of a computer program that uses a number of different algorithms which attempt to generate structures that match two of his generative patterns.

The Nature of Order

The four volumes of *The Nature of Order* [Alexander, 2004] explore the notion of *Wholeness* in relation to architectural structures. *Wholeness* is Alexander's enigmatic term for the "quality without a name" that he identified earlier in [Alexander, 1979]. In *The Nature of Order*, Alexander identifies 15 *generative properties* as the root characteristics of those architectural structures that form a satisfactory *whole*.

Alexander describes structures in terms of *centres*, each of which is "a zone of coherence in space". A centre is a region that is in some way *coherent* in the way it represents the space and its use. By *"coherence"* Alexander means that a centre is distinct from those around it and within it, but that in some way it contributes to the coherence of those other centres. Alexander refers to these as "centres" as they are "centres of influence, centres of action, centres of other centres" [Alexander, 2004, vol.1, p108]. One particular reason for using the word "centre" is that he is trying to describe things that have no specific boundary; a pond, for example, might include the pipes bringing in water, the rocks on its edge [Alexander, 2004, vol.1, p84] A centre is something noticeable about a structure; something that draws attention from neighbouring structures. Examples might be [Appleton, 1997] a row of tiles on a ceiling or floor, a hallway, a pond in the countryside, and—in the context of software development—what are known as "patterns" [Gamma et al., 1995].

The generative properties are used to describe a structure, as a system of centres, and to show the ways that that structure can be further elaborated and extended, or *generated*, as a region is architecturally developed. Alexander sees this as an evolutionary process, where the system of centres is progressively developed using the same set of generative processes which each application of these processes being dependent on the current structure. For example, Alexander [Alexander, 2004, vol.2, pp252–255] describes how the structure of St Mark's Square in Venice can be described as the current end product of an evolutionary process. At each step of this process, Alexander identifies *latent centres* and shows how, in his view, new building supported and strengthened these centres.

Generative Properties

The 15 properties are described in [Alexander, 2004, vol.1] as:

Levels of Scale *"how a centre is made stronger (more coherent) by the smaller strong centres within it and the larger strong centres that surround it."*

Positive Space *"the way that a given centre must draw its strength, in part, from the strength of other centres immediately adjacent to it in space."*

Roughness *"the way that the field effect of a given centre draws its strength, necessarily, from irregularities in the sizes, shapes and arrangements of other nearby centres"*

Alternating Repetition *"the way in which centres are strengthened when they repeat, by the insertion of other centres between the repeating ones"*

Thick Boundary *"the way in which the field-like effect of a centre is strengthened by the creation of a ring-like centre, made of smaller centres which surround and intensify the first. [It] also unites the centre with the centres beyond it, thus strengthening it further"*

Good shape *"the way that the strength of a given centre depends on its actual shape and the way this effect requires that even the shape, its boundary, and the space around it are made up on strong centres."*

Local Symmetry *"the way that the intensity of a given centre is increased by the extent to which other smaller centres that it contains are themselves arranged in locally symmetrical groups"*

Contrast *"the way that a centre is strengthened by the sharpness of the distinction between its character and the character of surrounding centres"*

Gradient *"the way in which a centre is strengthened by a global series of different-sized centres which then **point** to the new centre and intensify its field effect"*

Deep Interlock and Ambiguity *"the way in which the intensity of a given centre can be increased when it is attached to nearby strong centres, through a third set of strong centres that ambiguously belong to both"*

Echoes *"the way that the strength of a given centre depends on similarities of angle and orientation and systems of centres forming characteristic angles thus forming larger centres, among the centres it contains"*

Simplicity and Inner Calm *"the way the strength of a centre depends on its simplicity - on the process of reducing the **number** of different centres which exist in it, while increasing the **strength** of these centres to make them weigh more"*

The Void *"the way that the intensity of every centre depends on the existence of a still place - an empty centre - somewhere in its field"*

Not Separateness *"the way the life and strength of a centre depends on the extent to which that centre is merged smoothly - sometimes even indistinguishably - with the centres that form its surroundings"*

Strong Centre *"defines the way that a strong centre requires a special field-like effect, created by other centres, as the primary source of its strength"*

These 15 separate properties address the same thing: the manner in which centres interact to increase the overall coherence of the space. Our long term objective is to examine how these properties, or analogous ones, might apply in the context of the evolutionary development of *complex systems* architectures. We start by examining two of these properties in more detail: *Positive Space* and *Levels of Scale*.

Positive Space

"Positive Space" is conventionally used to describe *"space that is occupied by a filled shape or a positive form"* [Wong, 1993]. The positive space is the figure at the centre of attention; it is the part of the figure that the eye sees. In this sense positive space is in contrast with the negative space that surrounds the positive; it is the "figure" not the "ground".

Alexander describes the space between the artefacts of a built environment as ideally being *Positive Space*. This is in contrast with the conventional use of the term *negative space* where an artist "relies on the space that surrounds the subject to provide shape and meaning" [Bar, 2009].

Positive Space is that space which, although the space between other parts of a structure, itself contributes towards the "wholeness". That is, if the structure represents a coherent whole, then the space between the built artefacts is itself (also) positive, in that it contributes to the overall coherence rather than just being the (negative) space between those artefacts. So the figure *and* the ground are both positive, in a coherent whole.

An extreme example of this is the Escher wood-cut "Day and Night" [Escher, 1938]: the space between flying geese is yet more geese, heading in the opposite direction. That is, the "space" has its own positive structure. The same relationship appears in non-spatial examples, too. For example, Tsur shows how the same concepts occur in areas such as music and poetry [Tsur, 2000].

Levels of Scale

Centres, the structural components of the architectural space, are made more "coherent" by the presence of both larger and smaller centres in the overall structure. A particular architectural space is overall more coherent if the various structures, and indeed the non-structures that are the *Positive Space* display a degree of gradation in their sizes. For example, a large structure placed next to a collection of smaller structures might represent an overall structure that was more "whole".

If the changes in scale are too extreme the centres would not be seen as increasing each other's coherence. Alexander shows how coherent structures often contain a number of levels of scale in the ratio of about 3:1 [Alexander, 2004, vol.1]. The same ratio appears elsewhere; Salingaros shows levels of scale in the centres of a carpet design which appear in the ratio 3:1 over eight levels of scale [Salingaros, 1995].

BlobWorld: Exploring the properties

We first examine the properties of *Positive Space* and *Levels of Scale*. We do this in a very simplified simulation, of "blobs" (round or square) being placed in an environment of previously placed blobs.

Our *BlobWorld* application generates simple diagrams that have greater or lesser degrees of these properties, dependent on various parameter values and the particular algorithms used. These algorithms are designed in such a way that, are far as possible, aspects of the desired properties *emerge* as a result of the generative processes, rather than being explicitly encoded.

Contingent Placement Algorithm

The first algorithm, *contingent placement*, attempts to produce emergent *Positive Space*. It attempts to place a blob at a given position; if it is obstructed by existing blobs, the new blob is moved along a randomly chosen direction until it is no longer obstructed. So the placement is contingent on the presence of pre-existing blobs. The algorithm is given in figure 1, in which:

blobShape is "round" or "square".

sizePDF is the probability distribution function (pdf) used to generate blob sizes (see later).

visProb is the probability of a blob being visible. Early versions of BlobWorld did not have this parameter and blobs were always visible on the diagram. The addition of "invisible" blobs (which are not visible but nevertheless affect the placement of other blobs) has a significant effect on the appearance of *Positive Space* in the resulting diagrams.

blobCount is the total number of blobs (both visible and invisible).

```
blob[0] := new Blob(blobShape)
blob[0].setSize(sizePDF)
blob[0].setVis(boolean according to visProb)
blob[0].setPosition(origin)
blob[0].draw()
for i = 1..blobCount-1 do
    blob[i] := new Blob(blobShape)
    blob[i].setSize(sizePDF)
    blob[i].setVis(boolean according to visProb)
    blob[i].setPosition{blobs[0].getPosition()
            | blobs[i-1].getPosition()
            | blobs[random(0..i-1)].getPosition()}
    blob[i].setDirection(rand in 0 ... 360°)
    while not blob[i].isOverlapAcceptable(
            allowedOverlap) do
        blob[i].movePositionAlongDirection()
    end while
    blob[i].draw()
end for
```

Figure 1: Pseudo-code for the *contingent placement* algorithm

allowedOverlap determines how much a blob is allowed to overlap other blobs: when positive, blobs may overlap by an amount determined by the magnitude of this parameter; when zero blobs just touch; when negative, blobs have a small amount, determined by the magnitude of the parameter, of clear space around them.

setPosition takes one of three arguments: the centre of the initial blob, or the most recently placed blob, or a random blob, to start off the current blob. (In this paper, the initial blob position is always used.)

Every run creates a unique pattern of blobs which is highly dependent on the various parameters. Although the algorithm is simple, with appropriate parameter choices it is capable of generating patterns that display a significant degree of *Positive Space*. Three examples of generated patterns are shown in figure 2.

In most cases where **vis** = 1, (that is, where all blobs are always visible) the generated patterns show no significant degree of *Positive Space* (for example, figure 2a where the space is nothing more than a lack of blobs; it is ordinary "negative space").

The algorithm is more successful at generating *Positive Space* when some blobs are invisible (for example, figure 2b). The invisible blobs generate additional space, which enables the appearance of *Positive Space*. Figure 2b shows the effect of the *Positive Space* : in the left of these pictures, the observer gets a powerful impression of the space itself constraining, for example, the curve of blobs at the lower right corner. In many of the diagrams generated in this manner, the *Positive Space* does not exactly align with the invis-

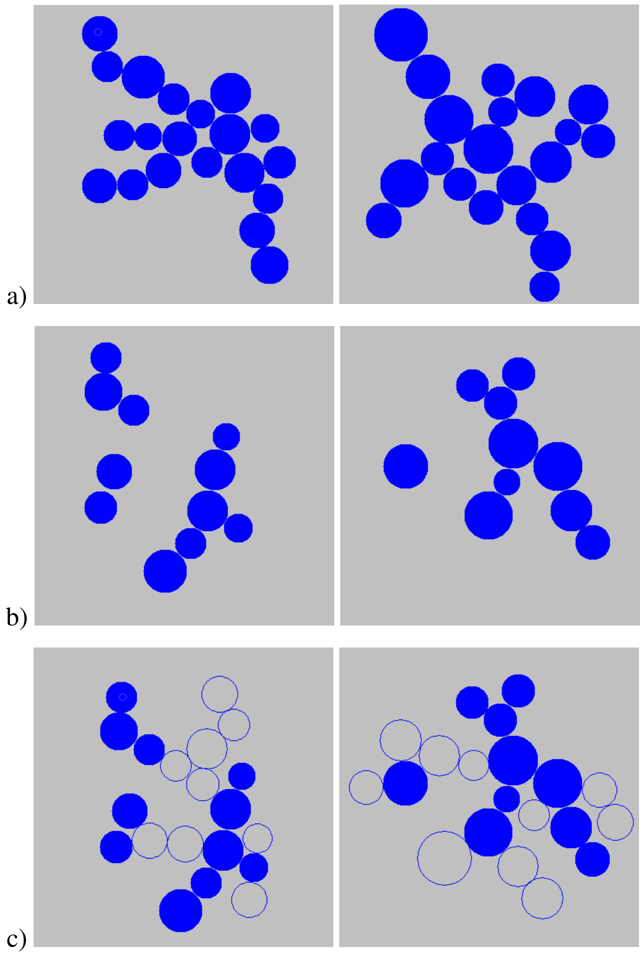


Figure 2: Results of the *contingent placement* algorithm with **blobCount** = 20, **sizePDF** = gaussian, **blobShape** = round, **allowedOverlap** = 0 : (a) **vis** = 1 ; (b) **vis** = 0.5 ; (c) as b, but with the position of the "invisible" blobs shown

ible blobs. That is, although the invisible blobs are in some way enabling the emergence of *Positive Space*, they are not themselves that space (figure 2c).

This successful generation of positive space is not dependent on using round blobs. The same effects are generated with square blobs (figure 3). Again, without the invisible blobs there is little sign of *Positive Space* (figure 3a), but when invisible blobs are introduced they create *Positive Space* (figure 3b).

With the square blobs, a further effect is visible. Here we have used a negative **allowedOverlap**, to separate the blobs from each other along their straight boundaries. Although the blobs are all perfectly aligned squares, an optical illusion makes some edges look slightly tilted or slightly bowed; this adds a degree of *Roughness* (another of Alexander's generative properties) to the picture.

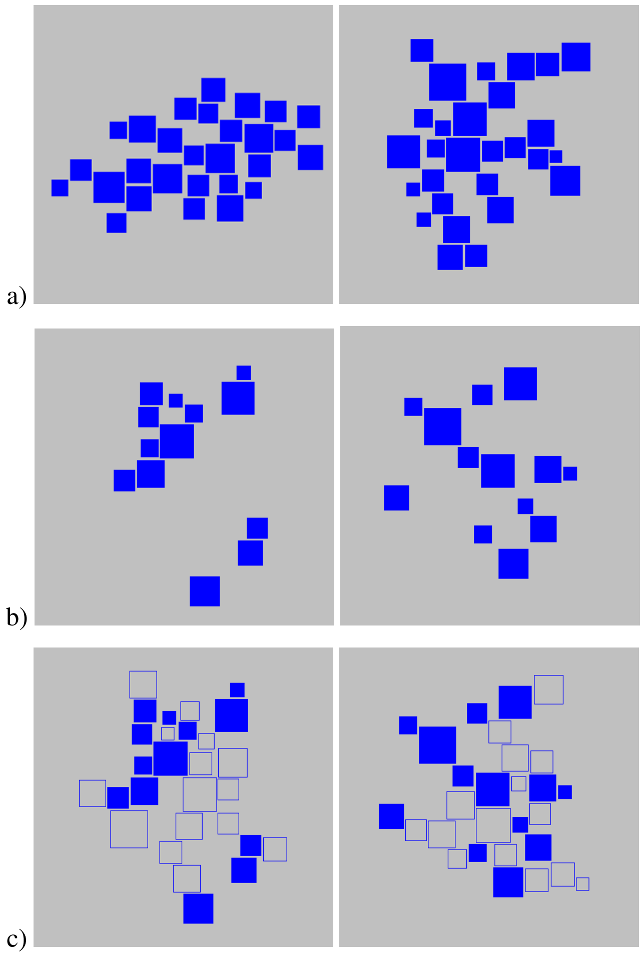


Figure 3: Results of the *contingent placement* algorithm with **blobCount** = 28, **sizePDF** = gaussian, **blobShape** = square, **allowedOverlap** < 0 : (a) **vis** = 1 ; (b) **vis** = 0.5 ; (c) as b, but with the "invisible" blobs shown

Independent Placement Algorithm

In order to test whether *Positive Space* is manifested in any diagram that merely contains "invisible" blobs a second algorithm is also implemented by BlobWorld. This *independent placement* algorithm positions blobs not as a consequence of the positions of other blobs but as an initial step of the algorithm. In essence, the *contingent placement* algorithm positions blobs of a pre-determined size in a field of other blobs as the diagram evolves from a single blob. In contrast, the *independent placement* places blobs entirely independently of each other but then manipulates the *size* of all of the blobs until the diagram, as a whole, achieves the stated requirements for blob overlap.

The *independent placement* algorithm is described by the pseudo-code in figure 4 in which:

growthPDF is the pdf used to generate the growth rate of each blob (see later).

```
for i = 0..blobCount-1 do
  blob[i] := new Blob(blobShape)
  blob[i].setGrowthRate(growthPDF)
  blob[i].setSize(1)
  blob[i].setVis(boolean according to visProb)
  blob[i].setPosition(positionPDF)
  blob[i].unfreeze()
end for
while exists an unfrozen blob do
  for i = 0..blobCount-1 do
    if blob[i] is unfrozen then
      blob[i].setSize(
          blobs[i].getSize * blobs[i].getGrowthRate)
      blob[i].draw()
    end if
    if blob[i].overlapsOtherBlob(allowedOverlap) then
      blob[i].freeze()
    end if
  end for
end while
```

Figure 4: Pseudo-code for the *independent placement* algorithm

positionPDF is the pdf used to generate the initial position of each blob. (Here it is a uniform distribution across the drawing space.)

Examples of the *independent placement* algorithm are shown in figure 5. (One of the effects of the algorithm is that pairs of same-sized blobs occur often: if two nearby blobs have the same growth rate, they both grow at this same rate until they come into contact and become frozen.) Although the diagrams generated with this algorithm do contain space, it is not *Positive Space*. That is, space that is there does not contribute to the overall coherence of the pattern; essentially, it is merely a random collection of blobs of different sizes.

Positive Space appears in the results of the contingent placement algorithm only when the invisible blobs are allowed. However, invisible blobs do not result in *Positive Space* in the independent placement algorithm (figure 6). It is clear that the space does not have the same coherent influence as that seen in the results of the contingent placement algorithm.

The essential difference between the two algorithms is that the *contingent placement* algorithm places blobs in positions determined, to some extent, by the blobs that already exist. That is, it is essentially generative in nature. In contrast, the *independent placement* algorithm pre-determines the placement of the blobs. It naturally results in space within the pattern: the blobs cannot enlarge to fill the entire space given their fixed starting positions. But it does not generate *Positive Space*.

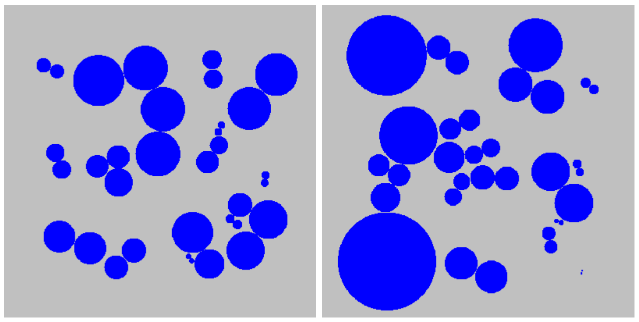


Figure 5: Typical results of the *independent placement* algorithm with **blobCount** = 34, **growthPDF** = gaussian, **blobShape** = round, **allowedOverlap** = 0 ; **vis** = 1

Levels of Scale Algorithm

With BlobWorld we can also start to explore the *Levels of Scale* property. As seen in the *placement* algorithms, the blob sizes are chosen according to a pdf; there a guassian (normal) distribution is used (with a user defined mean and standard deviation). This generates a range of sizes (figures 2, 3), resulting in some *Roughness*, but does not exhibit the 3:1 *Levels of Scale* property.

To investigate *Levels of Scale* we use bi-modal and tri-modal pdfs for size, where the mean (size) and occurrence likelihood (number) of blobs in the different modes have a fixed ratio of 3:1 (figure 7).

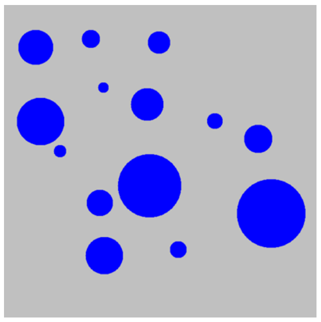
Figure 8 shows three blob figures generated using the bi-modal size distribution. The first and second examples show little evidence of the *Levels of Scale* property. The sizes follow the 3:1 distribution, but because that size has no effect on blob placement there is little evidence of any *coherence* in the size distributions spatially.

Our hypothesis is that to achieve the *Levels of Scale* property the various blob sizes would need to be arranged in such a way that changes in size are also, to some extent, reflected in their positions. Such an arrangement seldom appears in the context of either of the BlobWorld algorithms, as the blob sizes are either pre-determined, as in the contingent placement algorithm, or a consequence of the position of only the nearest other blob, as in the independent placement algorithm.

Occasionally, some degree of *Levels of Scale* is visible in BlobWorld patterns, for example in figure 8c in the two near-vertical "walls" at bottom centre, and in figure 9. This suggests that a small modification to the algorithm might well be capable of generated a suitable degree of *Levels of Scale*. This leads to our *generative size* algorithm.

Generative size algorithm

Experience with the *independent placement* and *contingent placement* algorithms shows that when blobs are positioned *generatively* then a diagram that demonstrates Alexander's

a)


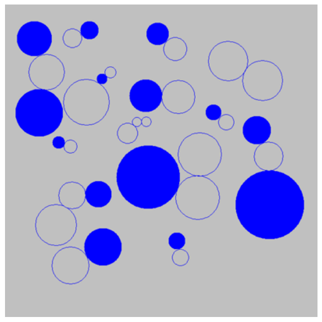
b)


Figure 6: Typical results of the *independent placement* algorithm with **blobCount** = 34, **growthPDF** = gaussian, **blobShape** = round, **allowedOverlap** = 0 ; **vis** = 0.5 (a) invisible blobs not shown; (b) as a, but with the "invisible" blobs shown

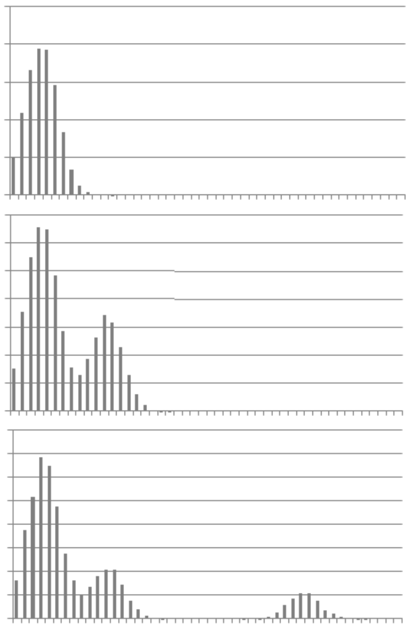


Figure 7: pdfs for investigating *Levels of Scale*. The x-axis is the blob size; the y-axis is the probability of that size: (a) single mode, gaussian distribution; (b) bi-modal, generating (approximately) three blobs of size 1 for every blob of size 3; (c) tri-modal generating (approximately) nine blobs of size 1 and three of size 3 for every blob of size 9

Positive Space property appears. That, when the diagram evolves from a small core in accordance then the result approximates a property that is observed in the end result of human-developed architecture.

However, the initial *contingent placement* algorithm is generative only with respect to the position of the blobs; their size is determined independently according to the pdfs discussed above.

A further algorithm exploits this observation by making both position and the size of the blobs the result of a generative process. It is essentially a simple modification of the *contingent placement* algorithm and the pseudo-code appears in figure 10 in which:

sizeRatio is the ratio is size between different "generations" of blob.

That is, as the algorithm is searching for a valid position for the blob it repetitively reduces the size of the blob in accordance with some predefined ratio. The effect of this is to make the size of each blob the result of a generative process which is influenced by the "environment" of each blob.

Results of executing this *generative size* algorithm are shown in figure 11. These diagrams are initially strongly reminscent of the diagrams Alexander shows as representative of the layout of cities and structures which are the result of long-term human development [Alexander, 2004]: the blobs are positioned and sized in an generative manner that is a consequence of the positioning and sizing of pre-existing blobs as the diagram evolves. As can be seen from the diagrams in the figure the blobs are now showing evidence of the *Levels of Scale* property in that the blobs appear in a wide range of sizes but there are frequent clumps of similarly sized blobs.

Conclusions

The results of these initial BlobWorld experiments are encouraging. Our *contingent placement* algorithm is capable of generating diagrams that exhibit the *Positive Space* property. That the alternative *indepenedent placement* algorithm does not have this capability indicates that the effects observed are more than mere chance.

It is likely that this capability of the *contingent placement* algorithm is due to the combination of two aspects. Firstly, the invisible blobs generate spaces that do indeed have a positive aspect, in that they contain blobs; the space is more than mere empty space, there is actually something there: (invisible!) blobs. Secondly, the algorithm is to some degree generative, in that blobs are placed in positions that are strongly conditioned by the position of existing blobs. That is, the pattern does in fact *grow* towards its final configuration.

Conversely, the *independent placement* algorithm does it-

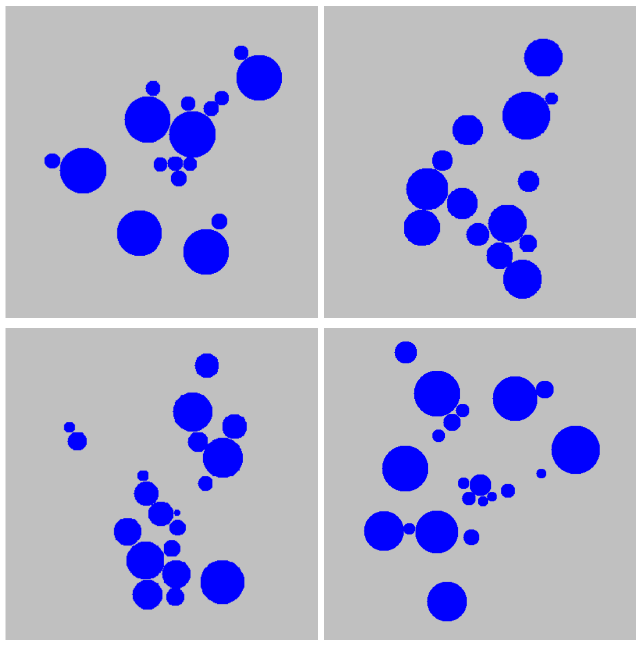


Figure 8: Attempts to generate *Levels of Scale*: contingent placement algorithm, bi-modal size distribution with a small standard deviation, **vis** = 0.5.

self naturally generate spaces. However, those spaces do not jostle directly against the blobs; the blobs jostle against each other. That is, the space is not *positive*, it is merely empty (negative) space.

Our attempts at generating the *Levels of Scale* property are also successful. The initial, somewhat explicit, attempt does not succeed in generating this property. However, the less explicit *generative size* algorithm shows that when blob size is made a direct consequence of the underlying generative process (that is when the size is a consequence of the evolution of the diagram) then the *Levels of Scale* property appears naturally in the resulting diagrams.

There is, therefore, a complex interaction of size and position taking place as the diagram evolves. Futher work is needed to establish the details of this interaction.

Future Work

This is the first step in a programme looking at Alexander's 15 generative properties. It is sufficiently successful to indicate immediately some further work, in particular on a generative algorithm that influences other properties. We have already remarked that a degree of *roughness* has emerged in the diagrams, as a consquence of optical effects and the inevitable quantisation of size and position due to the current algorithms.

What is obviously missing from the current work is some element of *measurement*. In particular, just because some diagrams appear to us to be more "whole" does not mean that

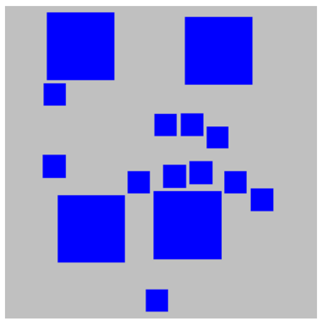


Figure 9: Attempt to generate *Levels of Scale* occasionally work: contingent placement algorithm, bi-modal size distribution, **vis** = 0.5.

```
blob[0] := new Blob(blobShape)
blob[0].setSize(sizePDF)
blob[0].setVis(boolean according to visProb)
blob[0].setPosition(origin)
blob[0].draw()
for i = 1..blobCount-1 do
  blob[i] := new Blob(blobShape)
  blob[i].setSize(sizePDF)
  blob[i].setVis(boolean according to visProb)
  blob[i].setPosition{blob[0].getPosition()
        | blob[i-1].getPosition()
        | blob[random(0..i-1)]. getPosition()}
  blob[i].setDirection(rand in 0 ... 360°)
  while not blob[i].isOverlapAcceptable(
        allowedOverlap) do
    blob[i].movePositionAlongDirection()
    blob[i].reduceSize(sizeRatio)
  end while
  blob[i].draw()
end for
```

Figure 10: Pseudo-code for the *generative size* algorithm

that is objectively true. The *Nature of Order* includes some work, in particular the "bead game" [Gabriel, 1996], that shows that some aspects of the perception of "wholeness" are universal. We will address this by means of a scoring exercise in which a number of subjects will attempt to mark different blob patterns. We will compare these scores with the parameters used to generate the patterns.

What is at the moment more speculative, though, is the relevance this work could have for that of *complex systems* architectures. For example, if *Positive Space* is a particularly advantageous aspect of building structures, what does that imply for the complex systems that are the end target of this work? We will start with flocking behaviour models [Reynolds, 1987, Andrews et al., 2008], and draw an analogy between blobs and boids: for example, how might the presence of "invisible" boids affect the observed emergent

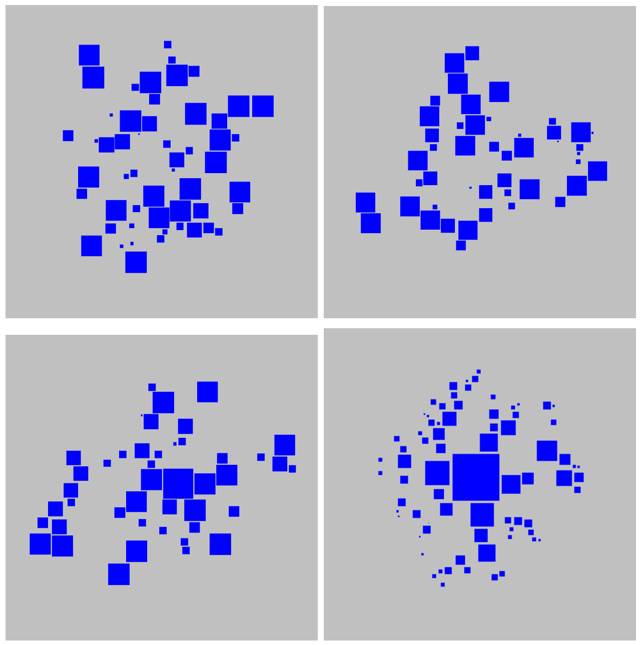


Figure 11: Typical results of the *generative size* algorithm with **blobCount** = 106, **sizePDF** = gaussian, **blobShape** = square, **allowedOverlap** = -3, **vis** = 0.5, **sizeRatio** = 1.4

flocking behaviour?

Additionally, the *Levels of Scale* property requires some form of inhomogeneous agents.

Positive Space indicates that the environment can play an imporant role in the development of the structure (recall that although "invisible blobs" are required to form *Positive Space* in our system, they are not coincident with it). This has led us to investigating the role of the environment in complex systems simulation, including taking an "environment-oriented" approach [Hoverd and Stepney, 2009] to modelling and implementation.

Alexander's properties are rooted in the consideration of structures in physical space. Design patterns [Gamma et al., 1995] are structures that exist in a design space. The emergent properties of a complex system are structures that exist in the execution space of that system, or at least of a simulation of that system. We are investigating the extent to which the ideas explored in the Nature of Order might apply to these non-physical spaces.

Acknowledgments

We would like to thank Chris Alexander for many interesting and helpful discussions.

The work described here is part of the CoSMoS[1] project, funded by EPSRC grant EP/E053505/1 and a Microsoft Research Europe PhD studentship.

[1]http://www.cosmos-research.org.

References

C. Alexander. *The Timeless Way of Building*. Oxford University Press, New York, 1979. ISBN 0195024028.

C. Alexander. *The Nature of Order, Volumes 1–4*. Center for Environmental Structure, 2004.

P. Andrews, A. Sampson, J. Bjørndalen, S. Stepney, J. Timmis, D. Warren, and P. Welch. Investigating patterns for the process-oriented modelling and simulation of space in complex systems. In *ALife XI*, pages 17–24. MIT Press, 2008.

B. Appleton. On the nature of the nature of order, 1997. http://www.cmcrossroads.com/bradapp/docs/NoNoO.html, last accessed 5 June 2010.

N. Bar. *Negative Space*. Mark Batty Publisher, 2009. ISBN 9778-0-9817805-5-9.

M. C. Escher. Day and Night, 1938. http://www.worldofescher.com/gallery/A11.html, last accessed 22 March 2010.

R. P. Gabriel. *Patterns of software: tales from the software community*. Oxford University Press, Inc. New York, NY, USA, 1996. URL http://dreamsongs.com/Files/PatternsOfSoftware.pdf.

E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995. ISBN 0-201-63361-2.

T. Hoverd and S. Stepney. Environment orientation: an architecture for simulating complex systems. In *Proceedings of the 2009 Workshop on Complex Systems Modelling and Simulation*, pages 67–82. Luniver Press, 2009.

C. W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4): 25–34, 1987. URL http://citeseer.ist.psu.edu/reynolds-flocks.html.

N. Salingaros. In defense of Alexander. *HALI: The International Magazine of Antique Carpet and Textile Art*, 78: 67–69, 1995.

R. Tsur. Metaphor and figure-ground relationship: Comparisons from poetry, music, and the visual arts. *PSYART: A Hyperlink Journal for the Psychological Study of the Arts*, (000201), 2000. http://www.clas.ufl.edu/ipsa/journal/2000_tsur03.shtml.

W. Wong. *Principles of Form and Design*. John Wiley and Sons, 1993.