1 INTRODUCTION

This book is to be an accessible book on patterns, their representation, and inference. There are a small number of ideas and techniques that, when mastered, make the subject more accessible. This book has arisen from ten years of a research program which the authors have embarked upon, building on the more abstract developments of metric pattern theory developed by one of the authors during the 1970s and 1980s. The material has been taught over multiple semesters as part of a second year graduate-level course in pattern theory, essentially an introduction for students interested in the representation of patterns which are observed in the natural world. The course has attracted students studying biomedical engineering, computer science, electrical engineering, and applied mathematics interested in speech recognition and computational linguistics, as well as areas of image analysis, and computer vision.

Now the concept of patterns pervades the history of intellectual endeavor; it is one of the eternal followers in human thought. It appears again and again in science, taking on different forms in the various disciplines, and made rigorous through mathematical formalization. But the concept also lives in a less stringent form in the humanities, in novels and plays, even in everyday language. We use it all the time without attributing a formal meaning to it and yet with little risk of misunderstanding. So, what do we really mean by a pattern? Can we define it in strictly logical terms? And if we can, what use can we make of such a definition?

These questions were answered by General Pattern Theory, a discipline initiated by Ulf Grenander in the late 1960s [1–5]. It has been an ambitious effort with the only original sketchy program having few if any practical applications, growing in mathematical maturity with a multitude of applications having appeared in biology/medicine and in computer vision, in language theory and object recognition, to mention but a few. Pattern theory attempts to provide an algebraic framework for describing patterns as structures regulated by rules, essentially a finite number of both local and global combinatory operations. Pattern theory takes a compositional view of the world, building more and more complex structures starting from simple ones. The basic rules for combining and building complex patterns from simpler ones are encoded via graphs and rules on transformation of these graphs.

In contrast to other dominating scientific themes, in Pattern Theory we start from the belief that real world patterns are complex: Galilean simplification that has been so successful in physics and other natural sciences will not suffice when it comes to explaining other regularities, for example in the life sciences. If one accepts this belief it follows that complexity must be allowed in the ensuing representations of knowledge. For this, probabilities naturally enter, superimposed on the graphs so as to express the variability of the real world by describing its fluctuations as randomness. Take as a goal the development of algorithms which assist in the ambitious task of image understanding or recognition. Imagine an expert studying a natural scene, trying to understand it in terms of the awesome body of knowledge that is informally available to humans about the context of the scene: identify components, relate them to each other, make statements about the fine structure as well as the overall appearance. If it is truly the goal to create algorithmic tools which assist experts in carrying out the time-consuming labor of pattern analysis, while leaving the final decision to their judgment, to arrive at more than ad hoc algorithms the subject matter knowledge must be expressed precisely and as compactly as possible.

This is the central focus of the book: ‘How can such empirical knowledge be represented in mathematical form, including both structure and the all important variability?’ This task of presenting an organized and coherent view of the field of Pattern theory seems bewildering at best. But what are today’s challenges in signal, data and pattern analysis? With the advent of
geometric increases in computational and storage resources, there has been a dramatic increase in the solution of highly complex pattern representation and recognition problems. Historically books on pattern recognition present a diverse set of problems with diverse methods for building recognition algorithms, each approach handcrafted to the particular task. The complexity and diversity of patterns in the world presents one of the most significant challenges to the pedagogical approach to the teaching of Pattern theory. Real world patterns are often the results of evolutionary change, and most times cannot be selected by the practitioner to have particular properties. The representations require models using mathematics which span multiple fields in algebra, geometry, statistics and statistical communications.

Contrasting this to the now classical field of statistical communications, it might appear that the task seems orders of magnitude bigger than modelling signal ensembles in the communication environment. Thinking historically of the now classical field of statistical communications, the discipline can be traced back far, to Helmholtz and earlier, but here we are thinking of its history in the twentieth century. For the latter a small number of parameters may be needed, means, covariances, for Gaussian noise, or the spectral density of a signal source, and so on. The development of communication engineering from the 1920s on consisted in part of formalizing the observed, more or less noisy, signals. Statistical signal processing is of course one of the great success stories of statistics/engineering. It is natural to ask why. We believe that it was because the pioneers in the field managed to construct representations of signal ensembles, models that were realistic and at the same time tractable both analytically and computationally (by analog devices at the time). The classical signalling models: choose \( s_0(t), s_1(t) \) to be orthogonal elements in \( L^2 \), with the noise model additive stationary noise with covariance representation via a complete orthonormal basis. Such a beautiful story, utilizing ideas from Fourier analysis, stationary stochastic processes, Toeplitz forms, and Bayesian inference! Eventually this resulted in more or less automated procedures for the detection and understanding of noisy signals: matched filters, optimal detectors, and the like. Today these models are familiar, they look simple and natural, but in a historical perspective the phenomena must have appeared highly complex and bewildering.

We believe the same to be true for pattern theory. The central challenge is the formalization of a small set of ideas for constructing the representations of the patterns themselves which accommodate variability and structure simultaneously. This is the point of view from which this book is written. Even though the field of pattern theory has grown considerably over the past 30 years, we have striven to emphasize its coherence. There are essentially two overarching principles. The first is the representation of regularity via graphs which essentially encode the rules of combination which allow for the generation of complex structures from simpler ones. The second is the application of transformations to generate from the exemplars entire orbits. To represent typicality probabilistic structures are superimposed on the graphs and the rules of transformation. Naturally then the conditional probabilities encode the regularity of the patterns, and become the central tool for studying pattern formation.

We have been drawn to the field of pattern theory from backgrounds in communication theory, probability theory and statistics. The overall framework fits comfortably within the source-channel view of Shannon. The underlying deep regular structures are descriptions of the source, which are hidden via the sensing channel. We believe that the principle challenge is the representation of the source of patterns, and for that reason the majority of the book is focused precisely on this topic. A multiplicity of channels or sensor models will be used throughout the book, those appropriate for the pattern class being studied. They are however studied superficially, drawn from the engineering literature and taken as given, but certainly studied more deeply elsewhere. The channel sensor models of course shape the overall performance of the inference algorithms; but the major focus of our work is on the development of stochastic models for the structural understanding of the variabilities of the patterns at the source. This also explains the major deviation of this pattern theory from that which has come to be known as pattern recognition. Only in the final chapters will pattern recognition algorithms be studied, attempting to answer the question of how well the algorithm can estimate (recognize) the source when seen through the noisy sensor channel.
6 THE CANONICAL REPRESENTATIONS OF GENERAL PATTERN THEORY

ABSTRACT Pattern theory is combinatory in spirit or, to use a fashionable term, connectionist: complex structures are built from simpler ones. To construct more general patterns, we will generalize from combinations of sites to combinations of primitives, termed generators, which are structured sets. The interactions between generators is imposed via the directed and undirected graph structures, defining how the variables at the sites of the graph interact with their neighbors in the graph. Probabilistic structures on the representations allow for expressing the variation of natural patterns. Canonical representations are established demonstrating a unified manner for viewing DAGs, MRFs, Gaussian random fields and probabilistic formal languages.

6.1 The Generators, Configurations, and Regularity of Patterns

To construct more general patterns, the random variables and state spaces are generalized via the introduction of primitives called generators which are structured sets. The generators become the random variables at the nodes of the graphs; the structure is imposed via the edge relations in the graphs constraining how the sets at the vertices of the graph interact with their neighbors. The graphs imply the conditional probabilities associated with the directed and undirected random fields.

When moving to the more general patterns on arbitrary graphs \( \sigma = (D, E) \) it is clear that building structure requires the aggregation of the field variables. This is precisely what the generators do! Examine the goal of moving from unstructured representations of pixelated images to those containing edge or line sites and continuing to object elements and highly structured sets. In an attempt to add structure, more complex abstractions are defined; the edge vertices which are aggregations of the pixels, and line and boundary vertices which are in turn aggregations of the edge and line vertices, respectively. The generators arise as aggregations of the field elements.

In an atomistic spirit we build the representations by combining simple primitives, generators. To begin with the generators will be treated as abstract mathematical entities whose only structure will be expressed in terms of bonds defining how they communicate on the graph. The mathematical objects, the generators, are from the generator space \( G \) and will appear in many forms: they can be positive pixel values in an image, states in a Markov chain, geometric objects such as vectors and surface elements, or rewriting rules in language theory.

As before, begin with the connector graph \( \sigma \) with \( n = |\sigma| \) the number of vertices in \( \sigma \). At each vertex place one of the generators \( g \in G \) the generator space. A pattern or configuration is denoted \( c(\sigma) = \sigma(g_1, \ldots, g_n) \). Associate with the structured generators bonds, information which is used to communicate to other neighbors in the graph. The bonds establish the local rules of interaction; an edge in the graph corresponds to a bond between generators. As the generators are stitched together, only certain patterns will have the structure regularity of the pattern class. To enforce rigid structure, a bond function \( \beta : G \to B \) is associated to each generator and its target defined by edges \( e = (i, j) \) between pairs of generators which are neighbors in the graph. If two vertices \( i, j \in D \) of the graph are neighbors, \( j \in N_i, i \in N_j \), then the generator \( g_i \) has a bond \( \beta_j(g_i) \) which must agree with the bond \( \beta_i(g_j) \) from \( g_j \). For sites \( i, j \) which have no edge, we assume the null bonds, with agreement trivially satisfied. The value of the bond couples determine how the generators interact and therefore how the patterns form.

The configurations are illustrated as in Figure 6.1. Panel 1 shows two generators in the graph with their bonds. Panel 2 shows a graph with the set of 6 vertices in \( D \) with associated edges dictating where the bonds must form; the right panel, the set of generators with their bonds.
6.1 GENERATORS, CONFIGURATIONS AND REGULARITY OF PATTERNS

Figure 6.1 Panel 1 shows two generators with their bonds; panel 2 shows a configuration with the set of generators, bonds, and edges in the graph.

The bonds constrain the configuration defining which ones are regular; the space of regular configurations will generally be a subset of the full configuration space $G^n$.

**Definition 6.1** Define the **unconstrained configurations** on graph $\sigma = (D, E)$ as a collection of $n$-generators,

$$C(\sigma) = \{ c(\sigma) = \sigma(g_1, g_2, \ldots, g_n) : g_i \in G^{(i)}, i = 1, \ldots, n \}. \quad (6.1)$$

A configuration is determined by its **content**, the set of generators in $G$ making it up, content$(c) = (g_1, g_2, \ldots, g_n)$, and its combinatorial structure determined by the graph with the **internal bonds** of the configuration denoted int$(c)$, and the set of the remaining ones, the **external bonds**, denoted by ext$(c)$.

Defining the **bond function** $\rho(\cdot, \cdot) : B \times B \to \{\text{TRUE}, \text{FALSE}\}$, then a configuration $c(\sigma) = \sigma(g_1, g_2, \ldots, g_n) \in C$ is said to be **regular** when the bond relation is satisfied over the graph $\sigma = (D, E)$:

$$\bigwedge_{e = (i, j) \in E} \rho(\beta_j(g_i), \beta_i(g_j)) = \text{TRUE}. \quad (6.2)$$

The **space of regular configurations**, a subset of the full configuration space, is denoted as

$$C_R \subseteq C(\sigma) = G^{(1)} \times \cdots \times G^{(n)}.$$

For many problems, we will also be interested in configuration spaces associated with the collection of graphs $C_R(\Sigma) = \bigcup_{\sigma \in \Sigma} C_R(\sigma)$.

Formula 6.2 is a structural formula expressing relations between the generators at the nodes of the graph. Sometimes **local regularity** may be distinguishable in which all of the bonds in the graph type $\sigma$ are satisfied, although the graph type itself may not be in the set of allowed graphs. The connected bonds are essentially the internal bonds of the configuration, with the set of the remaining ones, the external bonds.

In general, connector graphs may be sparse, so that many of the sites do not have an edge. The right graph in Figure 6.1 is such a case. It is then natural to explicitly index the bonds at each generator $\beta_k(g), k = 1, \ldots, \omega(g)$, with total arity $\omega(g)$ designating the number of bonds attached to $g$.

**Example 6.2 (Regular Lattice Magnetic Spins)** In the magnetic spin models such as the Ising model from statistical physics, $\sigma = \text{LATTICE}$, the generators are the magnetic dipoles in plus or minus orientation $G = \{+, -\}$, and the bond values are the terms which multiply to form the energy function: $B = \{-1, +1\}$ with $\omega(g) = 2, 4, 6$ for the
enforced via the regularity of bond consistency equals the inbond of the $\beta_{in}(g_i)$ as a $(n, g_1, \ldots, g_n)$

1,2,3D models, respectively. For the Ising case, $\rho(\beta, \beta') = \text{TRUE}$ for all $(\beta, \beta') \in B^2$; all $+1, -1$ configurations are allowed.

In 1D let $D = \{1 \leq i \leq n\}$ and choose the two bond values to be $\beta_1(g) = \beta_2(g)$ defined by $\beta(+) = 1, \beta(-) = -1$, with the truth table taking the form for the generators

$$
\begin{align*}
\rho : B \times B = & +1 & -1 \\
             & \rho = T & \rho = T \\
\end{align*}
$$

$\text{and } C_R = C = \{+, -\}^n$. (6.3)

Example 6.3 (Unclosed and Closed Contours) Examine unclosed contours in the plane, and generate the boundaries from line segments. Then $\sigma = \text{LINEAR}$, $\omega(g) = 2$, generators are arcs in the plane $g = (z_1, z_2); z_1, z_2 \in \mathbb{R}^2, G = \mathbb{R}^4$, and bond-values are start and end-points of the generators, $\beta_{in}(g) = z_1; \beta_{out}(g) = z_2$. As depicted in Figure 6.2, notice that the boundary vertex generators have one less in and out bond for LINEAR. For cyclic, the last generator interacts with the first generator.

The continuity constraints mean that consecutive line segments are joined to each other and enforced via the regularity of bond consistency $R(\text{LINEAR})$ so that the outbond of the $i$th generator equals the inbond of the $i + 1$st:

$$
\rho(\beta_{out}(g_i), \beta_{in}(g_{i+1})) = \text{TRUE} \quad \text{if and only if} \quad \beta_{out}(g_i) = \beta_{in}(g_{i+1}). \quad (6.4)
$$

For closed contours, $\sigma = \text{CYCLIC}$, with the first and last vertices having an arc in the graph, adding the cyclic regularity enforced by the added bond relation $\rho(\beta_{out}(g_n), \beta_{in}(g_1)) = \text{TRUE}$ (bottom row of Figure 6.2).

Here the generators are chosen as directed line segments, which could be generalized to other arc elements from conic sections or other curve families. For curves in $\mathbb{R}^3, z_1, z_2 \in \mathbb{R}^3, G = \mathbb{R}^6$.

Example 6.4 (Triangulated Graphs) In cortical surface generation the surfaces are generated from vertices using triangulations of the sphere. Panel 1 of Figure 6.3 depicts a triangulated sphere, the generators being

$$
\begin{align*}
\beta_{in}(g_i) &= \beta_{out}(g_i) \\
\beta_{in}(g_i) &= \beta_{out}(g_i) \\
\beta_{in}(g_i) &= \beta_{out}(g_i) \\
\end{align*}
$$

with $\omega(g) = 3$, and $\beta_j(g_i) = v_j^{(i)}; j = 1, 2, 3$. Thus $G = \mathbb{R}^9$. The graph family $\Sigma = \bigcup_n \text{TRIANG}(n)$ expresses the topology corresponding to patches which are topologically connected according to the triangulated sphere topology. A complete triangulation with congruent triangles is only possible for $n = 4, 6, 8, 12, 20$, the Platonic solids. Generally large $n$-values are used, so that noncongruent triangles form the patterns. Panels 2 and 3 show the triangulated graphs associated with the bounding closed surface of the hippocampus in the human brain and the macaque cortex.
6.1 GENERATORS, CONFIGURATIONS AND REGULARITY OF PATTERNS

Figure 6.3 Top row: Panel 1 shows the triangulated graph for the template representing amoeba corresponding to spherical closed surfaces. Generators are $g_i = (v_1, v_2, v_3)$, elements of $\mathbb{R}^3$. Panel 2 shows the triangulated for the closed surface representing the bounding closed surface of the hippocampus in the human brain. Panel 3 shows the surface representation of a macaque brain from the laboratory of David Van Essen. Bottom row: Shows a generator shown in standard position, orientation under rotation and translation transformation. (see plate 3).

Alternatively, choose the generators to be the vertices in $\mathcal{G} = \mathbb{R}^3$, and the generating points are the $n$-points sampled around the sphere:

$$g_{ij} = \left( \sin \frac{2\pi i}{m}, \cos \frac{2\pi j}{m}, \sin \frac{2\pi i}{m}, \sin \frac{2\pi j}{m}, \cos \frac{2\pi i}{m} \right), \quad (i, j) \in \mathbb{Z}_m^2.$$ (6.6)

Example 6.5 In the setup for observing moving bodies, say for tracking an aircraft, define each generator to an aircraft, $\mathcal{G} = \{AIRCRAFTS\}$ of specified types in specified positions and orientation. The center of gravity is located at some $(x_1, x_2, x_3) \in \mathbb{R}^3$; the attitude is the natural coordinate axes of the aircraft forming given angles with those of the inertial frame. Then $\dim(\mathcal{G}) = 6$. A convenient choice of generator space is to let each generator consist of an arc, for example a line segment, in location and orientation space.

A system of moving bodies need not be constrained by local regularity unless they do not occupy common volume. Then a generator has an indefinite number of bonds, all of whose values should be the set occupied by the body. The bond relation takes the form

$$\beta_1 \cap \beta_2 = \emptyset.$$ (6.7)

A configuration for tracking is a linear graph $\text{LINEAR}(m_1)$ of $m_1$ generators (airplanes) specified via the special Euclidean motion. Multiple airplanes correspond to unions of linear graphs, $\text{MULT}(m_2, \text{LINEAR})$. Examine the set of graph transformations associated with discovering tracks. These consist of such transformations as depicted in the Figure in the right panel of Figure 6.7 and in Figure 6.4 for airplane tracking.

Example 6.6 (Finite State Graphs) Let $X_i, i = 1, 2, \ldots$ be a realization from the binary 1-1s languages $X_0 = \{0, 1\}$ of binary strings not containing 2 consecutive 1 symbols in a row. Let the generators be the binary values, $\mathcal{G} = \{0, 1\}$. 

Example 6.7 Any graph transformation $\mathcal{T}$ of a linear graph $\text{LINEAR}(m)$ may be represented as a graph with $m$ generators and $m$ bonds of the form $\mathcal{T}(G)$, where $G$ is an $m 	imes m$ adjacency matrix of the linear graph.
The regular configurations become \( c(\text{LINEAR}) \in C_R(\text{LINEAR}) \) if and only if \( \beta_{\text{out}}(g_i) = \beta_{\text{in}}(g_{i+1}) \), \( i = 1, \ldots, n - 1 \).

**Example 6.7 (Model-Order, Sinusoid estimation)** In a sinusoid estimation such as for NMR [22], signals are constructed from sinusoids \( x(t) = \sum_{i=1}^{n} a_i \sin \omega_i t - \lambda_i t \), generators are sinusoids. Then a configuration \( c(\text{LINEAR}) = \text{LINEAR}(a_1, \ldots, a_n) \), linear graph denoting order so that \( a_i \) is associated with \( \sin \omega_i t \), and \( C_R(n) = C(n) = \mathbb{R}^n \). All choices of amplitudes are allowed: it is a vector space. The dimension of the model will not generally be known, so that \( C(\Sigma) = \bigcup_{n=0}^{\infty} \mathbb{R}^n \).

### 6.2 The Generators of Formal Languages and Grammars

The formal grammars of Chapter 3, Section 3.10 are an exquisite illustration of the pattern theory. They directly provide a mechanism for building the space of directed graphs \( \Sigma \) recursively: *the grammar specifies the legal rules of transformation of one graph type to another*. Naturally, finite-state grammars provide transformations which build the space of linear graphs \( \Sigma = \text{LINEAR} \), the context-free grammars the space of trees \( \Sigma = \text{TREE} \) and finally context-sensitive grammars provide transformations through the partially ordered set graphs \( \Sigma = \text{POSET} \).

What is so beautiful about the formal language theory of Chomsky is that the grammars precisely specify the generators for the configurations and the rules of transformation. They also illustrate one more non-trivial movement to more abstract generators. The patterns (strings) are built by connecting the generators in the various graph structures to satisfy the consistency relationship defined by the bond structure: \( \text{LINEAR}, \text{TREE}, \text{DAG} \). The connection of the Chomsky transformation rules is that the rules of transformation are the generators. These are the formal grammars! The transformations allowed by the rules are constrained by the consistency placed via the bonds between the generating rules.

The generators are the grammatical rules themselves, and the bond-values are subsets of the non-terminals and terminals on the right and left hand sides of the rules, elements of the terminal and non-terminal set. To see the forms for the rules of the three grammar types, refer to Figure 6.5 which shows various examples. Left panel shows a finite-state grammar production rule \( A \rightarrow B \), \( \omega_{\text{in}} = \omega_{\text{out}} = 1 \). The middle panel shows a production rule (generator) \( S \rightarrow NP VP \) for a context-free grammar; \( \omega_{\text{in}}(g) = 1, \omega_{\text{out}}(g) = 2, \beta_{\text{in}}(g) = S, \beta_{\text{out1}}(g) = NP, \) and \( \beta_{\text{out2}}(g) = VP \). The right panel shows a context-sensitive production rule \( A B \rightarrow CD \).
The local regularity comes from the fact that for two bonds $\beta, \beta'$ joined in the graph, then
\[ \text{outbond } \beta = \text{inbond } \beta'. \quad (6.9) \]
In other words, when a rewriting rule has produced certain non-terminal syntactic variables as
their daughters, then this rule can only be connected to a rule with those syntactic variables on
their left hand side: no jumps are allowed. In languages, we shall be interested in the set of strings
having only terminals.

**Definition 6.8**  Associated with every configuration define the **closed or internal bonds**
of the configuration, denoted $\text{int}(c)$, and the set of the remaining ones, the **external bonds**
denoted as $\text{ext}(c)$.

**Theorem 6.9**  Given is a formal grammar $G(P) = < V_N, V_T, R, S >$ with language $L(G)$. Then let the regular configurations be constructed from generators the rules $G = R$, with
the bonds the non-terminals and terminals on the right and left hand sides of the rules in
$B = V_N \cup V_T$:
1. for finite-state, $\sigma = \text{LINEAR}$ with generators and bonds

   \[
   A \xrightarrow{\beta} wB, \quad \beta_{\text{in}}(g) = A, \quad \beta_{\text{out}}(g) = A', \quad \omega(g) = 2; \quad (6.10)
   \]
2. for context-free, $\sigma = \text{TREE}$ with generators and bonds

   \[
   A \xrightarrow{\beta} \psi, \quad \beta_{\text{in}}(g) = A, \quad \beta_{\text{out}}(g) = \psi, \quad \omega(g) = 1 + |\psi|; \quad (6.11)
   \]
3. for context-sensitive, $\sigma = \text{DAG}$ with generators and bonds

   \[
   \alpha A \beta \xrightarrow{\psi} \alpha \psi \beta, \quad \beta_{\text{in}}(g) = \alpha A \beta, \quad \beta_{\text{out}}(g) = \psi, \quad \omega(g) = |\alpha A \beta| + |\psi|. \quad (6.12)
   \]

The regular configurations $c(\sigma) = \sigma(n, g_1, g_2, \ldots, g_n) \in C_R(\sigma)$ satisfy

\[
\bigwedge_{e = (i, j') \in E} \rho(\beta_{j(i)}(g_i), \beta_{j'(i)}(g_{i'})) = \text{TRUE}, \quad \text{where } \rho(\beta, \beta') = \text{TRUE} \iff \beta = \beta'. \quad (6.13)
\]

Then the regular configurations $C_R(\sigma) \subset C_R(\sigma), \sigma \in \Sigma = \{\text{LINEAR, TREE, DAG}\}$
having no unclosed external bonds is the formal language:

\[
\tilde{C}_R(\sigma) = \{c(\sigma) = \sigma(n, g_1, \ldots, g_n) \in C_R(\sigma) : \text{ext}(c) = \phi\} = L(G). \quad (6.14)
\]

Figure 6.5 depicts the various forms of the generators for the three grammar types. The
special rewrite rules involving the sentence symbol, $S \rightarrow \psi$ have $\omega_{\text{in}}(g) = 0$, and the rules
$\alpha A \beta \xrightarrow{\psi} \psi \in V_T^*$ with only terminal symbols on the right hand side, $\beta_{\text{out}}(g) = 0$. 

---

**Figure 6.5** Rewriting rules corresponding to the finite-state, context-free, and context-sensitive
generators.
The probabilistic versions of formal languages, the stochastic languages, the set of strings generated by probabilistic application of the rules. For the finite-state (regular) and context-free languages, these are Markov chains and multi-type random branching processes.

**Example 6.10 (Finite-State Languages)** Examine the finite-state languages of Example 3.43. Begin with the 1-1 constraint language, \( V_N = \{ \text{ZERO}, \text{ONE} \} \) with \( S = \text{ZERO} \), with the generators the transitions in the state graph:

\[
G = R = \{ \text{ONE} \xrightarrow{r_1} 0\text{ZERO}, \text{ZERO} \xrightarrow{r_2} 0\text{ZERO}, \text{ZERO} \xrightarrow{r_3} 1\text{ONE} \}. 
\]

Elements in the language can end in either states, augment the rule set with the terminating rules \( \text{ZERO} \xrightarrow{r_4} 1 \), \( \text{ZERO} \xrightarrow{r_5} 0 \), \( \text{ONE} \xrightarrow{r_6} 0 \).

For parity languages, \( V_N = \{ \text{EVEN}, \text{ODD} \} \) with \( S = \text{EVEN} \), and the rules (generators)

\[
G = R = \{ \text{EVEN} \xrightarrow{r_1} 0\text{EVEN}, \text{EVEN} \xrightarrow{r_2} 1\text{ODD}, \text{ODD} \xrightarrow{r_3} 0\text{ODD}, \text{ODD} \xrightarrow{r_4} 1\text{EVEN} \}. 
\]

Since only strings ending in \( \text{EVEN} \) parity are in the language, augment with the terminating rules \( \text{EVEN} \xrightarrow{r_5} 0 \), \( \text{ODD} \xrightarrow{r_6} 1 \). The graph is \( \sigma = \text{LINEAR} \), arity \( \omega(g) = 2 \) with in- and out-bond values

\[
\beta_{\text{in}}(g) = \text{LHS}(g), \quad \beta_{\text{out}}(g) = \text{RHS}(g), 
\]

with

\[
\rho(\beta_{\text{out}}(g_i), \beta_{\text{in}}(g_{i+1})) = \text{TRUE} \quad \text{if and only if} \quad \text{RHS}(g_i) = \text{LHS}(g_{i+1}). 
\]

The string 001001 is regular in the 1-1 language with generator representation \( \text{LINEAR}(r_2, r_2, r_3, r_1, r_2, r_4) \). The parity string 001001 in the \( \text{EVEN} \) parity language is regular with generator representation \( \text{LINEAR}(r_1, r_1, r_2, r_3, r_3, r_6) \). They both satisfy the structure relations \( \beta_{\text{out}}(g_n) = \phi \) with

\[
\beta_{\text{out}}(g_i) = \beta_{\text{in}}(g_{i+1}) \quad \text{for} \quad i = 1, \ldots, n-1. 
\]

For the **bigram** and **trigram** language models the \( m \)-gram model treats a string of words as a realization of an \( m \)-th order Markov chain in which the transition probability is the conditional probability of a word in the string given by the previous \( m - 1 \) words. The generators for the \( m \)-ary processes are

\[
\begin{align*}
\underbrace{w_{i-m}, w_{i-(m-1)}, \ldots, w_i} & \xrightarrow{A} \underbrace{w_{i-(m-1)}, w_{i-(m-2)}, \ldots, w_{i+1}} \\
\underbrace{w_{i-m}, w_{i-(m-1)}, \ldots, w_i} & \xrightarrow{A} \underbrace{w_{i+1-m}, w_{i+1-(m-1)}, w_{i+1-(m-2)}, \ldots, w_i, \phi}
\end{align*}
\]

where \( \phi = \text{null} \) (punctuation). Notice, for \( m = 1, m = 2 \) these rules are precisely the generators of node \( i \), \( (X_i, X_{n_i}) \) in Theorem 6.19, with in-arity 1 over the entire graph.

An element of the configuration space \( C(\text{LINEAR}) \) consists of a set of generators which are the rewrite rules \( \hat{G} = R \) placed at the vertices of the directed graph. A configuration consists of \( n \) nodes in a linear graph \( c = \text{LINEAR}(n, r_1, \ldots, r_n) \); rule \( r_i \in R \) transforms node \( i \).

**Example 6.11 (Phrase Structure Grammars)** Examine the structures associated with context free languages and their associated tree graphs.
The generators are the rewrite rules, and for each generator \( g \in \mathcal{G} \), the arity of the in-bonds \( \omega_{\text{in}}(g) = 1 \) and the arity of the out-bonds \( \omega_{\text{out}}(g) \geq 1 \). The in-bonds corresponding to the left hand side of the production rule, the out-bonds the right hand side of the production rule determined by the number of non-terminals. Context-free means \( \Sigma = \text{TREE} \), as depicted in 6.6. An element of the configuration space \( c(\text{TREE}) = \text{TREE}(n, g_1, g_2, \ldots, g_n) \in C \) consists of a set of \( n \)-vertices at which the rewriting rule generators are placed. The in- and out-bonds of a generator are \( \beta_1(g) \), the in-bond value the LHS of the production rule, and \( \beta_2(g), \ldots, \beta_{\omega_{\text{out}}}(g) \), the out-bond values which are elements of the set of bond values \( B = V_N \cup V_T \), where \( V_N \) is the set of non-terminal symbols and \( V_T \) is the set of terminal symbols.

Examine the simple phrase-structure grammar with seven basic rules

\[
R = \begin{cases} 
S \rightarrow NP \ VP \\
VP \rightarrow V \\
ART \rightarrow \text{the} \\
NP \rightarrow ART \ N \\
VP \rightarrow N \rightarrow V \\
\end{cases}
\]

(6.19)

Define the sentence–language as the set of all context-free trees rooted in the syntactic variable \( S \). Panel 1 of Figure 6.6 shows a sentence \( \text{TREE}(6, r_1, r_2, r_3, r_6, r_4, r_7) \). Notice, it is a single tree rooted in the sentence root node \( S = \text{sentence} \) with all

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure6_6.png}
\caption{Panel 1 shows the configuration diagram \( \text{TREE}(6, r_1, r_2, r_3, r_6, r_4, r_7) \) for the sentence \textit{The frogs jump}; panel 2 shows the configuration diagram \( \text{TREE}(6, r_1, r_2, r_3, r_6, r_4, r_5) \) for the sentence \textit{The frogs eats}, which is not locally regular. Panel 3 shows the configuration \( \text{PHRASE–FOREST} \) \((2, c_1, c_2) \) of two phrases \( \text{PHRASE} \) \((3, r_2, r_6, r_4) \), \( \text{PHRASE} \) \((2, r_3, r_7) \) each locally regular. Panel 4 shows the configuration \( \text{DIGRAPH} \) \((8, r_1, r_2, r_4, r_5, r_6, r_7, r_3, r_8) \) which is context-sensitive and is a DIGRAPH.}
\end{figure}

18 As proven by Chomsky a context-free language is fundamentally not finite-state, if at least one rule in the connected set has more than one non-terminal in the right-hand side \( (\omega_{\text{out}}(g) \geq 2) \), the so called self-embedding property.
bonds agreeing (local regularity). Panel 2 of Figure 6.6 shows a configuration $c = \text{TREE}(6, r_1, r_2, r_3, r_6, r_4, r_5)$ which is not regular. Notice the disagreement of the VP $\rightarrow V$, $N \rightarrow \text{jump}$ rules. Notice, $\beta_{\text{out}}(r_3) = V \neq \beta_{\text{in}}(r_5) = N$. Panel 3 of Figure 6.6 shows a configuration consisting of two trees each of which are phrases. The graph type PHRASE–FOREST, consists of all tree derivations rooted in any syntactic variable with leaves as the terminal strings. The configuration $c = \text{PHRASE–FOREST}(2, c_1, c_2)$ where

$$c_1 = \text{PHRASE}(3, r_2, r_6, r_4), \quad c_2 = \text{PHRASE}(2, r_3, r_7),$$

is locally and globally regular.

**Example 6.12 (Context-Sensitive Languages and POSET Graphs)** An example of a context-sensitive grammar is $V_N = \{S, NP, VP, V, DET, N\}$, $V_T = \{\text{The}, \text{dog}, \text{eat}, \text{meat}\}$ with rules

$$R = \left\{ \begin{array}{ll}
S & \Rightarrow NP \ VP, \\
NP & \Rightarrow DETN, \\
VP & \Rightarrow V \ NP, \\
DET & \Rightarrow \text{the}, \\
\text{dog} & \Rightarrow \text{eat}, \\
\text{eats} & \Rightarrow \text{eats meat}
\end{array} \right\}.$$ (6.21)

The context sensitive rules are those that rewrite a part of speech in the context of the previous word. Context-sensitive grammars derive configurations that are partially ordered sets. Examine panel 4 of Figure 6.6 showing the corresponding graph. Notice the choice of grammar places context at the terminal leaves resulting in loss of the conditional independence structure at the bottom of the tree.

### 6.3 Graph Transformations

Now explore graph transformations as the mechanism for constructing more complex structures from simpler ones. The construction of complex patterns requires rules of transformation so that more complex patterns may be constructed from simpler ones. This is certainly one of the most fundamental aspects of pattern theory, building more complex graphs from simpler ones. Graph transformations are familiar, as in linguistic parsing, where the role of the grammatical rules define the structure of the transformation. Such grammatical transformations are chosen to be rich enough to represent valid structures in the language, while at the same time restrictive enough to enforce regular grammatical structure associated with the language.

During transformation of graphs, bonds will be left unclosed, the so called external bonds, $(\text{ext}(c))$, and bonds will be closed during combination. For such changes, introduce the set of graph transformations $T : \mathcal{C}_R \rightarrow \mathcal{C}_R$ of either birth or death type.

**Definition 6.13** Define the discrete graph transformation $T \in T$ to be either of the **birth** or **death** type according to the following:

$$T \in T(c) : c(\sigma) \overset{\text{birth}}{\rightarrow} c'(\sigma') = \sigma(c, c''); \quad c, c', c'' \in \mathcal{C}_R;$$ (6.22)

$$T \in T(c) : c = \sigma(c', c'') \overset{\text{death}}{\rightarrow} c'(\sigma'); \quad c, c', c'' \in \mathcal{C}_R.$$ (6.23)

For any regular configuration $c$ then the **upper neighborhood** $\mathcal{N}_+(c)$ consists of all configurations obtained from $c$ by a **birth**ing and the **lower neighborhood** $\mathcal{N}_-(c)$ of all configurations obtained via a **death** transformation to $c$, with the full neighborhood

$$\mathcal{N}(c) = \mathcal{N}_+(c) \cup \mathcal{N}_-(c).$$

Then the set of graphs will be traversed by the **family of graph transformations** $T \in T = \bigcup_{c \in C} T(c)$ of simple moves consisting of either the **birth** or **death** types.
We shall see in defining inference algorithms with desirable properties that it is helpful to require that the transformations \(T\) give the graph space a reversibility and connectedness property.

**Definition 6.14** \(T : C_R \rightarrow C_R\) shall be **reversible** in the sense that

\[
T \in T(c) : c \mapsto c' \in C_R \implies \exists T' \in T(c) \subset T : c' \mapsto c = T(c') \in C_R
\]

\(T\) shall act transitively on \(C_R\) in the sense that for any \(c, c' \in C_R\) there shall exist a sequence of \(T_i \in T; i = 1, 2, \ldots\) such that \(c_{i+1} = T_i c_i\) with \(c_1 = c\) and \(c_n = c'\).

As the graphs are traversed, they are transformed into unions of simpler and more complex graphs.

**Definition 6.15** It is called **monotonic** if \(\sigma \in \Sigma\) implies that any subgraph of \(\sigma\) also belongs to \(\Sigma\); it is then also monatomic.

For any given connection type \(\Sigma\), it can be extended by adding all subgraphs of the connectors it contains, as well as repetitions containing multiple bodies. Define the **monotonic extension**, denoted by \(\text{MON}(\Sigma)\), to be the extension of \(\Sigma\) generated by adding all subgraphs of the connectors it contains.

Define the **multiple extension** of \(\Sigma\), denoted by \(\text{MULT}(\Sigma)\), to be the family of graphs generated by unions with a finite and arbitrary number of repetitions, for all \(\sigma_i \in \Sigma\),

\[
\sigma_1 \cup \sigma_2 \cup \sigma_3 \cdots
\]  

(6.24)

Figure 6.7 shows various examples to illustrate transformations which give rise to \(\text{MON}(\Sigma)\) and \(\text{MULT}(\Sigma)\). Panel 1 shows a monotonic extension transformation; panel 2 shows a multiple extension transformation. A configuration for tracking is a linear graph \(\text{LINEAR}(m_1)\) of \(m_1\).

**Example 6.16 (Multiple Object Discovery)** For multiple object recognition where the number of shapes and objects are unknown it is natural to define a hierarchy or nested family of graphs corresponding to unions of multiple graphs. The generators in the higher level graphs become the single objects themselves.

Begin with the object graphs \(\sigma \in \text{OBJ} = \{\text{CYCLIC, LINEAR}\}. Define m to be the number of objects present in the scene. Extend the graph space to include \(\text{MULT}(m_1, \text{LINEAR})\) and \(\text{MULT}(m_2, \text{CYCLIC})\), the disconnected union of \(m_1, m_2\) \(\text{LINEAR}\) and \(\text{CYCLIC}\) graphs. Then a scene \(\sigma \in \text{SCENE}\) is an element of the graph set

\[
\text{SCENE} = \bigcup_{m_1 \geq 0} \text{MULT}(m_1, \text{LINEAR}) \times \bigcup_{m_2 \geq 0} \text{MULT}(m_2, \text{CYCLIC})
\]  

(6.25)

consisting of all graphs that are finite disconnected unions of graphs from \(\text{OBJ}\). The entire configuration space becomes the union of configuration spaces

\[
\mathcal{C} = \bigcup_{\sigma \in \text{SCENE}} C(\sigma).
\]
A configuration $c(MULT(m)) \in C(MULT(m))$ associated with an $m$-object graph is just a collection of single object configurations making up the scene.

The multiple object scenes are synthesized by transformation of new objects in the multiple object graph or dropping already existing objects. The graph transformations become

\[
\begin{align*}
MULT(m_1) & \rightarrow MULT(m_1 + 1) \\
MULT(m_2) & \rightarrow MULT(m_2 - 1)
\end{align*}
\]

(6.26)

where the first adds a new object from OBJ, and the second deletes.

**Example 6.17 (Parsing and Language Synthesis)** Examine standard English parsing using rules from the phrase structure grammar. Then $\Sigma = PHRASE$–FOREST and is constructed recursively as the set of all phrases in the language which terminate in a given English sentence and can be generated by parsing the English string. Parsing is the process of transforming graph types $\sigma \rightarrow \sigma'$ so that the parse pushes towards a tree rooted in a complete sentence. A set of parse trees resulting from such transformations are depicted in Figure 6.8.

The set $\Sigma = PHRASE$–FOREST is generated recursively through the family $T \in T = \bigcup_{c \in C} T(c)$ of simple moves, beginning with the starting configuration, a sequence of isolated words. This is depicted in the left panel of Figure 6.8 showing the discrete graph corresponding to the three words *The frogs jump*. This is the starting configuration during the parsing process. Simple moves are allowed that add or delete exactly one generator at the boundary $\partial \sigma$ of the connector. From $c$ only configurations in the neighborhood $N_{\text{add}}(c) = \{c' | Tc = c', T \in T(c)\} \subset C$ where

\[
N_{\text{add}}(c) = \text{set of all } c' \text{ such that } c' \text{ is obtained from } c \text{ by adding a new generator.}
\]

![Figure 6.8](image-url) Top row shows graphs from $\Sigma = PHRASE$–FOREST corresponding to a sequence of graph transformations generated during the parsing process of the sentence *The frogs jump*. Left panel shows the starting configuration; successive panels show parsing transformations from the phrase–structure context-free grammar. Bottom row shows a sequence of transformations through $\Sigma = PHRASE$–FOREST to the root node $S$ resulting in the synthesis of the sentence *The frogs jump*. 
Notice, during parsing, \( \text{external}(c) \neq \phi \); upon completion of the parse to the root node \( \text{external}(c) = \phi \).

Language synthesis involves sampling elements in the language. For this, start with the root node \( S \) as the sole configuration. Using identical grammar generators are added to external bonds, with \( N_{\text{add}}(c) \) as above. See Figure 6.8 illustrating synthesis of the string in the language *The frogs jump.*

**Example 6.18 (Computational Biology)** Learning genetic regulatory networks from microarray data using Bayesian Networks is an area of active research in the Computational Biology community. The use of DAGs is an active area of research. Friedman et al. [2000] use a Bayesian approach for the scoring of learned graphical structures based on the use of a conditional log-posterior and the choice of Dirichlet distributions for the prior. Once an appropriate scoring function has been defined, the space of all possible directed acyclic graph is explored using graph transformations in order to find the structure that best fits the data. For this, simple moves or operations resulting

---

**Figure 6.9** Top panel shows the set of elementary graph transformations over a sample DAG. Lower panel depicts a sequence of transformations resulting in an illegal structure.

---

**Figure 6.10** Gene subnetwork for mating response in *Saccharomyces cerevisiae*, from [D. Pe’er: From Gene Expression to Molecular Pathways Ph D dissertation Hebrew University]. The widths of the arcs correspond to feature confidence and they are oriented only when there is high confidence in their orientation.
in so-called neighboring structures are used (namely the addition, removal or reversal of an edge, see figure 6.9 for some examples). At each step of the learning process, only operations leading to “legal” structures (i.e. structures with no directed cycles) are permitted.

Friedman et al. [2000] and Pe’er [D. Pe’er: From Gene Expression to Molecular Pathways Ph D dissertation Hebrew University]. have applied these ideas to the study of gene regulatory networks in Yeast (Saccharomyces cerevisiae) and they have been able to correctly identify some generally agreed upon relations, as shown in figure 6.10.

6.4 The Canonical Representation of Patterns: DAGs, MRFs, Gaussian Random Fields

In the pattern theory, the first job in constructing the solution to a problem is the construction of the generators and the graphs on which they sit. This is representation. It would be nice if for each problem there were a unique definition of generators, bonds and their respective graphs. This will not be the case; there are no unique choices in general. There is, however, a canonical representation in which the generators are defined from the field variables in such a way as to accommodate the arbitrary nature of cliques in at most a pairwise interaction on the graph. This is analogous to Gaussian fields in which there is at most pairwise interactions of the cliques via the pairwise product in the potentials. The challenge is that the regular structures associated with the variability of Markov random fields on an arbitrary graph can in general have binary, ternary, arbitrary dependencies. The order depends upon the locally supported potential functions and the size of the cliques in the graph. To construct a canonical representation in pairwise interactions only, the generators must aggregate the field variables so that arbitrary dependency structure of the cliques is accommodated. This is a very general idea: essentially that the potentials of the MRFs of regular patterns can always be realized as the products of pairs of generators. To this end an explicit construction is established incorporating all order dependencies reduced to binary interactions by the introduction of the more abstract generators.

Thus far, rigid regularity has been introduced through the bond function. To accommodate variability over the regular configurations we introduce a Gibbs probability \( P(\cdot) \) on the configuration space, \( C_{\mathcal{R}} \), the regular configurations. To place probabilities on the patterns assume that the generator spaces \( \mathcal{G} \) are discrete collections making them directly connected to Gibbs distributions. The probabilities are replaced with densities once the generators move to the continuum.

**Definition 6.19** Define the local potential functions expressing interaction between generators \( \Phi_{ij} : B_i \times B_j \to \mathbb{R}^+ \) governing the probabilistic behavior of the configurations as strictly finite so that in exponential form \( e^{-\Phi(\beta, \beta')} \).

Then the Gibbs probability of a configuration \( P(C) \) restricted to \( C_{\mathcal{R}} \) for discrete generator spaces \( \mathcal{G} \) can be written as the product of potentials over pairwise interactions in the graph:

\[
P(C) = \frac{1}{Z} \prod_{e=(i,j) \in E} e^{-\Phi_{ij}(\beta_i(g_i), \beta_j(g_j))},
\]

with \( Z \) normalizing to \( C_{\mathcal{R}} \) so that \( \sum_{C \in C_{\mathcal{R}}} P(C) = 1 \).

It is often useful to highlight Eqn. 6.27 via a modification which distinguishes the coupling between generators and their marginal frequency of occurrence in the graph:

\[
\frac{1}{Z} \prod_{e=(i,j) \in E} e^{-\Phi_0(\beta_i(g_i), \beta_j(g_j))} \prod_{i=1}^{n(\sigma)} Q(g_i),
\]
7 MATRIX GROUP ACTIONS TRANSFORMING PATTERNS

ABSTRACT Thus far Pattern theory has been combinatorial constructing complex patterns by connecting simpler ones via graphs. Patterns typically occurring in nature may be extremely complex and exhibit invariances. For example, spatial patterns may live in a space where the choice of coordinate system is irrelevant; temporal patterns may exist independently of where time is counted from, and so on. For this matrix groups as transformations are introduced, these transformations often forming groups which act on the generators.

7.1 Groups Transforming Configurations

Pattern theory is transformational. One of the most fundamental transformation types for studying shape is defined via groups and group actions acting on the generators, providing a vehicle for representing the natural invariances of the real world. The fundamental role of groups as transformations on a background space what we term the generators is familiar and is at the very core of much of the geometry which is familiar to us. Even though the concept of groups did not appear in Euclid’s original axioms, congruence did. For Euclid, congruence was defined through the equivalence defined by the rigid motions carrying one figure into another. Klein [147] discovered the central role of groups in all the classical geometries (see Boothby [148]). The Erlangen program was essentially that. Geometry becomes the study of the groups which leave geometric properties invariant. The emphasis is transferred from the things being acted upon to the things doing the action. This is familiar to the Kolmogoroff complexity model. With each geometry is associated a group of transformations on the background space in which the properties of the geometry and its theorems are invariant. In Euclid’s geometry the subgroup of rigid motions of the affine group leaves distances invariant and preserves the congruence relation. Two figures are congruent if they are equal modulo an element of the rigid motion group.

7.1.1 Similarity Groups

In Pattern theory, patterns or shapes are understood through exemplars or representers of equivalence classes. Elements in the equivalence classes will often be called congruent, with the exemplar congruent to all elements of its congruence class. The formal notion of pattern is based on congruence modulo the similarity group. This section follows Artin [149] and Boothby [148].

Definition 7.1 A group is a set $S$ together with a law of composition $\circ : S \times S \to S$ which is associative and has an identity element $e \in S$ and such that every element $s \in S$ has an inverse element $s^{-1} \in S$ satisfying $s \circ s^{-1} = e$.

A subset $H$ of a group $S$, $H \subset S$, is called a subgroup if
1. it is closed: $a, b \in H \implies a \circ b \in H$,
2. it has an identity $e \in H$ and
3. the inverse is in $H$: $a \in H \implies a^{-1} \in H$.

As a notational shorthand, a group will be represented by a set, leaving the law of composition implicit, when it can be done so without ambiguity. Subgroups will be important. Direct products of groups will allow for increasing dimensions.
Definition 7.2  The direct product group

\[(S_i, \circ) = (S_1, \circ_1) \times (S_2, \circ_2) \times \cdots \times (S_n, \circ_n)\]  

(7.1)

has law of composition \[s = (s_1, \ldots, s_n), s' = (s'_1, \ldots, s'_n) \in S_1 \times S_2 \times \cdots \times S_n\] given by

\[s \circ s' = (s_1 \circ_1 s'_1, \ldots, s_n \circ_n s'_n)\]  

(7.2)

with \(\circ_i\) the law of composition for group \(S_i\).

It will be helpful to partition groups according to their subgroups and associated cosets.

Definition 7.3  A left (right) coset of subgroup \(H \subset S\) is a subset of the form

\[aH = \{ah \mid h \in H\}\]  

(7.3)

\[Ha = \{ha \mid h \in H\}\]

Example 7.4 (Familiar groups and subgroups.) Groups and subgroups are very familiar.

1. The group \((\mathbb{Z}, +), (\mathbb{R}, +), (\mathbb{C}, +)\), the integers, reals and complex numbers with addition, \(e = 0\). A subgroup of the integers is \(H = m\mathbb{Z}\), the subset of integers of multiples of \(m\), \(m\mathbb{Z} = \{n \in (\mathbb{Z}, +) : n = mk, k \in (\mathbb{Z}, +)\}\).

2. The groups \((\mathbb{R} \setminus \{0\}, \times), (\mathbb{C} \setminus \{0\}, \times)\), the nonzero real and complex numbers, with multiplication, \(e = 1\). Obvious subgroups include \(H = \{c \in \mathbb{C}^\times : |c| = 1\}\), the points on the unit circle in the complex plane, or the discrete subgroup of equally spaced points on the unit circle \(H = e^{i(2\pi k/n)}, k = 0, 1, \ldots, n - 1\).

Unlike the matrix groups we study below, all these commute giving them the special name of Abelian groups.

7.1.2 Group Actions Defining Equivalence

Congruence or sameness is an equivalence relation.

Definition 7.5  Let \(X\) be a set. An equivalence relation on \(X\) is a relation which holds between certain elements of \(X\), written as \(x \sim y\), having the properties

1. transitivity: if \(x \sim y\) and \(y \sim z\) then \(x \sim z\),

2. symmetric: if \(x \sim y\) then \(y \sim x\) and

3. reflexive: \(x \sim x\) for all \(x \in X\).

Equivalence classes, i.e. subsets of \(X\) which contain equivalent elements, form a disjoint covering, or partition of \(X\). This is an important property which we now prove.

Theorem 7.6  Let \(\sim\) be an equivalence relation on set \(X\). Then the equivalence classes partition \(X\).

Proof  For \(x \in X\), let \([x]_\sim = \{y \in X : y \sim x\}\) denote the equivalence class containing \(a\). By the reflexive property \(\forall x \in X, x \sim x\), so \(x \in [x]_\sim\). Thus, the family or collection of equivalence classes covers \(X\). We need only show that two equivalence classes are disjoint or equal. Suppose we have two equivalence classes identified by \([x]_\sim\) and \([y]_\sim\), respectively, and that there exists a shared element \(z \in [x]_\sim, z \in [y]_\sim\). By definition, \(x \sim z\) and \(y \sim z\) so by commutativity \(z \sim y\), and by transitivity, \(x \sim y\). Now suppose \(w \in [x]_\sim\), then by the definition of equivalence class, \(w \sim x\), but by transitivity \(w \sim y\), so \(w \in [y]_\sim\). Thus, \([y]_\sim\) contains \([x]_\sim\). Similarly, \([x]_\sim\) contains \([y]_\sim\). So, \([x]_\sim\) and \([y]_\sim\) are equal, i.e. they represent the same equivalence class. \(\square\)
A classic example are the EVEN,ODD integers, represented by $\bar{0}, \bar{1}$, equivalent mod2. Through the group action on sets, particularly beautiful equivalence relationships emerge.

**Definition 7.7** Let $\mathcal{S}$ be a group with group operation $\circ$, and $X$ be a set. Then define a **group action** $\Phi$ on $X$ which is a mapping $\Phi: \mathcal{S} \times X \rightarrow X$, $\Phi(s, x) = s \cdot x$, $s \in \mathcal{S}, x \in X$, with the properties that

1. if $e$ is the identity element of $\mathcal{S}$ then
   \[ \Phi(e, x) = x \quad \forall x \in X, \quad (7.4) \]
2. if $s_1, s_2 \in \mathcal{S}$ then the associative law holds according to
   \[ \Phi(s_1, \Phi(s_2, x)) = \Phi(s_1 \circ s_2, x). \quad (7.5) \]

A shorthand notation used throughout for the group action will be $\Phi(s, x) = s \cdot x, s \in \mathcal{S}, x \in X$. This is extremely familiar for matrices.

The group action $\mathcal{S}$ allows the decomposition of the set $X$ into its orbits.

**Definition 7.8** The **orbit** $\mathcal{S}x \subset X, x \in X$ of the group $\mathcal{S}$ is just the set of all images of $x$ under arbitrary group action of $s \in \mathcal{S}$:

\[ \mathcal{S}x = \{ y \in X : y = sx \text{ for some } s \in \mathcal{S} \} \quad (7.6) \]

**Theorem 7.9** Let $\mathcal{S}$ denote a group, $X$ a set and $\Phi: \mathcal{S} \times X \rightarrow X$ a group action. Define the relation

\[ x \sim y \quad \text{if } \exists s \in \mathcal{S} : \Phi(s, x) = y. \quad (7.7) \]

With the equivalence classes under this relation denoting $[x]_\mathcal{S}$, the orbits are the equivalence classes, $[x]_\mathcal{S} = \mathcal{S}x$, and the orbits $\{\mathcal{S}x\}$ of the group $\mathcal{S}$ partition $X$.

**Proof** We need to show that this is an equivalence relation. First, $x \sim x$ since $x = e \cdot x, e \in \mathcal{S}$ the identity. $x \sim y$ implies $y \sim x$ (reflexivity) according to $x \sim y \Rightarrow y = sx$ implying $x = s^{-1}y, s^{-1} \in \mathcal{S}$. Finally, $x \sim y, y \sim z \Rightarrow y = sx, z = s'y$ giving $z = (s's)x, (s's) \in \mathcal{S} \Rightarrow x \sim z$.

That the equivalence classes and orbits are equal, $[x]_\mathcal{S} = \mathcal{S}x$ follows since $x \sim y$ implies $y = sx$ and thus $y \in \mathcal{S}x$ giving $[x]_\mathcal{S} \subset \mathcal{S}x$. Conversely, $y \in \mathcal{S}x$ implies $x \sim y$ so $\mathcal{S}x \subset [x]_\mathcal{S}$.

**Definition 7.10** Define the **set of equivalence classes** as $X/\mathcal{S}$ called the orbits of the action.

In pattern theory, the set of equivalence classes $X/\mathcal{S}$ will arise often, $\mathcal{S}$ expressing the natural invariances in which the pattern is viewed.

Certain sets are essentially the same from the point of view of the group actions, i.e. if we look for example at $X/\mathcal{S}$ there is but one equivalence class $X = [x]_\mathcal{S}$. This is formalized through the notion transitive action and homogeneous spaces.

**Definition 7.11** Let $\Phi: \mathcal{S} \times X \rightarrow X$ be a group action. Then $\Phi$ is **transitive** if for all $x, y \in X$ there exists $s \in \mathcal{S}$ such that $\Phi(s, x) = y$.

In turn, $X$ is said to be a **homogeneous space** of the group $\mathcal{S}$ if there exists a transitive group action on $X$.

**Example 7.12** Equivalence relations are fundamental to many things that we do in image analysis. Examine an oriented and piecewise linear closed curve in the plane. Assume that it does not intersect itself so that Jordan’s curve theorem dictates that it divides the plane into two parts, an inside and an outside. Define $\text{set}(c) =$ inside of the curve and an equivalence relation $\sim$ by

\[ c \sim c' \text{ implies } \text{set}(c) = \text{set}(c') \]. \quad (7.8)
This defines the equivalence relation, with $[c] \sim$ the orbit representing the image.

**Example 7.13 (Null Space of a Linear Operator)** Let $A : \mathcal{H} \rightarrow \mathcal{H}$, $A$ a linear operator $A : x \in \mathcal{H} \mapsto Ax$, $\mathcal{H}$ a Hilbert space, either finite or infinite dimensional. Two elements $x, x'$ can be identified via the identification rule if $x = x' + \text{null}$, where $\text{null}$ is an element from the null space of $A$.

In particular, let $A : \mathbb{R}^n \rightarrow \mathbb{R}^n$, with $A = \sum_{i=1}^{m} \alpha_i \phi_i(\phi_i, \cdot)_{\mathbb{R}^n}$, with $m < n$ and $\{\phi_i\}_{i=1}^{n}$ a C.O.N. basis for $\mathbb{R}^n$. Then $x \sim x'$ if and only if $Ax = Ax'$. Divide $\mathbb{R}^n$ into the null space of the operator and its orthogonal complement, $\mathbb{R}^n = N \oplus N^\perp$. To construct a disjoint partition via equivalence classes define $x^\perp = \sum_{i=1}^{m} \phi_i(\phi_i, x)_{\mathbb{R}^n}$, and $x^\perp$ is the projection onto the orthogonal complement of the null space, $x^\perp \in N^\perp$. The equivalence classes or images $[x] \sim$ become

$$[x] \sim = x^\perp \oplus N = \left\{ x' \in \mathbb{R}^n : x' = x^\perp + \sum_{i=m+1}^{n} \alpha_i \phi_i, (\alpha_{m+1}, \ldots, \alpha_{n}) \in \mathbb{R}^{n-m} \right\}.$$

### 7.1.3 Groups Actions on Generators and Deformable Templates

The patterns and shapes will be represented via the *exemplars* or *generators*; the natural invariances are expressed through the groups acting on the generators. Equivalence being defined through the orbits and partition of the pattern space into the orbits.

For this, let the similarity group be of some fixed dimension acting on the generator space. We shall use the shorthand notation to define the group action $\Phi(s, g) = sg$.

**Definition 7.14** The **group action on a single generator** $\Phi : S \times G \rightarrow G$ is defined as

$$\Phi(s, \Phi(s', g)) = \Phi(s \circ s', g) = (s \circ s', g).$$  \hspace{1cm} (7.9)

### 7.2 The Matrix Groups

Shapes and structures will be studied using the low-dimensional matrix groups translations, rotations, rigid motions of translations and rotations acting on the finite dimensional $X = \mathbb{R}^d$ background spaces.

#### 7.2.1 Linear Matrix and Affine Groups of Transformation

For the geometry of shape, the finite dimensional groups and their subgroups generated from the group of matrices, the *generalized linear group* is used throughout.

Define the matrix groups explicitly as follows.
Definition 7.15 (Generalized Linear Group) The \( d \times d \) general linear group, denoted as \( \text{GL}(d) \), is the group of all \( d \times d \) matrices

\[
\text{GL}(d) = \{ d \times d \text{ real matrices } A \text{ with } \det A \neq 0 \}, \tag{7.10}
\]

with non-zero determinant (invertible) and with law of composition matrix multiplication,

\[
A \circ B = AB = \left( \sum_j A_{ij} B_{jk} \right), \tag{7.11}
\]

with the identity element \( I = \text{diag}[1, \ldots, 1] \).

That this is a group follows from the fact that the identity \( I \in \text{GL}(d) \), the product of two matrices, is in the group \( A \circ B = AB \in \text{GL}(d) \) and the inverse is in the group as well \( A^{-1} \in \text{GL}(d) \).

Subgroups of \( \text{GL}(d) \) are used as well.

1. Define the special linear group \( \text{SL}(d) \subset \text{GL}(d) \) to be the subgroup of volume preserving transformations:

\[
\text{SL}(d) = \{ A \in \text{GL}(d) : \det A = 1 \}. \tag{7.12}
\]

2. Define the orthogonal group \( \text{O}(d) \subset \text{GL}(d) \) to be the orthogonal subgroup of matrices and \( \text{SO}(d) \subset \text{O}(d) \subset \text{GL}(d) \) to be the special orthogonal subgroup with determinant 1:

\[
\text{O}(d) = \{ A \in \text{GL}(d) : A^* A = I \}, \tag{7.13}
\]

\[
\text{SO}(d) = \{ A \in \text{O}(d) : \det A = 1 \} = \text{O}(d) \cap \text{SL}(d). \tag{7.14}
\]

3. Define the uniform scale group \( \text{US}(d) \subset \text{GL}(d) \) of diagonal matrices:

\[
\text{US}(d) = \{ A \in \text{GL}(d) : A = \rho I, \rho > 0 \}. \tag{7.15}
\]

Notice, the group operation for \( \text{GL}(d) \) does not commute. There are various groups which are generated as products of the subgroups.

Definition 7.16 The affine group \( \text{A}(d) \) is the semi-direct product of groups \( \text{GL}(d) \otimes \mathbb{R}^d \) with elements \( \{(A, a) : A \in \text{GL}(d), a \in \mathbb{R}^d \} \) and law of composition semi-direct product

\[
\text{A}(d) = \text{GL}(d) \otimes \mathbb{R}^d, \quad \text{with} \quad (A, a) \circ (B, b) = (AB, Ab + a). \tag{7.16}
\]

The Euclidean and special Euclidean groups \( \text{E}(d), \text{SE}(d) \), respectively, are the subgroups of the Affine Group consisting of the rigid motions generated from orthogonal and special orthogonal \( \text{O}(d), \text{SO}(d) \) motions with group operation, the semi-direct product:

\[
\text{E}(d) = \text{O}(d) \otimes \mathbb{R}^d, \quad \text{SE}(d) = \text{SO}(d) \otimes \mathbb{R}^d. \tag{7.17}
\]

The similitudes are given by the direct product of uniform scale with orthogonal motions

\[
\text{Sim}(d) = \text{US}(d) \times \text{SO}(d). \tag{7.18}
\]
7.2.2 Matrix groups acting on $\mathbb{R}^d$

When the matrix groups act on the infinite background space $X = \mathbb{R}^d$ according to the usual convention of identifying points $x \in X = \mathbb{R}^d$ with column vectors, the action is matrix multiplication on the vectors.

**Definition 7.17** Let $X$ be the background space and the affine group $A \in \text{GL}(d)$ acts on the background space according to

$$
\Phi(A, x) = Ax = \left( \sum_j A_1 x_j \right) \in X.
$$

(7.19)

Let $(A, a) \in S \subset A(d)$, then

$$
\Phi((A, a), x) = Ax + a.
$$

(7.20)

For homogeneous coordinates, then represent

$$
\tilde{A} = \begin{pmatrix} A & a \\ 0 & 1 \end{pmatrix} \in \tilde{A}, \quad A \in \text{GL}(d), \ a \in \mathbb{R}^d, \quad \tilde{x} = \begin{pmatrix} x \\ 1 \end{pmatrix},
$$

(7.21)

with group action $\Phi(\tilde{A}, \tilde{x}) = \tilde{A}\tilde{x}$.

We emphasize that the semi-direct product of Eqn. (7.16) matches $\Phi((A, a), x)$ of Eqn. 7.20 a group action. We leave it to the reader to verify this.

**Example 7.18 (Polar Decomposition)** Polar decomposition is useful for clarifying the degrees of freedom inherent in the class of operators. Any invertible matrix $A \in \text{GL}(d)$ can be written as

$$
A = PO = \underbrace{O^* \text{diag}(\lambda_1, \ldots, \lambda_d) O'}_{P} O,
$$

(7.22)

$P$ being a positive definite symmetric matrix, with $O, O' \in \text{O}(d)$ the orthogonal matrices. With that said, rewrite elements of $\text{GL}(2)$ to have 4 parameters, noticing $O, O' \in \text{SO}(2)$ have one each, with two for the diagonal entries. Similarly elements of $\text{GL}(3)$ have 9 free parameters, 3 each for the orthogonal groups and 3 diagonal entries.

**Example 7.19 (Tracking via Euclidean motions)** To accommodate arbitrary position and pose of objects appearing in a scene introduce the subgroups of the generalized linear group $\text{GL}(3) : X \leftrightarrow X$. The generators $G$ are the CAD models of various types as depicted in Figure 7.1 showing 3D renderings of sample templates. In this case each template consists of a set of polygonal patches covering the surface, the material description (texture and reflectivity), and surface colors.

The Euclidean group $E$ including rigid translation and rotation operate on the generators 2D surface manifolds in $\mathbb{R}^2$. The Euclidean group of rigid motions denoted $E(n) = \text{O}(n) \otimes \mathbb{R}^n$, where $O \in \text{O}(n)$ are $n \times n$ orthogonal matrices; the group action on $X = \mathbb{R}^n$ is $\Phi((O, a), \cdot) : \text{O}(n) \otimes \mathbb{R}^n \to \mathbb{R}^n$ is $\Phi((O, a), x) = Ox + a$, with the law of composition given by the semi-direct product: That Euclidean distance is preserved $d(\Phi((O, a), x), \Phi((O, a), y)) = d(x, y)$ where $d(x, y) = \sum_{i=1}^n |x_i - y_i|^2$ for rotation around the origin followed by translation is clear, hence the name rigid motions.

**Example 7.20 (Flip Group for Symmetry)** Biological shapes exhibit often symmetries. The right panel of Figure 7.1 illustrates symmetry seen in the human brain. The cosets of
Figure 7.1 The panels 1–3 show various CAD models for objects. Panel 4 shows the human brain section depicting symmetry.

The orthogonal group defined by the flips which are appropriate for studying symmetry, $\text{D}(2,3) = \{R, I\} \subset \text{O}(2,3)$, given by the $2 \times 2, 3 \times 3$ matrices

$$
\text{D} = \begin{cases}
I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, & R = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \\
I = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, & R = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix},
\end{cases}
$$

(7.23)

with the cosets $I[\text{SO}(2,3), R\text{SO}(2)(3)]$.

Example 7.21 (Similar and Congruent Triangles) Congruence: Let $S = \text{E}(2)$ be the set of Euclidean rigid motions in the plane, $X$ the set of points, lines, or triangles in the plane. For $X = \text{TRIANGLES}$, then $[\triangle_0]_S$ is the subset of $\text{TRIANGLES}$ congruent to $\triangle$. Let a nondegenerate $\triangle = (v_1, v_2, v_3) \in \mathbb{R}^6$ be identified with a $2 \times 3$ matrix constructed from the vertices, the triangle nondegenerate so that $\triangle$ is a rank 2 matrix. Position the triangle $\triangle_0$ at the origin so that $v_1 = (0, 0) \in \mathbb{R}^2$.

Let $G = \text{TRIANGLES} \subset \mathbb{R}^6$ be generated by the affine motions on $\triangle_0$:

$$
\text{TRIANGLES} = \{(A, a)\triangle_0, (A, a) \in A(2)\},
$$

(7.25)

where the group action applied to the $2 \times 3$ matrix generators becomes

$$(A, a)\triangle = A\triangle + \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}.$$

(7.26)

Define equivalence classes by the orbits of $E, [\triangle_0]_E$. Then two triangles $\triangle_1 \sim \triangle_2$ if both are in the same orbit: $\triangle_2 = s\triangle_1$ for some $s \in E$. Since orbits form disjoint partitions we have essentially established the Euclidean partition of $\text{TRIANGLES}$ into congruence classes.

Notice in this example a triangle $\triangle$ is not a picture but rather a point in the manifold of $2 \times 3$ matrices.

Example 7.22 (Similar Triangles) Now return to Euclid’s more general notion of similarity and notice that the subgroups are normal allowing us to use quotient group ideas. Examine the orbit of $\triangle_0$ under the scale-rotation group matrices of the type
$s = \left(\begin{array}{cc}
u & u \\ -v & u \end{array}\right), (u, v) \neq (0, 0)$. Then, both $\text{SO}(2)$, $\text{US}(2)$ are normal subgroups implying, for example, $\text{SR}(2)/\text{SO}(2)$ is the quotient group. This is the union of all congruence classes (left cosets of $\text{SO}(2)$) in the sense of Euclid:

$$[\Delta_0]_{\text{SR}(2)/\text{SO}(2)} = \{s\Delta_0 : s \in \rho\text{SO}(2), \rho > 0\}. \quad (7.27)$$

These are the congruence classes of triangles rooted at the origin. Each class is similar to each other in the extended sense of Euclid but not congruent.

**Example 7.23 (Target and Object Symmetry)** Examine target and object symmetry, the generators $G$ the targets. Man-made objects are characterized by well defined shapes with symmetries. The standard notion of symmetry of an object is studied through the subgroup for the object $\alpha \in \mathcal{A}$,

$$S(\alpha) = \{s \in S : sg^\alpha = g^\alpha\} \subset S.$$  

Due to this equivalence, the inference set has to be reduced to the quotient space $S/S(\alpha)$. For this define the identification rule corresponding to the set of symmetries:

$$[g]_\sim = \{g' : g' = sg, s \in S(g)\}. \quad (7.28)$$

Let $S = \text{SE}(3)$ the special Euclidean group, for $\alpha = \text{cylinder}$, then $S(\text{cylinder}) = \text{SO}(2)$ and the inference is performed over the 2-sphere, $\text{SO}(3)/\text{SO}(2) = S^2$.

If on the other hand more structure is added to the cylinder such as for a missile which also has four tail fins separated by the angle $90^\circ$ then the relative symmetry set instead includes the discrete $4$-torus.

### 7.3 Transformations Constructed from Products of Groups

Since complex patterns are often constructed by combining simpler ones, it will be necessary to work with the products of groups, each one applied to a single generator in the scene, thereby representing the variability of multiple subconfigurations. To extend the domain of the group transformations to the full configurations combining multiple generators, define the product group acting on the multiple generators making up the configurations. We now generalize the transformations so that they can be applied locally throughout the space required. The product groups are extended to act on the regular configurations $c(\sigma) = \sigma(g_1, \ldots, g_n) \in \mathcal{C}_R$ in the obvious way. The graph type $\sigma$ will remain unaffected by the transformations; however, the configuration may not be regular after the similarities $S = S^n_U$ act on them.

For this, let the similarity group be of dimension $d$ acting on the $i$th generator $S_0 : \mathcal{G}_0 \leftrightarrow \mathcal{G}_0$, with the product $S = S_0 \times S_0 \times \cdots$ of dimension $d$ times the number of generators.

**Definition 7.24** The group action on multiple generators $\Phi^n : S^n_0 \times \mathcal{G}^n \to \mathcal{G}^n$ becomes

$$\Phi^n((s_1, s_2, \ldots), \Phi^n((s'_1, s'_2, \ldots), (g_1, g_2, \ldots))) = ((s_1 \circ s'_1)g_1, (s_2 \circ s'_2)g_2, \ldots). \quad (7.29)$$

**Define the action on the configurations** $S_0^n : \mathcal{C}_R \to \mathcal{C}$ as

$$s\sigma(g_1, g_2, \ldots) = \sigma(s_1g_1, s_2g_2, \ldots). \quad (7.30)$$

Denote a particular regular configuration within the configuration space as the **template**, one fixed element in the orbit:

$$c^0 = \sigma(g_1^0, g_2^0, \ldots, g_n^0). \quad (7.31)$$

The **deformable template** becomes the orbit $[c^0]_S$.  

---
The significant extension being made here is to accommodate the glueing together of the structured generating sets, with the transformations acting locally upon them. Thus far the group transformations have acted globally on the entire generating background space. This presents several significant difficulties for the pattern theory; clearly with multiple groups acting regular configurations may not be regular. Examine the application to the representation of a semi-rigid chair swiveling around its base. Imagine that a translation and axis-fixed rotation group is applied to the chair, but equality can hold: should the cartesian product E(2) × E(2) unconstrained be applied to the rigid substructure? Apparently not, otherwise the chair would tear apart. It is natural to wonder why this difficulty does not arise in the application of single group transformations. For the matrix groups acting globally local structure is mapped with its topology preserved. This is the property of the matrix groups when viewed as smooth 1–1 and onto transformations (diffeomorphisms). We will return to this later.

It will be necessary to define the subspace of transformations for which the regular configurations stay regular.

**Definition 7.25** Then the regularity constrained subset of similarities $S_R \subset S = S^n_0$ become

$$S_R = \{s \in S^n_0 : sc \in C_R, \forall c \in C_R\} \quad (7.32)$$

$$= \cap_{c \in C_R} S_R(c) \quad \text{where} \quad S_R(c) = \{s \in S^n_0 : sc \in C_R\}. \quad (7.33)$$

From the configurations which have acted upon it is natural to denote a particular regular configuration within the configuration space of the template.

**Definition 7.26** The template is one fixed element in the orbit:

$$e^0 = \sigma(g^0_1, g^0_2, \ldots, g^0_n). \quad (7.34)$$

The deformable template is the full orbit $[e^0]_S$.

The template expresses typical structure, the similarities variability around it. These subsets play an important role in pattern theory, and in general they will not be subgroups of $S$! Typically they are of lower dimension than that of $S$, but equality can hold:

$$\dim(S_R(c)), \dim(S_R) \leq \dim(S). \quad (7.35)$$

Although they are not generally a subgroup, $S_R \subset S : C_R \rightarrow C_R$ is a semi-group with identity. Clearly $e \in S_R(c)$ for all $c$. If $s, s' \in S_R$ then this implies

$$(s \circ s')c = s(s'c) = sc', \quad c' \in C_R \quad (7.36)$$

and since $s \in S_R$ then $sc' \in C_R$. It is sometimes the case that $S_R$ is a subgroup.

**Example 7.27 (TANKS)** Examine the almost rigid body case corresponding to the representation of tanks with movable turrets atop tractor assemblies. Let $G = \{\text{tractors, turrets}\}$ the set of tractors and turrets at all possible orientations and positions, $G^0 = \{0 \times g^0_2\} \subset G$ the tractors and turrets at the origin at $0^\circ$ orientation. The graph type DIMER consists of bonds between the mounting support of the tractor and the pivot point of the turret: the configurations $c = \text{DIMER}(g^1_1, g^1_2)$ are regular if $\beta_{\text{out}}(g_1) = \beta_{\text{in}}(g_2)$ meaning the turret is mounted on the tractors. The basic similarity group $E(2) = SO(2) \otimes \mathbb{R}^2 : G \leftrightarrow G$ rotates the tractors and turrets around the body reference frame, and translates. The full product similarity $S = E(2)^2$ does not generate regular configurations; the similarity constrained subset $S_R = E(2) \times SO(2)$, with its action rotating the turret around the center point of contact of the tractor according to

$$s = ((O_1, a_1), (O_2, a_1)) \in S_R : \text{DIMER}(g^1_1, g^1_2) \leftrightarrow \text{DIMER}((O_1, a_1)g_1, (O_2, a_1)g_2).$$
Example 7.28 (Unclosed Contours: SNAKES and ROADS on the Plane) Examine the representation such as in HANDS [150] for unclosed ROAD contours in which the generators are vectors \( G = \mathbb{R}^2 \), essentially directed line segments from the origin, \( s^0 = \left( \begin{array}{c} \delta^0_1 \\ \delta^0_2 \\ \delta^1_0 \\ \delta^1_2 \end{array} \right) \), with graph type \( \sigma = \text{LINEAR}(n) \) denoting the order of generators. The polygonal representations are generated by adding the generators sequentially; hence the ordering is required in the configuration. To generate \( n \)-length curves in the plane, attach the basic similarities: rotations \( R = \text{SO}(2) \) rotating each of the generators:

\[
O(\theta) : g_k \mapsto \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} \delta^1_k \\ \delta^2_k \end{pmatrix}.
\]

The regularity constrained set is \( \mathcal{S}_R = \text{SO}(2)^n \), and the configuration space \( \mathcal{C}_R \) is the space of all \( n \)-length piece-wise linear curves rooted at the origin. This is a homogeneous space under the group action \( \mathcal{S}_R \) on \( \mathcal{C}_R \); the identity are copies of the 0-degree rotations, with law of composition

\[
s \circ s' = (\theta_1 + \theta'_1, \ldots, \theta_n + \theta'_n) \in \mathcal{S}_R.
\]

To generate all \( n \)-length, piece-wise linear curves, add the global translation group \( \mathcal{S}_R = \text{SO}(2)^n \times \mathbb{R}^2 \). An obvious template is a straight line along the \( x \)-axis of length \( n \) so that the configuration \( c^0 = \text{LINEAR}(\delta^0_1, \delta^0_2, \ldots) \), \( \delta^0_k = \left( \begin{array}{c} 0 \\ 1 \end{array} \right) \).

Example 7.29 (Closed Contours) Adding the closure condition, the regularity constrained product transformation \( \mathcal{S}_R \) on \( \mathcal{C}_R \) is no longer a subgroup, just a linear manifold. Take as generators \( G = \mathbb{R}^2 \) chords from the circle \( \delta^0_k = \left( \begin{array}{c} \cos(2\pi k/n) - \cos(2\pi (k-1)/n) \\ \sin(2\pi k/n) - \sin(2\pi (k-1)/n) \end{array} \right) \); with these similarity groups \( \mathcal{S}_0 = \text{US}(2) \times \text{O}(2) \) the scales/rotations \( s = \left( \begin{array}{cc} u_1 & u_2 \\ -u_2 & u_1 \end{array} \right) \), where \( u_1 = \rho \cos \phi, u_2 = \rho \sin \phi \), \( u_1, u_2 \in \mathbb{R}^2 \), \( \rho \) the scale parameters, and \( \phi \) the rotation parameter. The full space of generators is generated via the transitive action \( s \in \mathcal{S}_0 : \delta^0 \mapsto g = sg^0 \) according to

\[
\delta^0_k = \left( \begin{array}{c} \delta^1_k \\ \delta^2_k \end{array} \right) = \left( \begin{array}{cc} u_{1k} & u_{2k} \\ -u_{2k} & u_{1k} \end{array} \right) \left( \begin{array}{c} \delta^0_1 \\ \delta^0_2 \end{array} \right).
\]

The full space of transformations (regular and irregular configurations) becomes \( \mathcal{S} = \mathcal{S}_0^u = (\text{US}(2) \times \text{SO}(2))^2n \subseteq \mathbb{R}^{2n} \). Figure 7.2 shows the circular closed contour templates and an example deformation via the scale-rotation groups.

The template becomes \( c^0 = (\delta^0_1, \delta^0_2, \ldots, \delta^0_n) \). The closure condition means the endpoint of the \( n \)th generator must be 0 giving \( \sum_{i=1}^n \delta_i = 0 \) implying \( \mathcal{S}_R = \mathbb{R}^{2n-2} \subseteq \mathbb{R}^{2n} \). \( \mathcal{S}_0^u \) and \( \mathcal{S}_R \) is not a subgroup but is a submanifold! The loss of two-dimensions from the closure condition is a linear manifold constraint given by the discrete Fourier transform (DFT) condition, for \( j = \sqrt{-1} \), then

\[
\sum_{k=1}^n \left( \begin{array}{cc} u_{1k} & -u_{2k} \\ u_{2k} & u_{1k} \end{array} \right) \left( \begin{array}{c} \cos(2\pi k/n) - \cos(2\pi (k-1)/n) \\ \sin(2\pi k/n) - \sin(2\pi (k-1)/n) \end{array} \right) = \sum_{k=1}^n (u_{1k} - ju_{2k}) (e^{i(2\pi k/n)} - e^{i(2\pi (k-1)/n)}).
\]
Figure 7.2 Showing the circular closed contour template and its deformation via the scale-rotation groups.

This implies the highest frequency DFT component is zero:

\[
\sum_{k=1}^{n} (u_{1k} - ju_{2k}) e^{-j(2\pi k(n-1)/n)} = e^{-j(2\pi /n)} \sum_{k=1}^{n} (u_{1k} - ju_{2k}) e^{-j(2\pi k(n-1)/n)} = 0.
\]

(7.39)

This \( S_R \) is not a subgroup acting on \( C_R \), hence \( s \circ s' \notin S(c, R) \) since the closure condition is a linear constraint on the scale/rotation elements according to Eqn. 7.39 which is not satisfied. Addition of the vector of similarities would maintain closure since this would maintain the highest frequency DFT coefficient being zero. However, composition is not addition.

### 7.4 Random Regularity on the Similarities

A fundamental interplay in pattern theory is the interaction between probability laws on the generators and probability laws on the transformations or similarities. The probability law on transformations may be induced directly via the probabilities on the similarities \( S \) and the regularity constrained similarities \( S_R \). We are reminded of the work by Kolmogoroff in defining Kolmogoroff complexity, transferring the focus from the program (template) to the input to the program (transformation).

Assume that the configuration spaces can be identified with Euclidean vector spaces, \( C(\sigma) = \mathbb{R}^m \), with \( S: \mathbb{R}^m \rightarrow \mathbb{R}^m \) which is a bijection. Then the two probability densities are related via the standard transformation of probability formulas.

Since the roles of the generators and the similarities are so often interchanged in so many of the examples in pattern theory, it is instructive to be more explicit about how these densities are related. For this, assume there is a bijection between generators and similarities, so that the equation \( g = s g^0 \), \( g, g^0 \in G \) has a unique solution in \( s \in S \). Then the density on the similarities induces a density on the configurations.

**Theorem 7.30** Assume \( c = sc^0 \in \mathbb{R}^n \) with a bijection between configurations and similarities. The two densities on the similarities \( p_S(s) \) on \( S \) and configurations \( p_C(c) \) on \( C(\sigma) \) are related via the standard probability transformation formula with \( m(ds) \) the measure on similarities:

\[
p_C(c|_{sc^0}) dc = p_S(s) |\det D_{sc^0}| m(ds),
\]

(7.40)

with \( D_{sc^0} \) the Jacobian matrix of the transformation \( c = sc^0 \).

Here are several examples.
7.4 Random Regularity on the Similarities

Example 7.31 (Closed Contours with Directed Arcs as Generators) Examine closed contours, the graph \( \sigma = \text{CYCLIC}(n) \) with generators the directed arcs \( g_k = (g_{1k}, g_{2k}) \in \mathcal{G} = \mathbb{R}^4 \), \( g_{1k} = \text{start point}, g_{2k} = \text{end point} \), with bond values \( \beta_{\text{in}}(g) = g_1, \beta_{\text{out}}(g) = g_2 \) start and end points of the generators in the plane and bond function

\[
\beta_{\text{out}}(g_k) = \beta_{\text{in}}(g_{k+1}) \quad \text{so that} \quad g_{2k} = g_{1k+1}.
\]

The start point of the \( k+1 \)st arc equals the endpoint of the \( k \)th arc. For the graph type CYCLIC, unlike LINEAR, the in-bond of the first generator \( g_1 \) is connected to the out-bond of the last generator \( g_n \) as well.

Apply the similarity of scales-rotations-translations \( S_0 \) extending the template generators \( S_0 : \mathcal{G}^0 \rightarrow \mathcal{G} = \mathbb{R}^4 \) rotating and scaling the generators around their origin, followed by translation, so that \( s = (A, a) \in S_0 : g_k' \mapsto g_k = s_k g_k' \),

\[
g_k = s_k g_k' = (g_{1k}' + a_k), \quad \left( \begin{array}{cc}
  u_{1k} & u_{2k} \\
  -u_{2k} & u_{1k}
\end{array} \right) (g_{1k}' - g_{1k}) + g_{1k}' + a_k.
\]

(7.41)

This is a transitive action on \( \mathbb{R}^4 \). Choose as generators the chords from the circle and similarities so that

\[
g_k^0 = \left[ \begin{array}{cc}
  \cos(2\pi (k-1)/n) \\
  \sin(2\pi (k-1)/n)
\end{array} \right], \quad s_k^0 = \left( \begin{array}{cc}
  \cos(2\pi (k-1)/n) \\
  \sin(2\pi (k-1)/n)
\end{array} \right) ;
\]

then the identity \( I = \left( \begin{array}{cc}
  1 & 0 \\
  0 & 1
\end{array} \right) \) and the template is a connected circle \( c^0 = \text{CYCLIC}(s_1^0, s_2^0, \ldots, s_n^0) \). The graph type CYCLIC associated with the closure condition implies \( g_1 \) given the first translation, all others are fixed giving the set of \( 2n \) rotation/scale elements implying \( S_\mathcal{R} = \mathbb{R}^2 \times \mathbb{R}^{2n-2} \subset S = (\text{US}(2) \times \text{SO}(2)) \times \mathbb{R}^2 \)

which is not a group. The loss of two-dimensions follows from the closure condition on the sum of the generators, which is a linear manifold constraint.

The probability on the transformations induces the probability on the generators. Define the generators as \( 4 \times 4 \) matrices operating on the similarity according to

\[
g_k = s_k g_k^0 = \left( \begin{array}{cccc}
  s^0_{21,k} & s^0_{22,k} & 1 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 \\
\end{array} \right) \left( \begin{array}{c}
  u_{1k} \\
  u_{2k} \\
  a_{1k} \\
  a_{2k}
\end{array} \right) ;
\]

(7.42)

\( p_\mathcal{S}(s_1, s_2, \ldots, s_n)|m(ds) \) induces a probability on the generators according to

\[
p(g_1, \ldots, g_n) = p_\mathcal{S}(s_1 g_1^0 = g_1, \ldots, s_n g_n^0 = g_n) \prod_{k=1}^n |\det D_{g_k} g_k| |ds_k| .
\]

(7.43)

Since

\[
|\det D_{g_k} g_k| = \det \left| \begin{array}{cccc}
  s^0_{21,k} & s^0_{22,k} & 1 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 \\
\end{array} \right| = (s^0_{21,k})^2 + (s^0_{22,k})^2,
\]

(7.44)

then the determinant of the Jacobian matrix is the product of the lengths of the generators squared, \( \prod_{k=1}^n (s^0_{1,k})^2 + (s^0_{2,k})^2 \).