Fakultät für Informatik
der Technischen Universität München

Diplomarbeit in Informatik

Computergestütztes Zeichnen in den Symmetriegruppen der euklidischen Ebene

Martin von Gagern

Fakultät für Informatik
der Technischen Universität München

Diplomarbeit in Informatik

Computergestütztes Zeichnen in den Symmetriegruppen der euklidischen Ebene

Computer-aided drawing in the symmetry groups of the euclidean plane

Bearbeiter: Martin Franz Stephan Freiherr von Gagern
Aufgabensteller: Professor Tobias Nipkow & Professor Jürgen Richter-Gebert
Betreuer: Professor Jürgen Richter-Gebert
Abgabedatum: 15. Februar 2008

Erklärung

Ich versichere, dass ich diese Diplomarbeit selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Garching, den 15. Februar 2008

Martin v. Gagern

Zusammenfassung

Diese Diplomarbeit behandelt die theoretischen und praktischen Grundlagen eines Computerprogramms zum Zeichnen von Ornamenten in den Symmetriegruppen der euklidischen Ebene. Es werden Konzepte zur Berechnung von Orbits sowie zur effizienten Darstellung von Mustern mittels überlappender Puffer erläutert. Die Verwendung von Voronoi-Polygonen als vom Benutzer verformbare Fundamentalbereiche wird motiviert und ein Verfahren zu deren Berechnung beschrieben. Es folgt eine Diskussion unterschiedlicher Ansätze zum Export von Ornamenten in verschiedenen Formaten. Außerdem werden Schritte zur Erkennung der Symmetriegruppen von periodischen Mustern aus externem Bildmaterial dargestellt, basierend auf Autokorrelation sowie der Berechnung von Dominanzgebieten.

Abstract

This diploma thesis deals with theoretical and practical foundations of a computer program for drawing ornaments in the symmetry groups of the euclidean plane. Concepts for orbit calculation and for efficient display of patterns using overlapping buffers are explained. The use of voronoi polygons as user-modifiable fundamental domains is presented and a method for their calculation described. Different approaches for exporting to various formats are discussed. Furthermore, steps for the recognition of symmetry groups of patterns from external images are demonstrated, based on autocorrelation and the calculation of areas of dominance.

Inhaltsverzeichnis

I Einführung **6**

1 Einleitung **6**
1.1 Benutzersicht . . . 6
1.2 Entwicklung . . . 7
1.3 Vergleiche . . . 9

2 Mathematische Grundlagen **10**
2.1 Definitionen . . . 10
2.2 Formulierung von Transformationen . . . 14

3 Klassifikation von Symmetriegruppen **16**
3.1 Rosettengruppen . . . 16
3.2 Friesgruppen . . . 18
3.3 Kristallographische Gruppen . . . 18

II Konzepte der Implementierung **21**

4 Berechnung von Orbits **21**
4.1 Transformation in Hilfskoordinaten . . . 21
4.2 Wechsel der Translationszelle . . . 22
4.3 Anwenden innerer Symmetrien . . . 23
4.4 Anwenden der Translationen . . . 23
4.5 Rücktransformation . . . 23
4.6 Zusammenfassung . . . 23

5 Darstellung auf dem Bildschirm **26**
5.1 Verwendung rechteckiger Puffer . . . 26
5.2 Bestimmung von Überdeckungen . . . 26

6 Bestimmung von Fundamentalbereichen **29**
6.1 Verschiedene Fundamentalbereiche . . . 29
6.2 Voronoi-Polygone als Fundamentalbereiche . . . 31
6.3 Algorithmische Bestimmung der Voronoi-Polygone . . . 32
6.3.1 Schranke für den Einflussbereich . . . 33
6.3.2 Berechnung der Polygone . . . 42

7 Export der Zeichnungen **43**
7.1 Vektorformate . . . 43
7.1.1 Portable Document Format (PDF) . . . 43
7.1.2 Scaleable Vector Graphics (SVG) . . . 44
7.1.3 PostScript . . . 44
7.2 Rasterformate . . . 45
7.3 Unterschiedliche Export-Medien . . . 46

8 Automatische Mustererkennung **46**
8.1 Autokorrelation 47
8.2 Peaks 50
8.3 Gittervektoren 53
8.4 Drehzentren 56
8.5 Spiegelungen 59
8.6 Hintergrundbild 62

III Anwendung und Ausblick **63**

9 Einsatzmöglichkeiten **63**
9.1 Einsatz in der Lehre 63
9.1.1 Grundschule 63
9.1.2 Sekundarstufe 63
9.1.3 Hochschule 63
9.2 Professioneller Einsatz 63
9.3 Heimgebrauch 64
9.4 Ausstellungen 64

10 Ausblick **64**
10.1 Weiterentwicklung 64
10.2 Folgeprojekte 65

11 Verzeichnisse **66**
11.1 Literatur 66
11.2 Definitionen 67
11.3 Abbildungsverzeichnis 68
11.4 Tabellenverzeichnis 68
11.5 Index 69

IV Anhang **72**

12 Beispielornamente **72**
12.1 Rosettengruppen 72
12.2 Friesgruppen 74
12.3 Kristallographische Gruppen 76

Teil I
Einführung

1 Einleitung

Thema dieser Diplomarbeit ist die Entwicklung eines Computerprogramms zum effektiven, intuitiven und interaktiven Zeichnen symmetrischer Ornamente in der euklidischen Ebene (also der ganz gewöhnlichen Zeichenebene, wie man sie aus seiner Schulzeit kennt). Dabei werden zu jeder vom Benutzer gezeichneten Linie eine Menge von Kopien dieser Linie, der sogenannte Orbit, vom Programm ergänzt. Die Symmetriegruppe, die die Struktur des Ornaments festlegt und somit entscheidet, welche Kopien erzeugt werden, ist einstellbar.

Die vorliegende Arbeit soll einige Konzepte der Implementierung genauer erklären. Dabei geht es zum einen um eine effiziente Umsetzung der Grundfunktionalität, also der Berechnung und Darstellung der vom Benutzer gezeichneten Muster, zum anderen um die Realisierung ausgewählter Features.

1.1 Benutzersicht

Bevor näher auf die Geschichte und interne Funktionsweise des im Rahmen dieser Arbeit geschriebenen Programms samt der dazu erforderlichen mathematischen Grundlagen eingegangen wird,

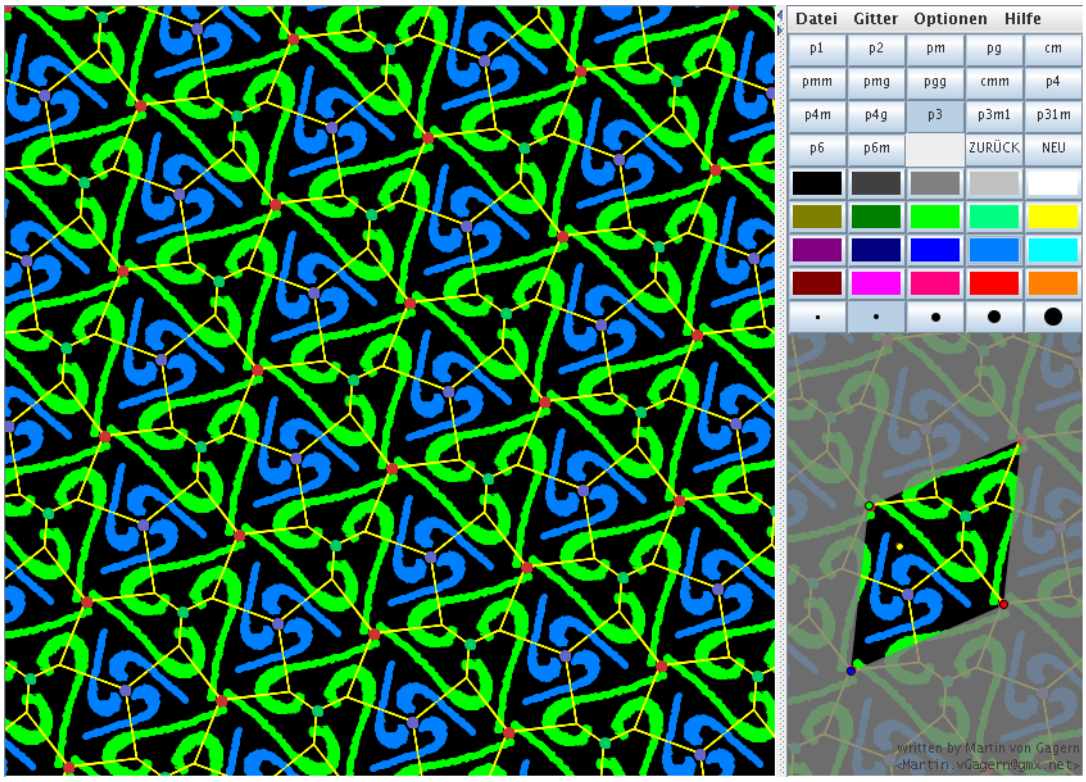


Abbildung 1: Screenshot der Anwendung

soll hier ein kurzer Überblick gegeben werden, wie sich das Programm dem Benutzer darstellt.

Das Programm bietet dem Benutzer einen dreigeteilten Anwendungsbereich, wie in Abbildung 1 auf der vorherigen Seite ersichtlich. Links befindet sich die Zeichenfläche, in der einfach mit der Maus oder einem Grafiktablett gezeichnet werden kann. Rechts unten ist eine einzelne Translationszelle extrahiert, also ein Bereich, dessen verschobene Kopien das gesamte Muster erzeugen. Durch farbige Kontrollpunkte in dieser Ansicht können Position, Größe und Form der Translationszelle eingestellt werden. Ist die Darstellung von Fundamentalzellen (die in Abschnitt 6 auf Seite 29 eingehend behandelt wird) aktiviert und deren Form veränderbar, so wird in diesem Teil der Ansicht ein Kontrollpunkt zur Beeinflussung dieser Form dargestellt.

Rechts oben findet sich eine Sammlung von Knöpfen mit denen Symmetriegruppe sowie Stiftfarbe und -dicke eingestellt werden können. Die Menüs darüber bieten Zugriff zu weitergehenden Funktionen. Besonders interessant ist das Gitter-Menü, in dem die Anzeige diverser Hilfslinien ein- und ausgeschaltet werden kann.

1.2 Entwicklung

Der „Großvater“ dieses Programms wurde 1986 von Jürgen Richter-Gebert auf einem Atari geschrieben. Anlass war die Ausstellung „Symmetrie in Kunst, Natur und Wissenschaft“ in Darmstadt[12]. Die Portierung dieses Atari-Programms nach DOS durch Rüdiger Krauße war der „Vater“ meines Programms[11]. Beide Programme entstanden unter der Betreuung von Professor Bernhard Ganter.

Im Laufe meines Studiums hatte ich in einer Vorlesung dieses DOS-Programm gesehen. Dabei hatte ich erfahren, dass Bedarf an einer Neuimplementierung bestand. Die Anwendung sollte in das Mathematik-Museum „ix-quadrat“ an der TU München integriert werden. Dieses befand sich zum damaligen Zeitpunkt im Aufbau und hat „Symmetrie“ als eines seiner drei Themengebiete. Um in die digitalen Exponate des Museums integriert werden zu können, musste die Neuimplementierung als Java-Applet lauffähig sein. Außerdem sollte die Benutzerschnittstelle zeitgemäßer gestaltet werden, um intuitiver bedienbar zu sein.

Ich begann, einen Prototypen einer solchen Neuimplementierung zu schreiben. Noch beim Nachdenken über dessen Implementierung kamen mir einige Ideen, wie man das Programm verbessern könnte. Die Grundfunktionalität kommt dadurch mit weniger Ressourcen aus, während die auf heutigen Computern verfügbare Rechenleistung genutzt werden kann, um das Programm ansprechender, intuitiver und vielseitiger zu machen. Im Laufe der Entwicklung, später dann auch unter Einbeziehung von Erfahrungen mit frühen Versionen im Museum, wuchs die Liste der Features und damit verbunden die Komplexität des Programms weiter an.

Mittlerweile verfügt das Programm unter anderem über die folgenden Features, die sich in dieser Kombination in keinem anderen mir bekannten Programm dieser Art finden lassen.

- Einstellbare Gittervektoren erlauben
  - Rotation des Musters,
  - Skalierung des Musters sowie
  - Verformung der Translationszelle.
- Vektororientierte Speicherung der Linienzüge wird verwendet für
  - Beibehaltung des Musters unter Veränderung der Gittervektoren,
  - Beibehaltung gezeichneter Linienzüge bei Wechsel der Symmetriegruppe und
  - Export in verschiedene Vektor- und Raster-Grafikformate.

- Einzeln aktivierbare Anzeige von
  - Rotationszentren, Spiegel- und Gleitspiegelachsen einer Symmetriegruppe,
  - Gitter der Translationszellen und
  - Parkettierung der Fundamentalzellen.
- Kontrolle über die Form der Fundamentalzellen.
- Graphische Feinheiten wie
  - verschiedene Stiftdicken oder
  - Kantenglättung.
- Ausführbar als
  - Applet,
  - JAR-Datei oder
  - per Java WebStart.
- Erkennung von Symmetriegruppen in Bilddateien

All diese Features sind implementiert, ohne die intuitive Benutzbarkeit des Programms zu beeinträchtigen. Der beste Beweis dafür dürften die vielen Kinder sein, die bei Besuchen im ix-quadrat regelmäßig beträchtliche Zeit mit diesem Programm verbringen. Dieses ist mittlerweile eines der meistgenutzten und beliebtesten Exponate im Symmetriebereich von ix-quadrat.

Abbildung 2: Kinder beim Zeichnen von Ornamenten im ix-quadrat

1.3 Vergleiche

Es gibt einige andere Programme, die sich mit dem Zeichnen von ebenen Ornamenten beschäftigen. Im Folgenden werden einige Repräsentanten vorgestellt und mit meinem Programm verglichen.

Ornamente für DOS[11]: Der „Vater“ meines Programms bietet, wie auch schon die Vorgängerversion auf dem Atari, neben einem Modul zum Zeichnen von Mustern auch die Möglichkeit, bei der Analyse eines Musters behilflich zu sein. Dafür können Fragen nach mögliche Symmetrieeigenschaften wie beispielsweise „es gibt eine dreizählge Drehung“ mit „ja“ oder „nein“ beantwortet werden. Die Reihenfolge der Fragen bleibt dem Benutzer überlassen. Zwingende Schlussfolgerungen aus den aktuell angegebenen Antworten werden automatisch getroffen und der Stand der Analyse an einem Verbandgraphen visualisiert. Diese Funktionalität wurde in mein Programm nicht integriert, sondern für das Museum in ein eigenständiges Programm ausgelagert. In gewissem Sinne ist die automatische Mustererkennung meines Programms jedoch ein Nachfolger dieses Moduls.

Unter dem Gesichtspunkt der intuitiven Bedienbarkeit ist mein Programm jedoch ein gutes Stück weiter als sein Vorgänger. Bei diesem ist die Form der Fundamentalzelle fest, und ein Größenwechsel löscht, genauso wie ein Gruppenwechsel, die aktuelle Zeichnung. Als Gitter für Symmetriegruppen mit dreizähligen Drehungen werden Dreiecke angezeigt, von denen je zwei benachbarte eine Translationszelle bilden.

Escher Web Sketch[22]: In diesem Programm kann man nicht nur Freihandlinien zeichnen, sondern auch bestimmte Objekte in der Zeichenebene platzieren. Der Orbit dieser Objekte wird bei gedrückter Maustaste bereits dargestellt, so dass man die Position justieren kann, bevor man das Objekt ablegt. Insgesamt ist dies in meinen Augen ein erster Schritt in Richtung „Konstruieren in kristallographischen Gruppen“, im Unterschied zu „Zeichnen in kristallographischen Gruppen“. Ich habe mich bewusst entschieden, in meinem Programm nur Freihandzeichnungen zu unterstützen. Ich kann mir jedoch gut vorstellen, dass einige der mit meinem Programm gewonnenen Erfahrungen langfristig in die Integration von Transformationsgruppen in die interaktive Geometrie-Anwendung Cinderella[20] mit einfließen, um dort vollwertige Konstruktionsmöglichkeiten in kristallographischen Gruppen zur Verfügung zu stellen.

Insgesamt wirkt Escher Web Sketch wenig intuitiv, insbesondere durch die vielen Drop-Down-Steuerelemente, die jeweils einen zweiten Klick erfordern. Bei der Platzierung von Objekten kommt es mitunter zu gegenseitiger Überdeckung der Bilder und dabei zu Brüchen in der Symmetrie. Die Modifikation der Translationszelle geschieht jeweils von Null ab in mehreren Einzelschritten; die Beeinflussung nur eines einzelnen Parameters ist ebenso unmöglich wie eine Rotation des gesamten Musters. Kantenglättung fehlt.

Kali[24]: Die Version für Windows ist aktueller als die für Mac. Das Programm beherrscht neben den kristallographischen Gruppen auch Punkt- und Friesgruppen. Die Auswahl der Gruppen über ein Icon, das einen kleinen Ausschnitt eines einfachen Motivs in der jeweiligen Gruppe darstellt, ist besonders intuitiv und hübsch. Eine weitere Besonderheit ist die Möglichkeit, neben geraden Linien auch Splines zum Zeichnen verwenden zu können.

Eine Möglichkeit zum Freihandzeichnen fehlt hingegen. Das Zeichnen der Linien mit zwei Maustasten ist eher gewöhnungsbedürftig. Die Darstellung der Gruppenstruktur, topologischer als „Singularitäten“ bezeichnet, ist eher dezent gehalten. Gleitspiegelachsen sind nicht eingezeichnet. Die Translationszelle ist festgelegt, noch nicht einmal die Größe des Musters kann verändert werden.

2 Mathematische Grundlagen

Das Programm wird in großem Umfang von mathematischen Laien genutzt, darunter Künstler, Pädagogen, Physiker, Chemiker sowie Ausstellungsbesucher und Privatanwender mit diversen Hintergründen. Da einige der hier dargestellten Verfahren für dieses breite Publikum interessant, mitunter auch fachlich relevant sein dürften, wurde der nachfolgende Text entsprechend gestaltet. Auf allzu umfangreiche Formalismen wurde bewusst verzichtet, und stattdessen soweit möglich allgemein verständlichen Formulierungen der Vorzug gegeben. Der mathematisch gebildetere Leser möge dies verzeihen.

2.1 Definitionen

Zunächst soll hier die begriffliche Grundlage gelegt werden für die weitere Arbeit. Als Erstes zum Begriff Symmetrie: Im alltäglichen Sprachgebrauch denkt man bei Symmetrie meist an Spiegelsymmetrie. Das ist eine sehr eingeschränkte Auffassung dieses Begriffs. Als entgegengesetztes Extrem kann man die sehr weite Auslegung von Hermann Weyl anführen:

> Symmetrisch ist ein Gebilde dann, wenn man es irgendwie verändern kann und im Ergebnis dasselbe erhält, womit man begonnen hat.[19]

In der vorliegenden Arbeit beschränken wir uns bei den „Gebilden“ auf Muster und Objekte in der Zeichenfläche, und bei den „Veränderungen“ auf Abbildungen, die die Form von Objekten nicht verändern. Während sie also Position, Ausrichtung und Drehsinn ändern können, bleiben Längen und Winkel unverändert (invariant). Die Invarianz der Längen alleine reicht als beschreibende Eigenschaft, da die Invarianz der Winkel daraus folgt.

Definition 1 (Isometrie): Eine Isometrie ist eine längenerhaltende Abbildung.

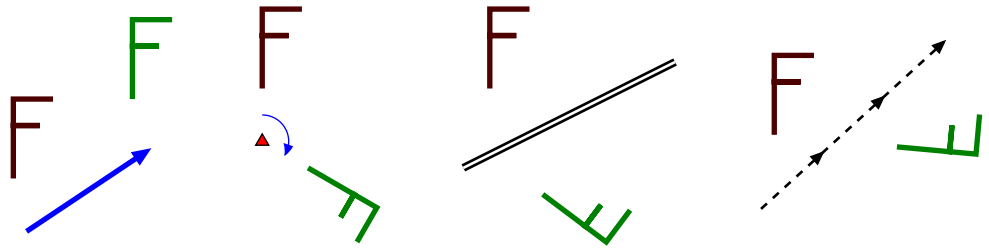
Andere Namen für diese Abbildungen sind „Kongruenzabbildung“, „Kongruenzbewegung“ oder einfach nur „Bewegung“. Bisweilen werden diese Abbildungen auch als „Symmetrien“ bezeichnet, wobei es hier auf eine passende Auffassung des Symmetriebegriffes ankommt.

Das Maß, bezüglich dem Längen gemessen werden, soll im Weiteren das ganz gewöhnliche euklidische Längenmaß sein, wenngleich sich auch mit anderen Metriken, wie beispielsweise hyperbolischer Maßbestimmung, symmetrische Ornamente definieren und erzeugen lassen. In Abschnitt 10.2 auf Seite 65 wird ein kleiner Ausblick auf diese Thematik gegeben.

Der Raum, auf dem die Abbildungen operieren, sei die gewöhnliche, unendlich große, zweidimensionale Ebene. Auch auf gekrümmten Mannigfaltigkeiten oder in höheren Dimensionen lassen sich Ornamente beschreiben, diese sind jedoch nicht Thema dieser Arbeit. Wir beschränken uns also auf die *euklidische Ebene*.

Formal wird die Anwendung einer Abbildung g auf einen Punkt p der Ebene als gp geschrieben. Das Ergebnis ist wieder ein Punkt der Ebene. Werden zwei Abbildungen hintereinander auf p angewandt, erst g und dann h, so schreibt man dies als hgp. Dies kann man als $h(gp)$ auffassen, also die Anwendung von h auf das Ergebnis von gp. Andererseits kann man dies auch als $(hg)p$ lesen, also die Anwendung einer kombinierten Operation hg auf einen Punkt p. Man beachte, dass bei dieser Schreibweise die zuerst ausgeführte Operation immer rechts steht. Die Hintereinanderausführung von Operationen wird auch als *Verkettung* bezeichnet. Zu jeder Isometrie existiert eine *Umkehrabbildung*, auch als „inverse Abbildung“ bezeichnet. Die zu einer Abbildung g inverse Abbildung wird als g^{-1} geschrieben.

In der euklidischen Geometrie der Ebene gibt es vier verschiedene Arten von Isometrien, die auch in Abbildung 3 auf der nächsten Seite dargestellt sind.



(a) Translation (b) Rotation (c) Reflexion (d) Gleitspiegelung

Abbildung 3: Die vier Symmetriearten

Translation ist die *Verschiebung* in eine bestimmte Richtung um eine vorgegebene Distanz.

Rotation ist die *Drehung* um einen bestimmten Punkt (das Drehzentrum) um einen gegebenen Winkel.

Reflexion ist die *Spiegelung* an einer Spiegelachse.

Gleitspiegelung ist eine Kombination aus Translation und Reflexion, bei der die Spiegelachse parallel zur Verschiebungsrichtung liegt.

Ein Sonderfall ist die *Identität*, auch „identische Abbildung“ genannt. Dies ist die Abbildung, die jeden Punkt auf sich selbst abbildet und daher das Muster unverändert lässt. Da man sie auch als Rotation um 0° oder als Translation um die Distanz 0 auffassen kann, lässt sie sich keiner der genannten vier Kategorien eindeutig zuordnen. Mitunter wird die Identität als Rotation um 360° beschrieben.

Bei den Rotationen sind vor allem die Rotationen um Winkel $\frac{2\pi}{n}$ mit ganzzahligem n von Bedeutung. Bei diesen kommt man nach Hintereinanderausführung von n gleichen Drehungen wieder bei der ursprünglichen Lage an. Man spricht von einer n-zähligen Drehung. Eine zweizählige Drehung, also eine Drehung um 180°, wird auch als Punktspiegelung bezeichnet.

Wenden wir nun eine Abbildung auf ein gegebenes Muster an, bilden also jeden Punkt des Musters entsprechend der Abbildungsvorschrift ab, so entsteht dadurch das Bild des Musters. In bestimmten Fällen ist dieses Bild mit dem ursprünglichen Muster identisch. Dann „passt“ die Abbildung in gewisser Weise zum Muster. Eine Abbildung mit dieser Eigenschaft wird als „Automorphismus“ bezeichnet.

> **Definition 2 (Automorphismus):** Ein Automorphismus ist eine strukturerhaltende bijektive Abbildung (Isomorphismus) eines Objektes auf sich selbst.

> **Definition 3 (Symmetriegruppe):** Die Menge aller Isometrien, die Automorphismen eines Musters sind, bilden mit der Verkettung von Abbildungen als Gruppenoperation die *Symmetriegruppe* dieses Musters.

Anschaulich gesprochen ist die Symmetriegruppe eines Musters die Menge all jener Kongruenzabbildungen, die das Muster als ganzes unverändert lassen. Nehmen wir als Beispiel das Ornament aus Abbildung 4 auf der nächsten Seite. Die Symmetriegruppe dieses Ornaments besteht aus acht Elementen: aus der Identität, drei Rotationen um Winkel 90°, 180° und 270° sowie insgesamt vier Spiegelungen an einer horizontalen, einer vertikalen und zwei diagonalen Achsen.

Abbildung 4: Ein symmetrisches Ornament

Die Symmetriegruppe G eines Musters ist eine Gruppe im Mathematischen Sinne. Dies bedeutet im Einzelnen:

1. Die Identität ist Element jeder Symmetriegruppe.

$$\mathrm{id} \in G \tag{2.1}$$

2. Für jede Abbildung der Symmetriegruppe ist die zugehörige Umkehrabbildung auch ein Element der Symmetriegruppe.

$$g \in G \;\Rightarrow\; g^{-1} \in G \tag{2.2}$$

3. Die Verkettung zweier Abbildungen der Symmetriegruppe ist selbst auch wieder ein Element der Symmetriegruppe.

$$g,h \in G \;\Rightarrow\; gh \in G \tag{2.3}$$

4. Die Verkettung von Abbildungen ist assoziativ.

$$a,b,c \in G \;\Rightarrow\; (ab)c = a(bc) \tag{2.4}$$

Der dritte Punkt bedeutet insbesondere, dass von einigen wenigen Symmetrieoperationen eine unendliche Menge von Symmetrien generiert werden kann. So entsteht aus zwei parallelen Spiegelachsen beispielsweise eine Verschiebung, und durch mehrmalige Ausführung dieser Verschiebung unendlich viele Verschiebungen in die gleiche Richtung.

> **Definition 4 (Generator):** Eine (endliche oder unendliche) Menge $\{g_1, g_2, \ldots\}$ von Elementen *generiert* eine Gruppe G, wenn G die kleinste mathematische Gruppe ist, die diese Elemente enthält.

Dazu werden zu den Generatorelementen deren Inverse hinzugenommen und anschließend alle möglichen Kombinationen dieser Elemente mit einander gebildet.

Bisher waren wir von einem gegebenen Muster ausgegangen, und haben dessen Symmetriegruppe analysiert. Umgekehrt kann man auch die Symmetriegruppe vorgeben. Wendet man alle Abbildungen dieser Gruppe auf eine beliebige Punktemenge an, so bildet die Menge aller dadurch

entstehenden Bildpunkte ein Muster. Alle ursprünglich vorgegebenen Isometrien sind zwangsläufig Automorphismen des so entstandenen Musters. Deshalb ist die Symmetriegruppe dieses Musters entweder die ursprünglich vorgegebene Symmetriegruppe, oder aber eine Obergruppe derselben.

> **Definition 5 (Orbit):** Die Menge $Gp = \{gp \mid g \in G\}$ aller Bilder eines Punktes p unter den Elementen der Gruppe G wird als *Orbit* von p unter G bezeichnet.

Der Orbit Gp ist unabhängig davon, welches seiner Elemente man als Ausgangspunkt p nimmt. Diese Eigenschaft ergibt sich aus den auf der vorherigen Seite dargestellten Gruppenaxiomen. Als Folge daraus stellen die Orbits *Äquivalenzklassen* der Ebene dar: jeder Punkt gehört zu genau einer dieser Klassen. Der Orbit einer Punktemenge ist die Vereinigung der Orbits aller Punkte der Menge.

Wenn man aufs Geratewohl einige Abbildungen vorgibt, und die dadurch generierte Gruppe als Symmetriegruppe für ein Muster verwenden will, wird man dabei häufig eher langweilige Gruppen erhalten, bei denen sich Bildpunkte beliebig nah aneinander drängen und somit nicht mehr einzeln erkennbar sind.

> **Definition 6 (Häufungspunkt):** Ein Punkt h der Ebene heißt *Häufungspunkt* einer Punktemenge M, wenn in jeder noch so kleinen Umgebung um h mindestens ein von h verschiedener Punkt aus M liegt. Formal ausgedrückt:

$$\forall \varepsilon > 0 \; \exists p \in M : \|p - h\| \leq \varepsilon \wedge p \neq h \tag{2.5}$$

Aus der Definition folgt automatisch, dass in jeder Umgebung um einen Häufungspunkt unendlich viele Punkte p liegen müssen. Andernfalls könnte man ε kleiner als den Abstand zum nächstgelegenen Punkt wählen, um eine leere Umgebung zu erhalten, was der Definition des Häufungspunktes widerspräche. Das Konzept der Häufungspunkte kann nun verwendet werden, um „sinnvolle“ Abbildungsgruppen zu definieren.

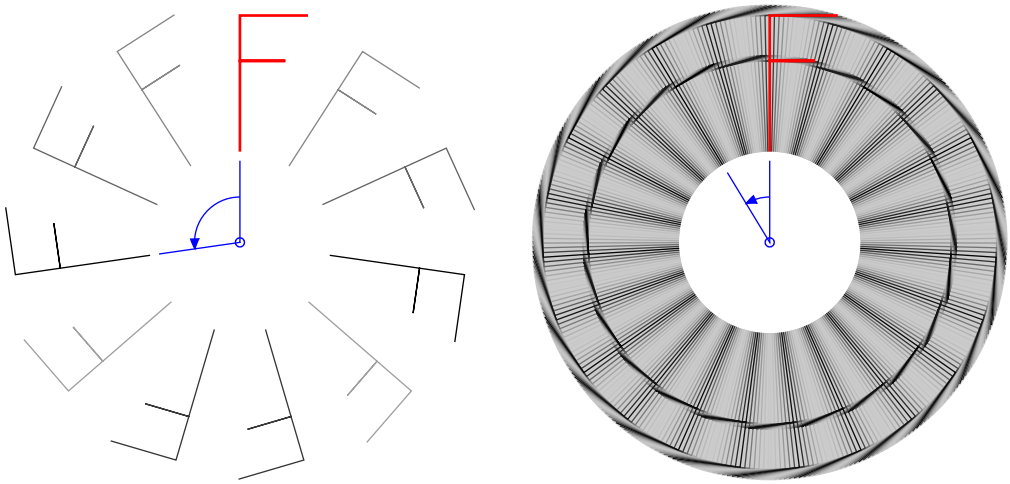
> **Definition 7 (diskrete Gruppe):** Eine Gruppe von Abbildungen heißt *diskret*, wenn kein Orbit eines Punktes unter dieser Gruppe einen Häufungspunkt hat.

Die Bedeutung diskreter Gruppen ist in Abbildung 5 auf der nächsten Seite illustriert: Nur bei diesen sind die Bilder eines Punktes klar von einander getrennt, da es sowohl für Translationsentfernungen als auch für Drehwinkel eine untere Schranke gibt, diese also nicht beliebig klein werden können. In der weiteren Betrachtung wollen wir uns auf diskrete Gruppen beschränken.

Wendet man auf ein Muster eine Isometrie a an, die nicht Element der Symmetriegruppe dieses Musters ist, so hat das Bild des Musters eine Symmetriegruppe, die etwas andere Abbildungen enthält. Jeder Abbildung g in der ursprünglichen Symmetriegruppe entspricht eine Abbildung aga^{-1} in der Symmetriegruppe des Bildmusters. Aus einer Rotation um einen Punkt wird beispielsweise eine Rotation um einen anderen Punkt. Die Struktur der Gruppe bleibt durch diese *Konjugation* jedoch erhalten, die Rotation bleibt beispielsweise eine Rotation mit dem gleichen Drehwinkel. Um die Struktur zu erhalten muss a noch nicht einmal zwingend eine Isometrie sein, sondern darf eine beliebige affine Abbildung sein, so lange die Elemente aga^{-1} wieder Isometrien sind. Auf affine Abbildungen wird auf Seite 15 genauer eingegangen.

> **Definition 8 (Äquivalente Gruppen):** Zwei Symmetriegruppen G und H sind äquivalent, wenn es eine affine Abbildung a gibt, so dass $H = \{aga^{-1} \mid g \in G\}$.

Dieser Begriff äquivalenter Gruppen ist eng verwandt mit dem Begriff der *Gruppenisomorphismen*. Während diese jedoch keinerlei geometrische Interpretation haben und daher potentiell Translationen mit Rotationen vertauschen oder dergleichen, erhält die eben definierte Äquivalenzrelation die Kategorien der Abbildungen und damit die geometrische Interpretation.



(a) Eine diskrete Symmetriegruppe, erzeugt von einer Rotation um $\frac{6}{11}\pi$

(b) Eine nicht diskrete Symmetriegruppe, erzeugt von einer Rotation um $\frac{6}{11}$

Abbildung 5: Diskrete und nicht diskrete Symmetriegruppen

Definition 9 (Struktur): Die Struktur einer Symmetriegruppe sind diejenigen Eigenschaften, die allen äquivalenten Gruppen gemeinsam sind.

Wenn Symmetriegruppen mit einem Namen bezeichnet werden, so ist dabei im Allgemeinen nur die Struktur der Gruppe von Bedeutung, nicht die genaue Definition jeder einzelnen Abbildung. Aus diesem Grund bezeichnen im folgenden Text Gruppennamen immer jeweils eine ganze Klasse von äquivalenten Gruppen.

2.2 Formulierung von Transformationen

Zur Darstellung von Punkten in der Ebene werden kartesische Koordinaten eingesetzt. Man arbeitet dann mit Vektoren der Form $\begin{pmatrix} x \\ y \end{pmatrix}$. Eine Verschiebung um t_x in x-Richtung sowie um t_y in y-Richtung hat die folgende Form:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{2.6}$$

Eine Drehung um den Ursprung um einen Winkel α lässt sich schreiben als:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\alpha x - \sin\alpha y \\ \sin\alpha x + \cos\alpha y \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{2.7}$$

Eine Spiegelung an der x-Achse lässt sich so formulieren:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ -y \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{2.8}$$

Die unterschiedliche Notation dieser Operationen kann man vermeiden, indem man homogene Koordinaten verwendet. Dazu wird eine dritte Vektorkomponente hinzugenommen, die immer

den Wert 1 beibehält. Die drei oben beschriebenen Operationen nehmen damit die folgende Form an:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \\ 1 \end{pmatrix} \qquad = \qquad \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.9}$$

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\alpha x - \sin\alpha y \\ \sin\alpha x + \cos\alpha y \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.10}$$

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ -y \\ 1 \end{pmatrix} \qquad = \qquad \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.11}$$

Jede Abbildung wird also durch die Multiplikation mit einer Matrix ausgedrückt. Eine kurze Einführung in die Rechnung mit Matrizen findet sich in den Gleichungen 2.13 bis 2.15 auf den Seiten 15–16. Die Anwendung von zwei Abbildungen hintereinander, also deren Verkettung, wird zu einem Matrixprodukt. Dabei steht die Matrix des zuerst angewandten Operators rechts in der Formel, wie sich aus dem Assoziativgesetz leicht ablesen lässt.

$$A \cdot (B \cdot p) = (A \cdot B) \cdot p$$

Das Produkt zweier 3×3-Matrizen ist wieder eine 3×3-Matrix. Bei der speziellen Form der Matrizen, wie sie oben angegeben sind, ist auch nach Multiplikationen die dritte Zeile stets (0 0 1), so dass der letzte Eintrag der Vektoren immer beibehalten wird. Sie gehören somit zur Klasse der sogenannten *affinen Transformationen*, deren Matrixdarstellung die folgende allgemeine Form hat.

$$\begin{pmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{pmatrix} \tag{2.12}$$

Ebenfalls zu den affinen Transformationen gehören *Streckungen* sowie *Scherungen*. Die Menge aller Isometrien ist also eine Teilmenge der affinen Transformationen.

Für Leser, die mit Matrizenrechnung nicht vertraut sind, seien hier noch die wichtigsten Formeln zum Rechnen mit affinen Abbildungen angegeben. Die Anwendung einer affinen Abbildung auf einen Vektor:

$$\begin{pmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} ax + cy + e \\ bx + dy + f \\ 1 \end{pmatrix} \tag{2.13}$$

Die Multiplikation zweier Matrizen zur Verkettung von Abbildungen:

$$\begin{pmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} g & i & k \\ h & j & l \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} ag + ch & ai + cj & ak + cl + e \\ bg + dh & bi + dj & bk + dl + f \\ 0 & 0 & 1 \end{pmatrix} \tag{2.14}$$

Die Inverse als Matrix der Umkehrabbildung:

$$\begin{pmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{pmatrix}^{-1} = \frac{1}{ad-bc} \begin{pmatrix} d & -c & cf-ed \\ -b & a & eb-af \\ 0 & 0 & ad-bc \end{pmatrix} \tag{2.15}$$

Die Programmiersprache Java bietet mit der Klasse `java.awt.geom.AffineTransform` eine Implementierung dieser affinen Abbildungen. An den meisten Stellen verwendet mein Programm Objekte dieser Klasse zur Repräsentation von Abbildungen.

3 Klassifikation von Symmetriegruppen

Der folgende Abschnitt soll die verschiedenen Klassen von Symmetriegruppen benennen und beschreiben. Das erste Unterscheidungsmerkmal bei der Klassifizierung ist die Anzahl der linear unabhängigen TranslationVerschiebungen, die eine Symmetriegruppe enthält.

Rosettengruppen enthalten keine Translationen.

Friesgruppen enthalten Translationen nur in einer Richtung.

kristallographische Gruppen enthalten Translationen in zwei unterschiedliche Richtungen.

Die Tabellen 1 bis 3 auf den Seiten 17–19 enthalten Abbildungen verschiedener Ornamente aus diesen drei Familien[10]. Im Anhang sind diese Ornamente auf den Seiten 72–80 noch einmal größer abgebildet. Das Ornament wird jeweils in ein Strukturdiagramm übergeblendet, das die Symmetrieeigenschaften des Musters abstrakt darstellt. Es sind alle Drehzentren und Spiegel- sowie Gleitspiegelachsen abgebildet. Eine Legende ist am unteren Rand von Tabelle 4 auf Seite 20 gegeben. Darüber hinaus sind zwei repräsentative Translationsvektoren angegeben, deren ganzzahlige Linearkombinationen alle Translationen der Gruppe erzeugen. Bei Gruppen mit geringer Symmetrie sind die Grenzen von Fundamentalbereichen angedeutet.

3.1 Rosettengruppen

Rosettengruppen, auch *Punktgruppen* genannt, enthalten keine Translationen. Dadurch scheiden auch Gleitspiegelungen als mögliche Elemente aus, da die doppelte Ausführung einer Gleitspiegelung eine Verschiebung ergibt. Als mögliche Symmetrieoperationen verbleiben Drehungen und Spiegelungen. Beispiele für Ornamente mit Rosettengruppen sind in Tabelle 1 auf der nächsten Seite abgebildet.

Es gibt zwei verschiedene Arten von Rosettengruppen. Als Unterscheidungsmerkmal dient die Präsenz von Spiegelungen. Rosettengruppen ohne Achsspiegelungen werden mit dem Symbol C_n gekennzeichnet und als *zyklischen Gruppen* oder auch *Drehgruppen* bezeichnet. Rosettengruppen mit Spiegelung hingegen werden mit D_n gekennzeichnet und als *Diedergruppen* bezeichnet. In beiden Fällen bedeutet der ganzzahlige Index n, dass die kleinste vorkommende Drehung $\frac{2\pi}{n}$ beträgt. Da dieser Winkel beliebig klein sein kann, also n jede positive natürliche Zahl annehmen kann, gibt es unendlich viele Klassen strukturell unterschiedlicher Rosettengruppen.

Besondere Betrachtung verdienen die Klassen mit Index 1. Die Symmetriegruppe C_1 enthält nur die Identität als einziges Element. Dies ist daher die Symmetriegruppe unsymmetrischer Muster. Die Gruppe D_1 enthält neben der Identität noch eine Achsspiegelung als zweites Element. Dies ist die Symmetriegruppe ausschließlich spiegelsymmetrischer Muster. Die Gruppe C_2

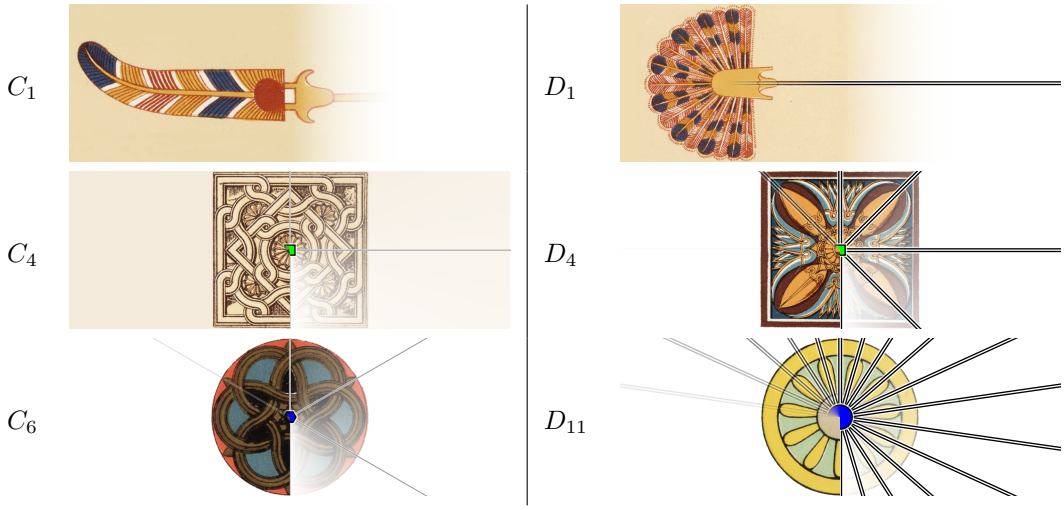


Tabelle 1: Beispiele von Ornamenten mit Rosettengruppen

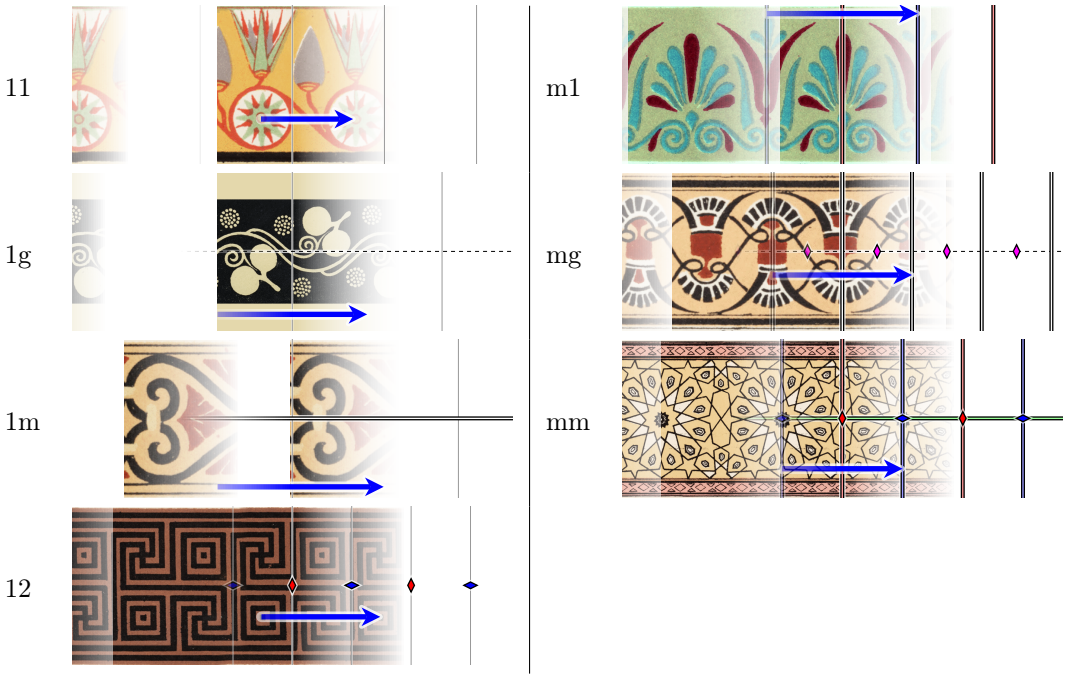


Tabelle 2: Beispiele von Ornamenten mit Friesgruppen

aller ausschließlich Punktspiegelungpunktsymmetrischen Muster gehört ebenfalls zu den Rosettengruppen.

Bei den Beispielen für die Gruppen C_4 und C_6 in Tabelle 1 wird die Spiegelsymmetrie jeweils nur dadurch gebrochen, welches Band an den Kreuzungspunkten jeweils oben liegt. Allgemein ist es bei dekorativen Ornamenten häufig so, dass das Ornament in groben Zügen eine stärkere Symmetrie aufweist, die im Detail dann jedoch an einzelnen Stellen bewusst gebrochen wird.

3.2 Friesgruppen

Friesgruppen, auch *Bandornamentgruppen* genannt, enthalten Translationen in einer Richtung. Es gibt sieben verschiedene Klassen von Friesgruppen, die alle in Tabelle 2 auf der vorherigen Seite mit Beispielen abgebildet sind.

Die Benennung der Gruppen erfolgt hier mit dem Hermann-Mauguin-Symbol, auf das im folgenden Kapitel noch genauer eingegangen wird. Sie setzt sich aus zwei Symbolen zusammen. Das erste beschreibt Spiegelachsen senkrecht zur Richtung der Translation. Ein m steht für die Existenz derartiger Spiegelachsen, eine 1 für deren Abwesenheit. Das zweite Symbol beschreibt Symmetrieachsen in Richtung der Translation. Der Buchstabe m steht wieder für eine Spiegelachse, g für eine Gleitspiegelachse. Die Ziffer 2 symbolisiert zweizählige Drehungen in Abwesenheit von Spiegelungen oder Gleitspiegelungen. Ist keine dieser Eigenschaften gegeben, steht eine 1, was sich auch als 1-zählige Drehung lesen lässt, also als Identität. Häufig wird auch für die Friesgruppen der Buchstabe p vorangestellt. Zur besseren Unterscheidung von den kristallographischen Gruppen wurde hier jedoch darauf verzichtet.[23]

3.3 Kristallographische Gruppen

Kristallographische Gruppen, im Englischen als „wallpaper groups“ (Tapetengruppen) bezeichnet, enthalten mindestens zwei linear unabhängige Translationen, also Verschiebungen in zwei unterschiedliche Richtungen. Als Folge davon enthalten sie automatisch Translationen in unendlich viele Richtungen.

Es gibt mehrere verschiedene gebräuchliche Bezeichnungen für jede dieser Gruppen. Im Folgenden wird überwiegend die Kurzform des Hermann-Mauguin-Symbols verwendet. Dabei steht eine Zahl für ein Drehzentrum entsprechender Zähligkeit, ein m für eine Spiegelachse und ein g für eine Gleitspiegelachse. Die Reihenfolge sagt etwas über die Lage dieser Symmetrien zueinander aus. Bei der Kurzform werden, im Unterschied zu der aus immer vier Zeichen bestehenden Langform, alle sich implizit ergebenden Eigenschaften weggelassen, solange die Eindeutigkeit gewahrt bleibt. Diese Art der Bezeichnung kommt aus der Kristallographie, also der Untersuchung realer Kristallstrukturen. Diese sind typischerweise dreidimensional, und eine genaue Definition der Notation benötigt etwas mehr Hintergrundwissen aus dieser Wissenschaft. Darauf soll hier verzichtet und die Namen einfach nur als Namen behandelt werden.[8]

Eine zweite, etwas mathematischere Art der Bezeichnung ist die *Orbifold-Notation*. Sie ist in Tabelle 4 auf Seite 20 mit angegeben. Die Orbifold-Notation ist interessant, da sie sich leicht auch auf nicht-euklidische Geometrien anwenden lässt. Außerdem kann sie verwendet werden um zu beweisen, dass es nicht mehr als diese 17 kristallographischen Gruppen in der euklidischen Ebene gibt.[4]

Beispiele von Ornamenten dieser Symmetriegruppen sind in Tabelle 3 auf der nächsten Seite abgebildet. Manchmal ist eine potentiell stärkere Symmetrie wieder nur durch kleine Details bewusst gebrochen worden, wie etwa die weißen Streifen in p2, die feinen Linien in pg und pgg, die Farbgebung in pmm oder die scheinbare vertikale Anordnung in p3.

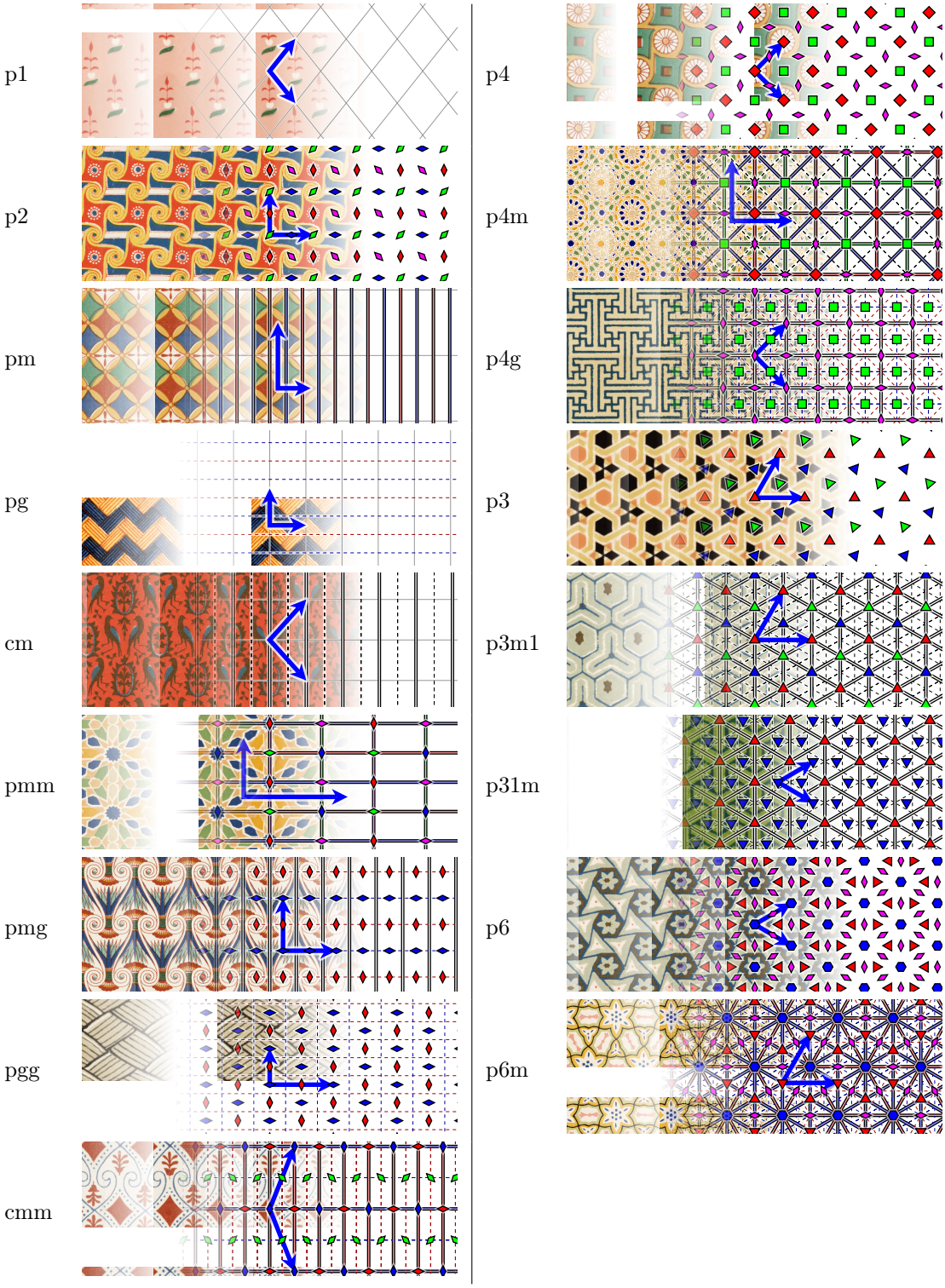
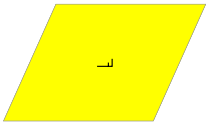
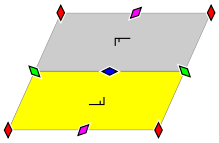


Tabelle 3: Beispiele von Ornamenten mit kristallographischen Gruppen

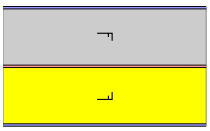
Name: p1
Orbifold-Symbol: ∘1
TZ: Parallelogramm
FB: 1



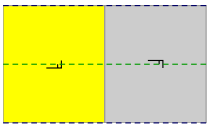
Name: p2
Orbifold-Symbol: 2222
TZ: Parallelogramm
FB: 2



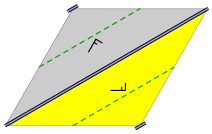
Name: pm
Orbifold-Symbol: ∗∗
TZ: Rechteck
FB: 2



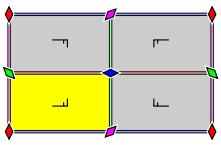
Name: pg
Orbifold-Symbol: ××
TZ: Rechteck
FB: 2



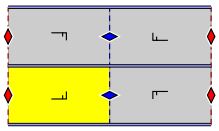
Name: cm
Orbifold-Symbol: ∗×
TZ: Raute
FB: 2



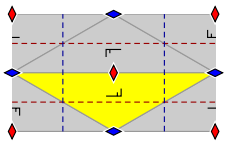
Name: pmm
Orbifold-Symbol: ∗2222
TZ: Rechteck
FB: 4 Rechtecke



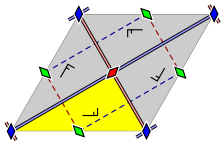
Name: pmg
Orbifold-Symbol: 22∗
TZ: Rechteck
FB: 4



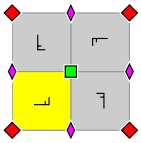
Name: pgg
Orbifold-Symbol: 22×
TZ: Rechteck
FB: 4



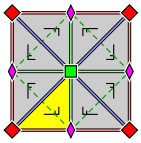
Name: cmm
Orbifold-Symbol: 2∗22
TZ: Raute
FB: 4



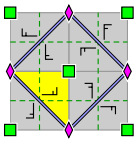
Name: p4
Orbifold-Symbol: 442
TZ: Quadrat
FB: 4



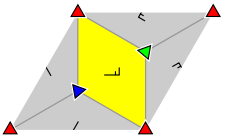
Name: p4m
Orbifold-Symbol: ∗442
TZ: Quadrat
FB: 8 rechtwinklige △



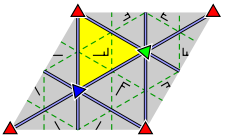
Name: p4g
Orbifold-Symbol: 4∗2
TZ: Quadrat
FB: 8



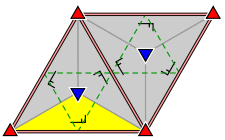
Name: p3
Orbifold-Symbol: 333
TZ: 2 gleichseitige △
FB: 3



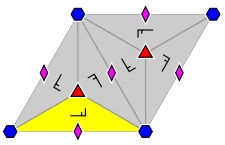
Name: p3m1
Orbifold-Symbol: ∗333
TZ: 2 gleichseitige △
FB: 6 gleichseitige △



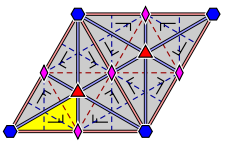
Name: p31m
Orbifold-Symbol: 3∗3
TZ: 2 gleichseitige △
FB: 6



Name: p6
Orbifold-Symbol: 632
TZ: 2 gleichseitige △
FB: 6



Name: p6m
Orbifold-Symbol: ∗632
TZ: 2 gleichseitige △
FB: 12 rechtwinklige △



Spiegelung — 2-zählige Drehung ♦ — 3-zählige Drehung ▲
Gleitspiegelung — 4-zählige Drehung ■ — 6-zählige Drehung ⬢
Unterschiedliche Farben repräsentieren unterschiedliche Äquivalenzklassen

TZ: Allgemeine Form eine minimalen konvex polygonalen Translationszelle
FB: Anzahl Fundamentalbereiche pro Translationszelle, ggf. Form

Tabelle 4: Übersicht über die ebenen kristallographischen Gruppen

Die Struktureigenschaften dieser Gruppen sind in Tabelle 4 auf der vorherigen Seite noch einmal größer schematisch dargestellt. Diese Tabelle listet auch weiterführende Informationen auf, wie die Form einer Translationszelle, wie sie in Definition 10 eingeführt wird, oder die Anzahl der Fundamentalbereiche gemäß Definition 11 auf Seite 29. Die Implementierung beschränkt den Benutzer auf die jeweils angegebene Form der Translationszelle. In manchen Fällen sind andere Formen möglich, führen jedoch nicht zu anderen Symmetriegruppen innerhalb der gleichen Klasse. In diesem Sinne ist die Form der Translationszellen allgemein, da jede mögliche Symmetriegruppe einer Klasse auswählbar ist, wenn auch nicht jede mögliche Unterteilung in Translationszellen.

Das Programm beherrscht derzeit nur das Zeichnen in diesen kristallographischen Gruppen. Aus diesem Grund beschränken sich auch die weiteren Ausführungen in dieser Arbeit auf kristallographische Gruppen.

Teil II
Konzepte der Implementierung

4 Berechnung von Orbits

Wie nach Definition 8 auf Seite 13 beschrieben, bezeichnet ein Gruppenname prinzipiell eine ganze Äquivalenzklasse von Symmetriegruppen mit gleicher Struktur. Welches Mitglied aus dieser Klasse tatsächlich gemeint ist, wird durch zwei linear unabhängige Translationsvektoren a und b definiert, die die Ebene mit Kopien einer *Translationszelle* überdecken. Die allen Mitgliedern einer Klasse gemeinsame Struktur hingegen wird durch einen Satz *innerer Abbildungen* festgelegt, die die Symmetrien innerhalb einer Translationszelle beschreiben.

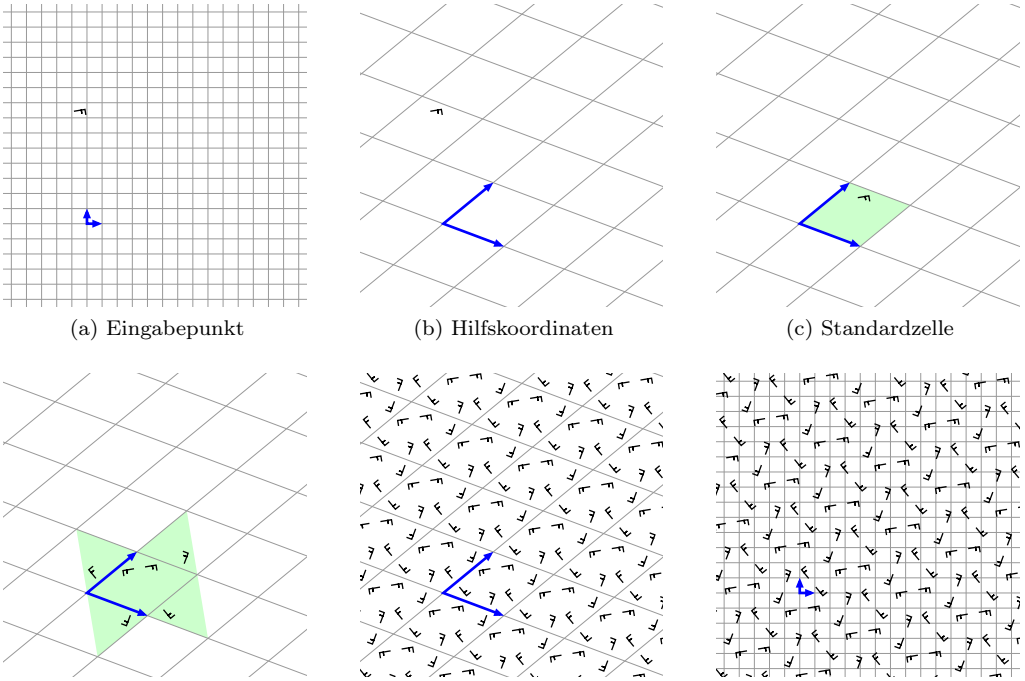
> **Definition 10 (Translationszelle):** Eine Translationszelle einer Symmetriegruppe G ist eine abgeschlossene Teilmenge T der Ebene, so dass die Bilder von T unter den in G enthaltenen Translationen die gesamte Ebene überdecken und der Schnitt zweier Bilder keine inneren Punkte hat.

Im Rahmen dieser Arbeit werden Translationszellen von den Vektoren a und b aufgespannt und haben daher immer die Form eines Parallelogramms.

Beim Zeichnen innerhalb einer kristallographischen Gruppe muss zu dem Punkt, an dem der Benutzer gerade mit dem Cursor zeichnet, der zugehörige Orbit gefunden werden. Dazu geht mein Programm wie im Folgenden beschrieben vor. Der Algorithmus ist auch in Abbildung 7 auf Seite 24 als Programmfragment formuliert sowie in Abbildung 4 auf der nächsten Seite graphisch veranschaulicht.

4.1 Transformation in Hilfskoordinaten

Jeder Punkt p der Ebene kann als Linearkombination der beiden Translationsvektoren a und b dargestellt werden. Daher kann man diese beiden Vektoren als Einheitsvektoren eines schiefen (also nicht kartesischen) Hilfs-Koordinatensystems auffassen. In diesem Koordinatensystem hat ein Punkt die Koordinaten (p_a, p_b) während er die kartesischen Bildschirmkoordinaten (p_x, p_y) hat. Im Folgenden werden diese Hilfskoordinaten auch als (a,b)-Koordinaten bezeichnet. Die



(a) Eingabepunkt (b) Hilfskoordinaten (c) Standardzelle

(d) innere Symmetrien (e) Translationen (f) Bildschirmkoordinaten

Abbildung 6: Schritte der Orbitberechnung

Transformation der Bildschirmkoordinaten in Hilfskoordinaten erfolgt über eine affine Abbildung.

$$\begin{pmatrix} p_a \\ p_b \\ 1 \end{pmatrix} = \begin{pmatrix} a_x & b_x & 0 \\ a_y & b_y & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} p_x \\ p_y \\ 1 \end{pmatrix} \tag{4.1}$$

4.2 Wechsel der Translationszelle

Die Translationszellen, die die Ebene überdecken, sind alle zueinander äquivalent. Man kann sich also aussuchen, in welcher dieser Zellen die weiteren Berechnungen ausgeführt werden. Als Standard-Translationszelle wird $\{q_a a + q_b b \,|\, 0 \leq q_a, q_b < 1\}$ verwendet, also die Menge aller Punkte, bei denen die Hilfskoordinaten zwischen 0 und 1 liegen. Sie erreicht man, indem man von den Hilfskoordinaten einfach geeignete Ganzzahlen abziehe, nämlich jeweils den abgerundeten Wert. Bei positiven Zahlen entspricht das einfach einem Ersetzen der Ziffern vor dem Dezimalkomma durch eine 0. Geometrisch entspricht diese Transformation der Ausführung entsprechender Vielfacher der bestimmenden Translationen, der Punkt bleibt also innerhalb des gleichen Orbits.

$$\begin{pmatrix} p'_a \\ p'_b \\ 1 \end{pmatrix} = \begin{pmatrix} p_a - \lfloor p_a \rfloor \\ p_b - \lfloor p_b \rfloor \\ 1 \end{pmatrix} \tag{4.2}$$

4.3 Anwenden innerer Symmetrien

Innerhalb einer Translationszelle werden dann der Gruppenstruktur entsprechende Transformationen ausgeführt. Diese beschreiben die Symmetrie innerhalb einer Translationszelle, im Folgenden kurz „innere Symmetrie“ genannt. Eine Liste der inneren Symmetrien aller Gruppen findet sich in Tabelle 5 auf Seite 25. Das Ergebnis der Transformation ist nicht zwingend auf die Standardzelle beschränkt, liegt aber zumindest in ihrer Nähe. An dieser Stelle werden erstmals aus dem einen Punkt mehrere, was im Programm durch eine entsprechende Schleife ausgedrückt werden kann.

$$\begin{pmatrix} p''_a \\ p''_b \\ 1 \end{pmatrix} = M_i \begin{pmatrix} p'_a \\ p'_b \\ 1 \end{pmatrix} \qquad M_i \text{ innere Symmetrie der Gruppe} \tag{4.3}$$

4.4 Anwenden der Translationen

Die kleine Menge von Bildpunkten in der unmittelbaren Umgebung der Standardzelle wird jetzt mit den bestimmenden Translationen kombiniert, um so die gesamte Ebene zu füllen. In der Theorie ist die Ebene unendlich groß, so dass unendlich viele solche Translationen notwendig sind. In der Praxis reicht natürlich die Beschränkung auf den tatsächlich sichtbaren Ausschnitt der Ebene, so dass auch hier die Anzahl der Bildpunkte endlich bleibt. Auf das Verfahren zur Ermittlung der relevanten Bereiche wird in Abschnitt 5.2 auf Seite 26 noch genauer eingegangen. Die Translationen sind einfach realisiert durch die Addition ganzer Zahl auf die Hilfskoordinaten.

$$\begin{pmatrix} p'''_a \\ p'''_b \\ 1 \end{pmatrix} = \begin{pmatrix} p''_a \\ p''_b \\ 1 \end{pmatrix} + \begin{pmatrix} \Delta_a \\ \Delta_b \\ 1 \end{pmatrix} \qquad \Delta_a, \Delta_b \in \mathbb{Z} \tag{4.4}$$

4.5 Rücktransformation

Vor der Darstellung auf dem Bildschirm müssen die Hilfskoordinaten wieder zurück in kartesische Bildschirmkoordinaten transformiert werden.

$$\begin{pmatrix} p'_x \\ p'_y \\ 1 \end{pmatrix} = \begin{pmatrix} a_x & b_x & 0 \\ a_y & b_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p'''_a \\ p'''_b \\ 1 \end{pmatrix} \tag{4.5}$$

4.6 Zusammenfassung

In Abbildung 7 auf der nächsten Seite ist der gesamte Algorithmus zur Berechnung eines Orbits noch einmal in Java formuliert. Die Kommentare verweisen auf den Abschnitt, in dem der in dieser Zeile ausgeführte Schritt beschrieben ist.

Es ist zu beachten, dass einige der verwendeten affinen Abbildungen in kartesischen Koordinatensystemen betrachtet keine Isometrien darstellen. Dies kommt von der möglichen Verzerrung des Hilfskoordinatensystems, dessen Basisvektoren oft weder gleich lang noch zueinander senkrecht sind. In diesem speziellen Koordinatensystem sind die inneren Symmetrien tatsächlich Isometrien, und die Transformationen zwischen Hilfskoordinaten und Bildschirmkoordinaten drücken keine Veränderung des Punktes an sich aus, sondern nur seiner Repräsentation. Die Kombination aus allen ausgeführten Abbildungen ist auch in kartesischen Koordinaten immer eine Isometrie. Dazu müssen jedoch je nach Gruppe Einschränkungen für die Form der Translationszelle gelten, wie sie auch in Tabelle 4 auf Seite 20 angegeben sind.

Eingabe:
- Die Menge M der inneren Symmetrien
- Der Punkt p, dessen Orbit berechnet wird
- Die Komponenten der Basisvektoren a und b
- Schleifengrenzen entsprechend des sichtbaren Bereiches

Ausgabe: der den Schleifengrenzen entsprechende Ausschnitt aus dem Orbit von p, repräsentiert als Liste von Punkten

```
List<Point2D> orbit(AffineTransform[] m, Point2D p,
                    double ax, double ay, double bx, double by,
                    int minA, int maxA, int minB, int maxB)
        throws NoninvertibleTransformException
{
    List<Point2D> result = new ArrayList<Point2D>();
    Point2D.Double p1 = new Point2D.Double();
    Point2D.Double p2 = new Point2D.Double();
    Point2D.Double p3 = new Point2D.Double();
    Point2D p4;
    AffineTransform ab  = new AffineTransform(ax, ay, bx, by, 0, 0);

    ab.inverseTransform(p, p1);                                // 4.1
    p1.x = p1.x - Math.floor(p1.x);                            // 4.2
    p1.y = p1.y - Math.floor(p1.y);                            // 4.2
    for (int i = 0; i < m.length; ++i) {
        m[i].transform(p1, p2);                                // 4.3
        for (int da = minA; da <= maxA; ++da) {
            p3.x = p2.x + da;                                  // 4.4
            for (int db = minB; db <= maxB; ++db) {
                p3.y = p2.y + db;                              // 4.4
                p4 = ab.transform(p3, null);                   // 4.5
                result.add(p4);
            }
        }
    }

    return result;
}
```

Abbildung 7: Algorithmus zur Erzeugung des Orbits eines Punktes

p1	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$						
p2	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$					
pm	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$					
pg	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0{,}5 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$					
cm	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$					
pmm	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$			
pmg	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & 1 & 0{,}5 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 0{,}5 \\ 0 & 0 & 1 \end{pmatrix}$			
pgg	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 0{,}5 \\ 0 & 1 & 0{,}5 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0{,}5 \\ 0 & -1 & 0{,}5 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$			
cmm	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$			
p4	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$			
p4m	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$
p4g	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 0{,}5 \\ 0 & 1 & 0{,}5 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 0{,}5 \\ 1 & 0 & 0{,}5 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0{,}5 \\ 0 & -1 & 0{,}5 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0{,}5 \\ -1 & 0 & 0{,}5 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$
p3	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ -1 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$					
p3m1	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & -1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ -1 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -1 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$		
p31m	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ -1 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$		
p6	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ -1 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$		
p6m	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 1 \\ -1 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 1 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$		
		$\begin{pmatrix} 0 & 1 & 0 \\ -1 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & -1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	

Tabelle 5: Innere Symmetrien in (a,b)-Koordinaten

5 Darstellung auf dem Bildschirm

5.1 Verwendung rechteckiger Puffer

Besonders wenn der Bildschirm viele Kopien einer recht kleinen Translationszelle darstellt, bedeutet das Wiederholen einer Linie in jeder einzelnen dieser Zellen einen erheblichen Rechenaufwand. Dieser kann vermieden werden, indem man in einen Zwischenpuffer zeichnet, der in etwa die Größe einer Translationszelle hat, und dann mehrere Kopien dieses Puffers durch schnelle Rasteroperationen auf dem Bildschirm darstellt. Im Unterschied zu Translationszellen sind Raster zwangsläufig rechteckig. Die Verwendung von Transparenz würde die Geschwindigkeit deutlich reduzieren. Ein geschicktere Alternative ist es, die Kopien des Puffers auf dem Bildschirm einander überlappen zu lassen, wie in Abbildung 8 dargestellt.

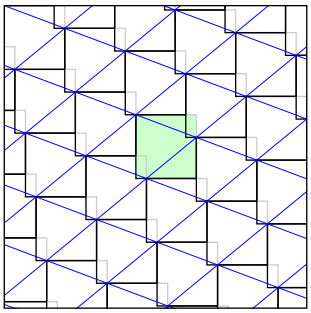


Abbildung 8: Translationszellen und überlappende Puffer

Die Breite w und Höhe h des verwendeten Puffers ergeben sich aus den Koordinaten der Translationsvektoren a und b:

$$w = \max\{0, a_x, b_x\} - \min\{0, a_x, b_x\} \tag{5.1}$$

$$h = \max\{0, a_y, b_y\} - \min\{0, a_y, b_y\} \tag{5.2}$$

5.2 Bestimmung von Überdeckungen

Da der Puffer den Inhalt mehrerer benachbarter Translationszellen enthalten kann, müssen alle Translationszellen gezeichnet werden, von denen mindestens ein Pixel im Puffer sichtbar ist. Im nächsten Schritt müssen dann vom Puffer alle Kopien gezeichnet werden, von denen mindestens ein Pixel auf dem Bildschirm sichtbar ist. Beide Male werden dazu zwei geschachtelte Schleifen verwendet, deren Schleifengrenzen geeignet gewählt werden müssen. Die einzelnen Schritte zur Bestimmung geeigneter Grenzen sind in Abbildung 9 auf Seite 28 dargestellt, auf deren Teilbilder im Folgenden an entsprechenden Stellen verwiesen wird.

Etwas abstrakter betrachtet stellt sich das Problem wie folgt dar:
Gegeben zwei achsenparallele rechteckige Bereiche (a)

$$R = \left\{ \begin{pmatrix} r_x \\ r_y \end{pmatrix} \,\middle|\, r_x^{\min} \leq r_x \leq r_x^{\max} \text{ und } r_y^{\min} \leq r_y \leq r_y^{\max} \right\} \tag{5.3}$$

$$S = \left\{ \begin{pmatrix} s_x \\ s_y \end{pmatrix} \,\middle|\, s_x^{\min} \leq s_x \leq s_x^{\max} \text{ und } s_y^{\min} \leq s_y \leq s_y^{\max} \right\} \tag{5.4}$$

Gesucht werden (mindestens) alle ganzzahligen Paare

$$(i,j) \in \mathbb{Z}^2 \tag{5.5}$$

die einen Verschiebungsvektor

$$t = \begin{pmatrix} t_x \\ t_y \end{pmatrix} = ia + jb \tag{5.6}$$

so bestimmen, dass R verschoben um t mindestens einen Punkt mit S gemeinsam hat.

Im ersten Anwendungsfall, der Überdeckung des Puffers mit Kopien der Translationszelle, ist R das umschließende achsparallele Rechteck der Translationszelle, und S das Rechteck des Puffers. Im zweiten Anwendungsfall, der Überdeckung des Anzeigefensters mit Kopien des Puffers, ist R das Rechteck des Puffers und S das Rechteck des Anzeigefensters.

Der im Folgenden beschriebene Algorithmus ermittelt Schleifengrenzen für i und j. Er ermittelt damit möglicherweise auch Verschiebungsvektoren, bei denen keine gemeinsamen Punkte mehr vorliegen. Daher ist oben in der Zielsetzung die Rede von *mindestens* allen Paaren, bei denen es zu Überlappungen kommt.

Betrachten wir nun die Extremsituationen, in denen die beiden Rechtecke gerade noch einen einzigen Eckpunkt gemeinsam haben (b). Daraus folgen entsprechende Schranken für den Verschiebungsvektor t:

$$r_x^{\min} + t_x^{\max} = s_x^{\max} \quad \Rightarrow \quad t_x^{\max} = s_x^{\max} - r_x^{\min} \tag{5.7}$$

$$r_y^{\min} + t_y^{\max} = s_y^{\max} \quad \Rightarrow \quad t_y^{\max} = s_y^{\max} - r_y^{\min} \tag{5.8}$$

$$r_y^{\max} + t_y^{\min} = s_y^{\min} \quad \Rightarrow \quad t_x^{\min} = s_x^{\min} - r_x^{\max} \tag{5.9}$$

$$r_y^{\max} + t_y^{\min} = s_y^{\min} \quad \Rightarrow \quad t_y^{\min} = s_y^{\min} - r_y^{\max} \tag{5.10}$$

Es gibt also einen rechteckigen Bereich

$$T = \left\{ \begin{pmatrix} t_x \\ t_y \end{pmatrix} \,\middle|\, t_x^{\min} \le t_x \le t_x^{\max} \text{ und } t_y^{\min} \le t_y \le t_y^{\max} \right\} \tag{5.11}$$

in dem die relevanten Verschiebungsvektoren liegen, und der sich leicht aus den gegebenen Rechtecken R und S berechnen lässt. Im nächsten Schritt muss das entsprechend umschließende achsparallele Parallelogramm im (a,b)-Hilfskoordinatensystem bestimmt werden (c). Zu dessen Bestimmung genügt es, die Ecken des Rechtecks T zu betrachten. Die Menge aller Ecken von T ist

$$C_{xy} = \left\{ \begin{pmatrix} t_x^{\min} \\ t_y^{\min} \end{pmatrix}, \begin{pmatrix} t_x^{\max} \\ t_y^{\min} \end{pmatrix}, \begin{pmatrix} t_x^{\max} \\ t_y^{\max} \end{pmatrix}, \begin{pmatrix} t_x^{\min} \\ t_y^{\max} \end{pmatrix} \right\} \tag{5.12}$$

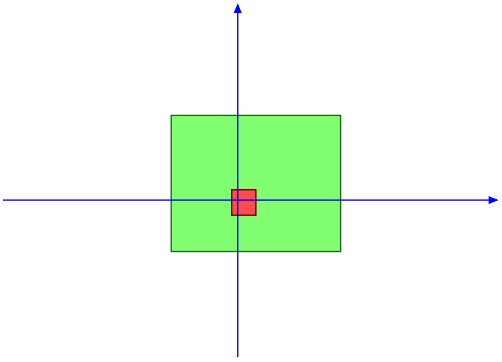
Diese werden in Hilfskoordinaten transformiert

$$C_{ab} = \left\{ \begin{pmatrix} a_x & b_x \\ a_y & b_y \end{pmatrix}^{-1} c \,\middle|\, c \in C_{xy} \right\} \tag{5.13}$$

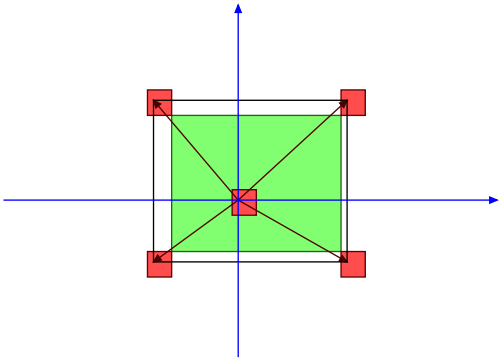
und anschließend in ihre Komponenten zerlegt.

$$C_a = \left\{ a \,\middle|\, \begin{pmatrix} a \\ b \end{pmatrix} \in C_{ab} \right\} \tag{5.14}$$

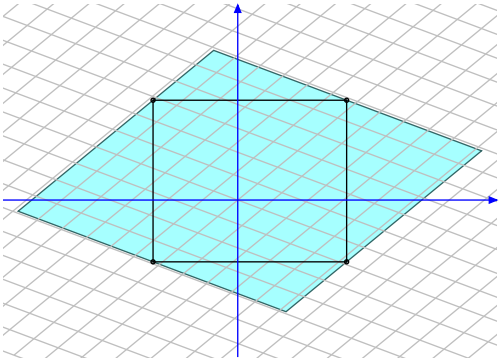
$$C_b = \left\{ b \,\middle|\, \begin{pmatrix} a \\ b \end{pmatrix} \in C_{ab} \right\} \tag{5.15}$$



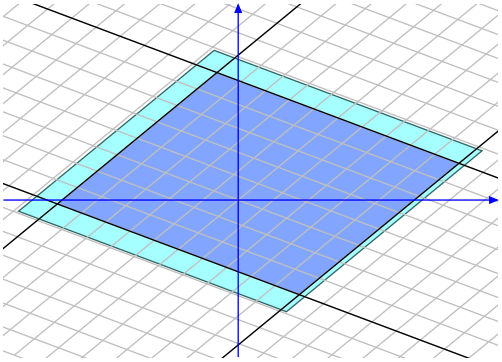
(a) Die Rechtecke R (rot) und S (grün)



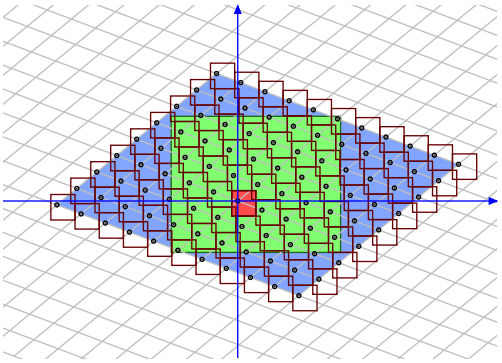
(b) T bestimmt durch maximale Verschiebungen



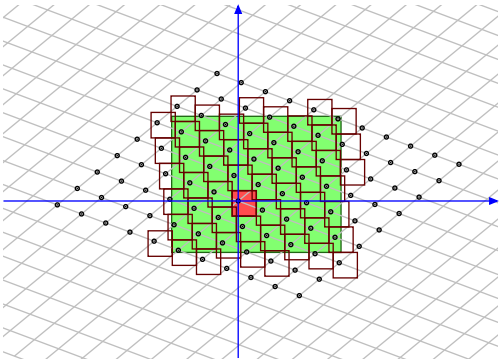
(c) Umschließendes Parallelogramm zu T



(d) Nach innen gerundetes Parallelogramm



(e) Überdeckung von S durch R



(f) Sparsame Überdeckung durch Überprüfung

Abbildung 9: Schritte zur Bestimmung von Überdeckungen

Durch Bestimmung der Minima und Maxima der jeweiligen Mengen ergeben sich Schranken für die Laufvariablen:

$$\min C_a \leq i \leq \max C_a \tag{5.16}$$

$$\min C_b \leq j \leq \max C_b \tag{5.17}$$

Diese Schranken können nach innen gerundet werden, da i und j nur ganzzahlige Werte annehmen sollen (d). Die gerundeten Schranken können anschließend in zwei geschachtelten Schleifen verwendet werden, um garantiert alle Gitterpunkte in T und somit alle relevanten Verschiebungsvektoren zu durchlaufen. Damit erklärt sich auch die Herkunft der in Abbildung 7 auf Seite 24 als Parameter angenommenen Schleifengrenzen.

Wie bereits erwähnt wurde, werden durch diesen Ansatz auch Positionen ermittelt, bei denen die beiden Rechtecke keine Punkte mehr gemeinsam haben. Um unnötige Rechenarbeit zu vermeiden, kann man im Rumpf der Schleife noch überprüfen, ob sich bei dem entsprechenden Verschiebungsvektor die beiden Rechtecke überhaupt schneiden (f). Die Erfahrung zeigt jedoch, dass diese Arbeitseinsparung in den meisten Situationen nicht erforderlich ist.

6 Bestimmung von Fundamentalbereichen

Bei fast allen Symmetriegruppen enthält die Translationszelle aufgrund innerer Symmetrien mehr als eine Kopie des Motivs eines Musters. Ein kleinster Bereich, der genau eine Kopie des Motivs enthält, wird als *Fundamentalbereich* oder *Fundamentalzelle* bezeichnet.

> **Definition 11 (Fundamentalbereich):** Ein Fundamentalbereich einer Symmetriegruppe G ist eine abgeschlossene Teilmenge F der Ebene, so dass die Bilder von F unter den Operationen aus G die gesamte Ebene überdecken und der Schnitt zweier Bilder keine inneren Punkte hat.[26]

Fundamentalbereiche mit fraktalem Rand sind Gegenstand aktueller Forschung. In dieser Arbeit werden sie jedoch nicht näher betrachtet, so dass wir uns im Folgenden auf Bereiche mit abschnittsweise linearem Rand beschränken wollen. Für jeden Orbit der Ebene gilt entweder

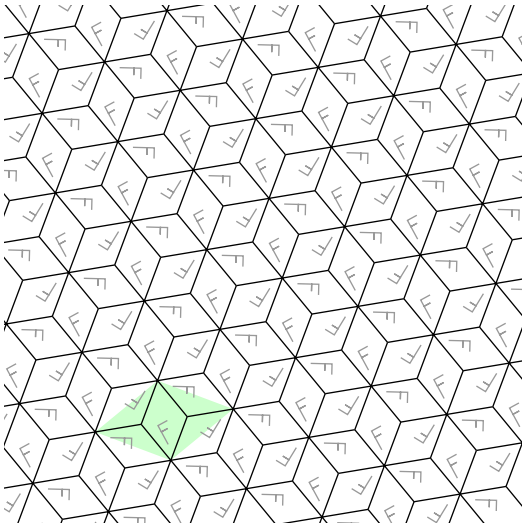
1. genau ein Repräsentant des Orbits liegt im Inneren des Fundamentalbereiches oder
2. mindestens ein Repräsentant des Orbits liegt auf dem Rand des Fundamentalbereiches.

Hat man einen Fundamentalbereich bestimmt, so erhält man durch Anwendung der Gruppensymmetrien auf den Inhalt des Fundamentalbereiches das komplette Muster. Anschaulich ist der Fundamentalbereich die „Einheitsfliese“, mit deren identischen und spiegelverkehrten Kopien die komplette Ebene entsprechend dem Muster gefliest werden kann. In Tabelle 4 auf Seite 20 ist jeweils ein Fundamentalbereich gelb hinterlegt.

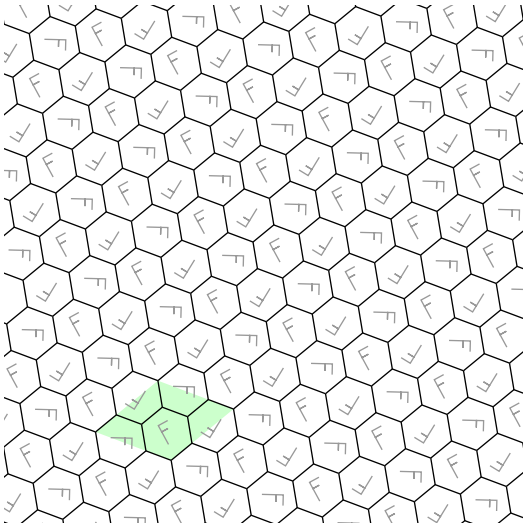
Um ein Muster besser verstehen zu können, ist es hilfreich, sich die Zerlegung in Fundamentalbereiche anzeigen zu lassen. Auch beim Abzeichnen einer Vorlage oder beim Entwurf eines neuen Ornaments kann die Darstellung der Fundamentalbereiche von Vorteil sein. Daher ist die Darstellung einer Parkettierung durch Fundamentalbereiche in die Anwendung integriert und durch den Benutzer aktivierbar.

6.1 Verschiedene Fundamentalbereiche

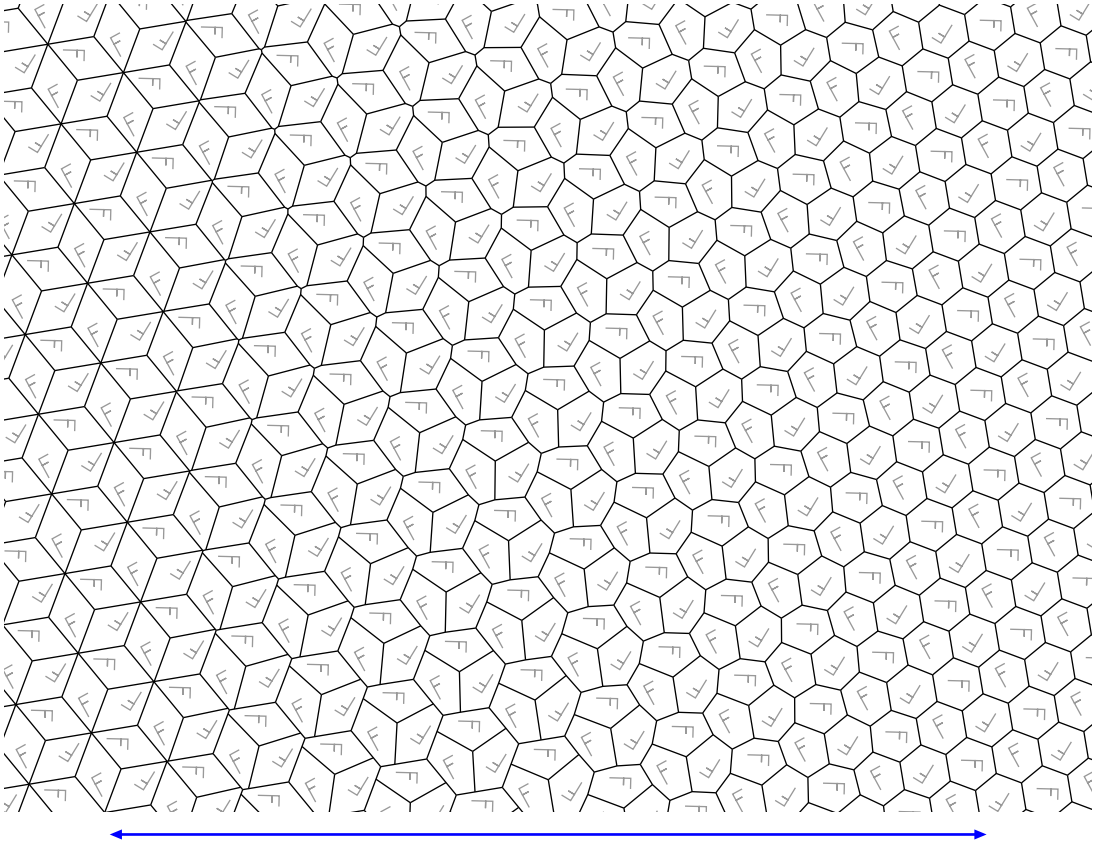
Die Zerlegung der Ebene in Fundamentalbereiche ist nicht eindeutig. Beispielsweise kann die Gruppe p3 sowohl in rautenförmige als auch in regelmäßig sechseckige Fundamentalbereiche zerlegt werden, wie in Abbildung 10 auf der nächsten Seite dargestellt ist. Dazwischen liegt ein



(a) Rauten



(b) regelmäßige Sechsecke



(c) Übergang (aufgrund der Darstellung in einer einzigen Abbildung nur näherungsweise echte Fundamentalzellen)

Abbildung 10: Verschiedene Fundamentalzellen der Gruppe p3

kontinuierlicher Raum von Zerlegungen in unregelmäßige Fünf- oder Sechsecke, angedeutet in Abbildung 10c. Je nach Muster ist oft eine dieser Zerlegung intuitiver als die anderen. Daher soll das Programm die Auswahl unterschiedlicher Zerlegungen erlauben. Dabei sind jedoch Einschränkungen angebracht, welche Arten von Fundamentalzellen leicht zu beschreiben und intuitiv zu navigieren sind. Eine erste sinnvolle Auswahl scheint die Beschränkung auf konvexe Polygone zu sein. Damit fallen viele komplizierte Fälle weg, wie unzusammenhängende Fundamentalbereiche, geschwungene Ränder oder puzzelteilartige Verzahnungen der Fundamentalbereiche.

Eine frühere Version des Programms hatte für viele Gruppen die Fundamentalbereiche entsprechend dieser Vorgabe einstellbar. Dazu wurden eine oder mehrere Ecken des Fundamentalbereiches als durch den Benutzer verschiebbare Kontrollpunkte gekennzeichnet. In Gruppen mit hoher innerer Symmetrie funktioniert dieser Ansatz ganz gut. Bei Gruppen mit niedriger Symmetrie wurde die Zahl der Freiheitsgrade und damit die der Kontrollpunkte immer größer, und die damit konfigurierbaren Fundamentalbereiche waren mitunter weder intuitiv erwartet noch ästhetisch ansprechend. Dadurch wirkte die Steuerung der Fundamentalzellen ziemlich uneinheitlich, und der Ansatz war insgesamt unbefriedigend. Ein sehr viel einheitlicheres Konzept ist die Zerlegung der Ebene in *Voronoi-Polygone*.

6.2 Voronoi-Polygone als Fundamentalbereiche

Definition 12 (Voronoi-Polygon): Sei S eine diskrete Menge von Punkten in der Ebene, und $s \in S$ einer dieser Punkte. Dann ist das zu s gehörige Voronoi-Polygon V_s die Menge aller Punkte der Ebene, die zu keinem anderen Punkt der Menge S einen geringeren Abstand haben als zu s.

$$V_s := \{p \in \mathbb{R}^2 \mid \forall s' \in S : \|p - s\| \leq \|p - s'\|\} \tag{6.1}$$

Die Menge aller Voronoi-Polygone zu einer Punktemenge S wird als das *Voronoi-Diagramm* dieser Punktemenge bezeichnet. Das Voronoi-Polygon einer allgemeinen, nicht symmetrischen Punktemenge ist in Abbildung 11a auf Seite 33 dargestellt. Verwendet man den Orbit eines Punktes, der nicht auf einem Rotationszentrum oder einer Spiegelachse liegt, als Punktemenge, so sind die Voronoi-Polygone Fundamentalzellen des Musters. Der Punkt, dessen Orbit das Voronoi-Diagramm definieren soll, kann als Kontrollpunkt vom Benutzer positioniert werden.

Die Zerlegung in Voronoi-Polygone ist ein sehr einheitliches Konzept. Unabhängig von der gewählten Gruppe wird der Fundamentalbereich durch einen einzigen Kontrollpunkt, also durch zwei Freiheitsgrade, festgelegt. Der Zusammenhang zwischen Kontrollpunkt und Fundamentalzelle ist zwar nicht ganz so direkt wie beim Verschieben einzelner Eckpunkte, aber dennoch hinreichend intuitiv. Der Kontrollpunkt ist in der Translationszelle frei positionierbar; dadurch kann seine Position bei einem Wechsel der Symmetriegruppe beibehalten werden.

Beim Experimentieren mit verschiedenen durch diesen Ansatz erreichbaren Fundamentalzellen fällt auf, dass einige ästhetisch ansprechender wirken als andere. Manche dieser ansprechenden Gitter ergeben sich, wenn der Kontrollpunkt auf bestimmten festen (a,b)-Koordinaten liegt. Viele dieser Positionen wurden von Hand ermittelt und in das Programm eingespeichert, so dass der Kontrollpunkt beim Bewegen auf diese Punkte oder Geraden einrastet.

In manchen Gruppen niederer Symmetrie hängt die Position des Kontrollpunktes zur Erreichung besonders ästhetischer Fundamentalzellen-Parkettierungen auch von der Form der Translationszelle ab, so dass keine festen (a,b)-Koordinaten dafür angegeben werden können. Ein interessantes Ziel für die Zukunft wäre ein Einrasten des Kontrollpunktes auch für diese Situationen. Um dies programmatisch zu lösen, müsste die Ästhetik einer Fundamentalzellen-Parkettierung mathematisch bewertet werden. Die folgenden Faktoren scheinen dabei eine Rolle zu spielen:

Degenerierte Fälle, bei denen Punkte im Voronoi-Diagramm zusammenfallen, so dass Fundamentalzellen weniger als sechs Ecken haben, oder an einer Ecke mehr als drei Fundamentalzellen zusammenstoßen.

Symmetrie der Form einer einzelnen Fundamentalzelle oder der gesamten Fundamentalzellen-Parkettierung, die über die von der Gruppe vorgegebenen Symmetrieeigenschaften hinaus geht.

Koinzidenz einzelner Segmente der Fundamentalzellen-Parkettierung mit den Rändern der Translationszelle.

Zunächst müssen jedoch noch Erfahrungen gesammelt werden, welche Parkettierungen von verschiedenen Betrachtern als wie harmonisch empfunden werden. Erst dann kann versucht werden, ein Modell für diese Bewertungen zu entwickeln.

Manche ästhetisch interessanten Voronoi-Parkettierungen ergeben sich als Grenzfall, bei dem der Kontrollpunkt mit einem seiner Bildpunkte zusammenfällt. Geschieht dies an einem Drehzentrum, so ist die Richtung der an dieser Stelle notwendigen Trennkante nicht eindeutig definiert. Sie hängt von der im Grenzwertprozess verwendeten Annäherungsrichtung ab. In der derzeitigen Implementierung bedeutet das, dass der Benutzer an diesen Stellen entweder noch vor dem erreichen des Grenzfalls stehen bleiben muss, oder aber praktisch keine Kontrolle über diesen Aspekt der Parkettierung hat. Ein Mechanismus, der dem Benutzer auf intuitive Weise die Festlegung der Trennungsrichtung auch im absoluten Grenzfall erlaubt, wird noch gesucht.

6.3 Algorithmische Bestimmung der Voronoi-Polygone

Bis hier wurden Voronoi-Polygone als geeignete Fundamentalzellen aus Benutzersicht dargestellt. Der folgende Abschnitt befasst sich mit der Berechnung dieser Diagramme.

Eine duale Sicht auf das Problem der Voronoi-Diagramme ist das Problem der *Delaunay-Triangulierung*:

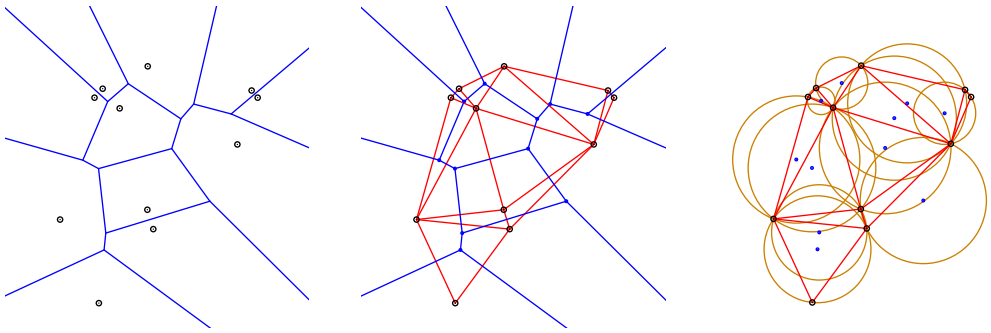
> **Definition 13 (Delaunay-Dreieck):** Ein Delaunay-Dreieck zu einer Punktemenge S ist ein Dreieck, dessen Ecken Punkte aus S sind und das einen Umkreis hat, in dessen Innerem kein Punkt der Menge S liegt.

> **Definition 14 (Delaunay-Triangulierung):** Eine Delaunay-Triangulierung zu einer Punktemenge S ist eine Menge von Delaunay-Dreiecken zu S, die untereinander keine inneren Punkte gemeinsam haben, und die zusammen die konvexe Hülle von S überdecken.

Die Voronoi-Polygone zweier Punkte, die in der Delaunay-Triangulierung eine Dreieckskante bestimmen, berühren sich. Die Ecken der Voronoi-Polygone entsprechen den Umkreismittelpunkten der Delaunay-Dreiecke, da dies diejenigen Punkte sind, die zu mindestens drei Punkten aus S den gleichen Abstand haben. Dieser Zusammenhang ist in Abbildung 11 auf der nächsten Seite für eine allgemeine, nicht symmetrische Punktemenge S veranschaulicht.

Liegen mehr als drei Punkte der Menge auf einem Kreis (kozirkular), oder drei Punkte auf einer Gerade (kollinear), so ist die Delaunay-Triangulierung eventuell nicht eindeutig. Die zu den nicht eindeutig bestimmten Delaunay-Kanten dualen Voronoi-Kanten haben jedoch die Länge 0, so dass die Voronoi-Polygone von der Entscheidung, welche Delaunay-Dreiecke gebildet werden, unabhängig sind.

Es gibt viele Publikationen darüber, wie man Voronoi-Diagramme zu endlichen Punktemengen bestimmt, beispielsweise durch einen Sweepline-Algorithmus[5]. Im vorliegenden Fall haben



(a) Voronoi-Diagramm

(b) Voronoi-Diagramm und Delaunay-Triangulierung

(c) Delaunay-Triangulierung mit Umkreisen

Abbildung 11: Voronoi-Zellen und Delaunay-Triangulierung

wir es jedoch mit dem Orbit des Kontrollpunktes unter einer unendlichen Gruppe zu tun, also mit unendlich vielen Punkten. Relevant ist nur der Ausschnitt des Diagramms, der innerhalb der Standard-Translationszelle liegt. Allerdings können auch Punkte außerhalb der Standard-Translationszelle Einfluss auf das Diagramm in dieser Zelle haben. Ein erster Schritt ist daher die Ermittlung einer oberen Schranke für den Einflussbereich einzelner Punkte.

6.3.1 Schranke für den Einflussbereich

Bei Überlegungen zu diesem Einflussbereich erweisen sich häufig intuitiv plausible Argumente als falsch. So genügt es beispielsweise nicht immer, nur die benachbarten acht Translationszellen mit zu betrachten, die die Standard-Translationszelle über Kante oder Ecke berühren. Um derartige Fehler zu vermeiden, wurde der folgende Abschnitt als Beweis strukturiert. Die einzelnen Schritte, die im Programm zur Berechnung des Einflussbereiches durchgeführt werden, sind auch in Abbildung 12 auf der nächsten Seite illustriert.

Betrachten wir zunächst alle Gitterpunkte, die von den beiden Translationsvektoren aufgespannt werden:

$$\{ia + jb \,|\, i,j \in \mathbb{Z}\} \tag{6.2}$$

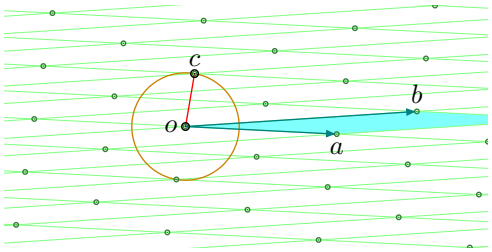
Wir suchen jetzt für dieses regelmäßige Punktegitter eine gültige Delaunay-Triangulierung. Als ersten Punkt des ersten Delaunay-Dreiecks wählen wir den Koordinatenursprung $o := (0,0)$. Als zweiten Punkt wählen wir einen Punkt c, der zu o minimalen Abstand hat. Das ist zwar meist einer der Vektoren $\pm a$ oder $\pm b$, kann bei stärker verzerrten Translationszellen jedoch durchaus auch eine Linearkombination sein, wie der Punkt $c = 2b - 3a$ in Abbildung 12a auf der nächsten Seite zeigt. Allgemein gilt also

$$c = c_a a + c_b b \quad c_a, c_b \in \mathbb{Z} \tag{6.3}$$

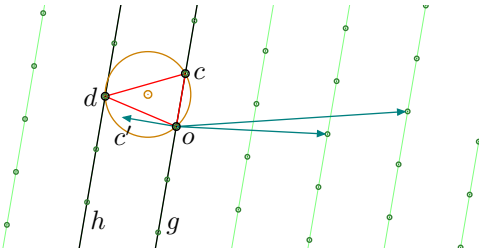
Der Abstand $\|o - c\|$ ist nicht nur der minimale Abstand des Ursprungs zu einem anderen Gitterpunkt, sondern aufgrund der Regelmäßigkeit des Gitters der kleinste Abstand, der überhaupt in diesem Gitter auftritt.

Lemma 1: Haben zwei Punkte der Menge minimalen Abstand zueinander, so gibt es eine Delaunay-Triangulierung, bei der diese beiden Punkte durch eine Dreieckskante verbunden sind.

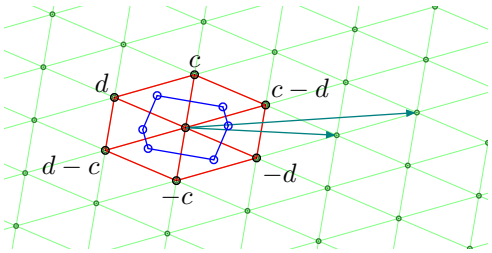
Beweis: Bezeichnen wir die beiden Punkte mit minimalem Abstand der Anwendung entsprechend als o und c, und betrachten wir die Menge $[oc]$ aller Punkte auf der Verbindungsstrecke.



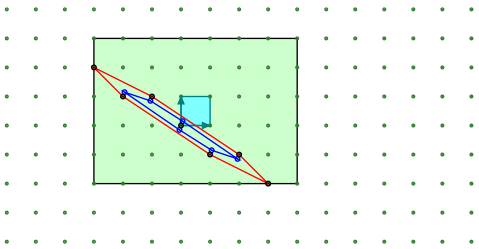
(a) Der Punkt c als Gitterpunkt mit minimalem Abstand



(b) d liegt auf h, der nächsten Parallelen zu g



(c) Sechs Nachbarpunkte und das durch sie bestimmte Voronoi-Polygon



(d) Verzerrte Darstellung, in der das Gitter kartesisch erscheint, mit beschränktem Einflussbereich (grün)

Abbildung 12: Schritte zur Beschränkung des Einflussbereiches

Die Punkte gehören jeweils zu einem oder mehreren Voronoi-Polygonen, je nachdem zu welchen Punkten aus S sie minimalen Abstand haben. Angenommen ein Punkt p auf der Strecke hätte zu einem dritten, von o und c verschiedenen, Punkt q aus S geringeren Abstand als zu c, wie es in 13 dargestellt ist.

$$\|p - q\| < \|p - c\| \tag{6.4}$$

Da der Punkt auf der Strecke $[oc]$ liegt, gilt weiterhin

$$\|p - o\| + \|p - c\| = \|o - c\| \tag{6.5}$$

Aus der Dreiecksungleichung sowie Gleichungen (6.4) und (6.5) folgt

$$\|q - o\| \leq \|p - q\| + \|p - o\| < \|p - c\| + \|p - o\| = \|o - c\| \tag{6.6}$$

Dies steht im Widerspruch zur Annahme, dass $\|o - c\|$ minimal ist. Der Fall $\|p - q\| < \|p - o\|$ ist analog zu behandeln.

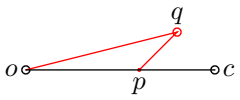


Abbildung 13: Illustration zum Beweis von Lemma 1

Es kann also keinen Punkt q geben, zu dem Punkte auf $[oc]$ geringeren Abstand als zu o oder c haben. Deswegen sind alle Punkte auf der Verbindungsstrecke Teil der Voronoi-Polygone von o oder c, in der Mitte der Strecke auch beider zusammen. Andere Voronoi-Polygone können die

Strecke zwar berühren, aber nicht in ihrem Inneren enthalten. Die Voronoi-Polygone von o und c berühren sich also. Deshalb gibt es eine Delaunay-Triangulierung, in der diese beiden Punkte die Enden einer Kante der Delaunay-Triangulierung sind. □

Als nächstes benötigen wir d, den dritten Eckpunkt des ersten Dreiecks. Dieser soll nicht auf der Geraden g liegen, die von o und c definiert wird. Andererseits muss er hinreichend nah an o und c liegen, damit am Ende die Umkreisbedingung erfüllt ist. Man kann sich unendlich viele zu g parallele Geraden denken, auf denen in gleichmäßigem Abstand die Gitterpunkte liegen. Für einen hinreichend nahen Punkt d kommt nur eine direkt zu g benachbarte Gerade in Frage. Nennen wir eine solche zu g parallele Gerade mit minimalem Abstand h.

Lemma 2: Ein Punkt $d' = d'_a a + d'_b b$, der die Gleichung $|d'_b c_a - d'_a c_b| = \mathrm{ggT}(c_a, c_b)$ erfüllt, liegt auf einer von g verschiedenen Gitterparallelen zu g mit minimalem Abstand zu o und somit zu g.

Beweis: Der Abstand einer Geraden zum Ursprung wird senkrecht zu c gemessen. Definieren wir c' als einen solchen senkrechten Vektor, und drücken ihn gemäß Gleichung 6.3 durch die Komponenten von a und b aus:

$$c' := \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} c = \begin{pmatrix} -c_y \\ c_x \end{pmatrix} = \begin{pmatrix} -c_a a_y - c_b b_y \\ c_a a_x + c_b b_x \end{pmatrix} \tag{6.7}$$

Der Betrag des Skalarprodukts $\langle p,c' \rangle$ eines beliebigen Vektors p mit c' ist proportional zum Abstand, den die durch p verlaufende Gerade zu g hat. Ist p ein Gittervektor, so gilt unter Verwendung von Gleichung 6.7:

$$\langle p,c' \rangle = \langle ia + jb,c' \rangle = i \langle a,c' \rangle + j \langle b,c' \rangle \tag{6.8}$$

$$\langle a,c' \rangle = -a_x c_a a_y - a_x c_b b_y + a_y c_a a_x + a_y c_b b_x = -c_b \begin{vmatrix} a_x & b_x \\ a_y & b_y \end{vmatrix} \tag{6.9}$$

$$\langle b,c' \rangle = -b_x c_a a_y - b_x c_b b_y + b_y c_a a_x + b_y c_b b_x = c_a \begin{vmatrix} a_x & b_x \\ a_y & b_y \end{vmatrix} \tag{6.10}$$

$$\langle p,c' \rangle = (-ic_b + jc_a) \begin{vmatrix} a_x & b_x \\ a_y & b_y \end{vmatrix} \tag{6.11}$$

Gitterpunkte mit Abstand 0 liegen auf g. Wir sind also interessiert an Punkten mit minimalem positivem Abstandsbetrag. Die Determinante stellt in dieser Gleichung einen konstanten Faktor dar und kann bei der Suche nach dem minimalen Abstand ignoriert werden. Das Minimum für den verbleibenden Term ist der größte gemeinsame Teiler der ganzen Zahlen c_a und c_b:

$$\min\{|jc_a - ic_b| \mid i,j \in \mathbb{Z} \text{ und } jc_a - ic_b \neq 0\} = \mathrm{ggT}(c_a,c_b) \tag{6.12}$$

□

Lemma 3: Ein Gitterpunkt d' kann berechnet werden, so dass $h := \{d' + \lambda c \mid \lambda \in \mathbb{R}\}$ eine Parallele zu g ist, die unter all den durch Gitterpunkte verlaufenden Parallelen minimalen Abstand hat.

Beweis: Um die Gleichung $d'_b c_a - d'_a c_b = \mathrm{ggT}(c_a, c_b)$ nach d'_a und d'_b aufzulösen kann der erweiterte euklidische Algorithmus verwendet werden[28]. Der so erhaltene Punkt d' ist ein Gitterpunkt, da der erweiterte euklidische Algorithmus ganzzahlige Lösungen ermittelt. Die Minimalität des Abstandes folgt dann aus Lemma 2. □

Eingabe: Zwei ganze Zahlen, a und b

Ausgabe: Ein Array aus drei Zahlen, g, x und y, so dass

1. $g = \mathrm{ggT}(a, b)$
2. $g = xa + yb$
3. $g \geq 0$

```
int[] extendedEuclid (final int a, final int b) {
    int tmp, a1, x1, y1, a2, x2, y2;
    if (Math.abs(a) >= Math.abs(b)) {
        a1 = a; x1 = 1; y1 = 0;
        a2 = b; x2 = 0; y2 = 1;
    }
    else {
        a1 = b; x1 = 0; y1 = 1;
        a2 = a; x2 = 1; y2 = 0;
    }
    assert Math.abs(a1) >= Math.abs(a2);
    assert a1 == x1*a + y1*b;
    assert a2 == x2*a + y2*b;
    while (a2 != 0) {
        int q = a1 / a2;
        tmp = a1; a1 = a2; a2 = tmp - q*a2;
        tmp = x1; x1 = x2; x2 = tmp - q*x2;
        tmp = y1; y1 = y2; y2 = tmp - q*y2;
        assert a1 == x1*a + y1*b;
        assert a2 == x2*a + y2*b;
    }
    if (a1<0) return new int[] { -a1, -x1, -y1 };
    else return new int[] { a1, x1, y1 };
}
```

Abbildung 14: Erweiterter euklidischer Algorithmus

Es gibt viele Texte, die das Konzept des erweiterten euklidischen Algorithmus abstrakt erläutern. Einfache, verständliche und effiziente Implementierungen, die nicht über ihre Zwischenschritte Buch führen müssen, sind hingegen schwerer zu finden. Daher ist in Abbildung 14 auf der vorherigen Seite die verwendete Implementierung als Java-Code angegeben. Die Invarianten sind durch Assertions ausgedrückt.

Lemma 4: Die Gitterpunkte auf h sind $\{d' + ic \mid i \in \mathbb{Z}\}$.

Beweis: Dass die Punkte $d' + ic$ Gitterpunkte sind folgt daraus, dass sowohl d' als auch c Gittervektoren sind. Da außerdem der Vektor c minimale Länge hat, können zwischen diesen Gitterpunkten keine weiteren Gitterpunkte liegen. □

Ausgehend von d', einem beliebigen Gitterpunkt auf h, kann jetzt d gesucht werden, ein geeigneter Gitterpunkt auf h, der mit o und c ein gültiges Voronoi-Polygon bildet. Der gesuchte Punkt sollte möglichst nah bei sowohl o als auch c liegen, weshalb ein minimaler Abstand zum Mittelpunkt der Strecke $[oc]$ eine sinnvolle Forderung ist.

Lemma 5: Der Punkt

$$d := d' + \left\lceil -\frac{\langle d',c\rangle}{\langle c,c\rangle} \right\rceil c \tag{6.13}$$

ist ein Gitterpunkt auf h und hat von all diesen Punkten den minimalen Abstand zum Mittelpunkt der Strecke $[oc]$.

Beweis: Betrachten wir zunächst den Punkt (nicht unbedingt Gitterpunkt) auf der Geraden h, der minimalen Abstand zu o hat. Der Vektor dieses Punktes steht senkrecht auf c.

$$\langle d' + \lambda c,c\rangle = 0 \tag{6.14}$$

$$\langle d',c\rangle + \lambda \langle c,c\rangle = 0 \tag{6.15}$$

$$\lambda = -\frac{\langle d',c\rangle}{\langle c,c\rangle} \tag{6.16}$$

Da der gesuchte Punkt minimalen Abstand zur Mitte der Strecke haben soll, addieren wir noch einmal $\frac{c}{2}$, indem wir λ um $\frac{1}{2}$ erhöhen. Der so erhaltene Wert muss zum nächsten Gitterpunkt gerundet werden. Da sich die Rundung zur nächsten Ganzzahl auch als Subtraktion von $\frac{1}{2}$ mit anschließendem Aufrunden schreiben lässt, bleibt im Ergebnis nur das Aufrunden übrig.

$$\left\lceil \left(\lambda + \frac{1}{2}\right) - \frac{1}{2} \right\rceil = \lceil \lambda \rceil \tag{6.17}$$

□

Der so ermittelte Punkt d auf der Geraden h ist auch in Abbildung 12b auf Seite 34 dargestellt.

Lemma 6: Der Umkreismittelpunkt des Dreiecks ocd liegt im Inneren oder auf dem Rand dieses Dreiecks.

Beweis: Das Dreieck hat keine stumpfen Innenwinkel. Die Winkel bei o und c können nicht stumpf sein, da sonst ein anderer Gitterpunkt auf h geringeren Abstand zum Mittelpunkt der Strecke $[oc]$ hätte als d. Auch der Winkel bei d kann nicht stumpf sein, da sonst der Abstand $\|o - d\|$ kleiner wäre als der Abstand $\|o - c\|$, was der Annahme widersprechen würde, dass c ein Punkt mit minimalem Abstand zu o ist. Der Umkreismittelpunkt eines rechtwinkligen Dreiecks liegt in der Mitte der Hypotenuse, der eines Dreiecks mit drei spitzen Winkeln liegt im Inneren des Dreiecks. □

Lemma 7: Der Abstand D_{gh} zwischen den Geraden g und h beträgt mindestens $\frac{\sqrt{3}}{2}D_{oc}$, wobei $D_{oc} := \|o - c\|$.

$$D_{gh} \geq \frac{\sqrt{3}}{2}D_{oc} \tag{6.18}$$

Beweis: $\|o - c\| \leq \|o - d\|$ und $\|o - c\| \leq \|c - d\|$, da der Abstand $\|o - c\|$ definitionsgemäß minimal ist. Unter all den nicht-stumpfwinkligen Dreiecken über $[oc]$, die diese Ungleichungen erfüllen, hat das gleichseitige Dreieck, das beide Ungleichungen mit Gleichheit erfüllt, die minimale Höhe. Dieses Dreieck ist in Abbildung 15 dargestellt. Der Abstand der Geraden g und h zueinander entspricht der Höhe eines solchen Dreiecks. Die Höhe im gleichseitigen Dreieck beträgt das $\frac{\sqrt{3}}{2}$-fache der Grundseite. □

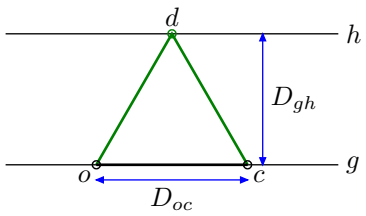


Abbildung 15: Illustration zum Beweis von Lemma 7

Lemma 8: Der Umkreis um ocd schneidet außer g und h keine weitere zu g parallele Gerade, auf der Gitterpunkte liegen.

Beweis: Betrachten wir D_{oc}, den Abstand zwischen o und c, sowie D_{gh}, den Abstand zwischen g und h, als gegeben. Durch Umformung von Gleichung 6.18 gilt für diese Abstände die Ungleichung

$$D_{oc} \leq \sqrt{\frac{4}{3}}D_{gh} \tag{6.19}$$

Setzen wir außerdem gemäß Lemma 6 voraus, dass das Dreieck ocd keine stumpfen Innenwinkel besitzt und somit der Umkreismittelpunkt im Inneren des Dreiecks liegt. Unter allen Dreiecken, die diese Bedingungen erfüllen, haben rechtwinklige Dreiecke den maximalen Umkreisradius. Für den Durchmesser D_k des Umkreises gilt daher unter Verwendung von Gleichung 6.19

$$D_k^2 \leq D_{oc}^2 + D_{gh}^2 \leq \left(\frac{4}{3} + 1\right) D_{gh}^2 \tag{6.20}$$

$$D_k \leq \sqrt{\frac{7}{3}}D_{gh} < 2D_{gh} \tag{6.21}$$

Alle zu g Parallelen Geraden im Gitter haben zu ihren nächsten Nachbarn jeweils den gleichen Abstand D_{gh}. Da der Durchmesser des Kreises geringer ist als der Abstand zwischen einer dieser Parallelen und der übernächsten, kann der Kreis höchstens zwei Gittergeraden schneiden. Er schneidet bereits g und h, weshalb er keine weiteren Geraden mehr schneiden kann. □

Satz 9: Das Dreieck ocd ist ein gültiges Delaunay-Dreieck.

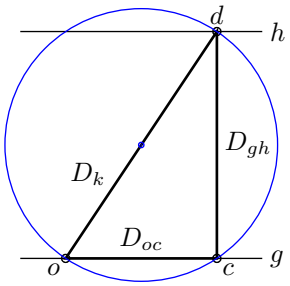


Abbildung 16: Illustration zum Beweis von Lemma 8

Beweis: Um zu zeigen, dass das Dreieck ein gültiges Delaunay-Dreieck ist, muss argumentiert werden, dass die Umkreisbedingung immer erfüllt ist, dass also keine anderen Gitterpunkte im Umkreis k dieses Dreiecks liegen können. Unterscheiden wir verschiedene Fälle, auf welcher zu g parallelen Gerade ein potentieller Punkt p im Kreisinneren liegen könnte, und führen dies jeweils zu einem Widerspruch.

1. p liegt auf g selbst. Da d nicht auf g liegt, ist der Umkreis wohl definiert und nicht zu einer Geraden entartet. Da o und c auf dem Kreisrand liegen, befinden sich alle Punkte von g, die im Inneren von k liegen, zwischen o und c. keiner dieser Punkte kann ein Gitterpunkt sein, weil dies der Behauptung, dass der Abstand $\|o - c\|$ minimal sei, widersprechen würde.

2. p liegt auf h. Da der Mittelpunkt des Umkreises irgendwo auf der Mittelsenkrechten von oc liegen muss, ist der zweite Schnittpunkt von k mit h das Spiegelbild d'' von d an eben dieser Mittelsenkrechten. Diese Situation ist in Abbildung 17 dargestellt. Betrachten wir den Abstand der beiden Schnittpunkte, so ergeben sich zwei Fälle.

   (a) $\|d - d''\| \leq \|o - c\|$. Dann kann dazwischen kein weiterer Gitterpunkt liegen, weil dies der Minimalität von $\|o - c\|$ widersprechen würde.

   (b) $\|d - d''\| > \|o - c\|$. Dann läge dazwischen ein Punkt, der näher am Mittelpunkt von $[oc]$ liegen würde als d. Dies widerspräche der Konstruktion von d.

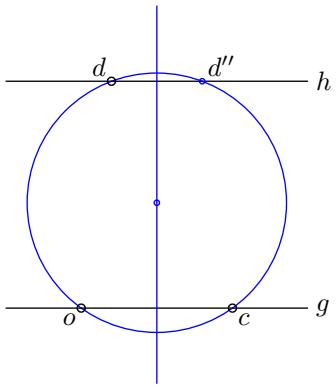


Abbildung 17: Illustration zum Fall 2 des Beweises von Satz 9

3. p liegt auf einer anderen Parallelen zu g. Dann müsste k diese andere Parallele schneiden, was in Lemma 8 widerlegt wurde.

Da kein weiterer Punkt im Inneren des Umkreises liegen kann, ist das Dreieck ocd ein gültiges Delaunay-Dreieck. □

Lemma 10: Spiegelung des Dreiecks ocd an einem seiner Seitenmittelpunkte führt wieder zu einem gültigen Delaunay-Dreieck.

Beweis: Das Gitter selbst hat mindestens die Symmetriegruppe p2 und ist daher punktsymmetrisch bezüglich Spiegelung an jeder ganz- oder halbzahligen Kombination der beiden Gittervektoren. Dies sieht man auch, wenn man einen allgemeinen Gitterpunkt p an einem allgemeinen möglichen Spiegelpunkt r spiegelt.

$$p = p_a a + p_b b \qquad p_a, p_b \in \mathbb{Z} \tag{6.22}$$

$$r = \frac{r_a}{2} a + \frac{r_b}{2} b \qquad r_a, r_b \in \mathbb{Z} \tag{6.23}$$

$$p' = r + (r - p) = (r_a - p_a)a + (r_b + p_b)b \qquad (r_a - p_a),(r_b - p_b) \in \mathbb{Z} \tag{6.24}$$

Wenn nach der Spiegelung ein Punkt des Gitters im Umkreis läge, so müsste vor der Spiegelung bereits das zu diesem Punkt gehörige Urbild im Umkreis gelegen haben. Dies widerspräche der in Satz 9 bewiesenen Tatsache, dass das Dreieck ocd ein gültiges Delaunay-Dreieck ist. □

Lemma 11: Kongruente Bilder des Dreiecks ocd, erhalten durch wiederholte Punktspiegelung an den Seitenmitten, stellen eine zulässige Delaunay-Triangulierung des Gitters dar.

Beweis: Es genügt eine Spiegelung an $\frac{c}{2}$, dem Mittelpunkt der Strecke $[oc]$, um ein zweites Dreieck $o(c-d)c$ zu erzeugen. Dieses Dreieck bildet zusammen mit ocd ein Parallelogramm, das von den Vektoren c und d aufgespannt wird und das man als Translationszelle verwenden kann. Verschiebt man diese um alle ganzzahligen Kombinationen von $c - d$ und d, so überdecken die Bilder die Ebene. Das Ergebnis entspricht dem, das durch Punktspiegelungen an den Seitenmitten entsteht. Die Umkreiseigenschaft bleibt dabei erhalten, weil das Punktegitter selbst die gleiche Translationssymmetrie aufweist. Überschneidungen zweier Dreiecke sind ausgeschlossen, da sich die Parallelogramme nur am Rand berühren. □

Die so entstehende Delaunay-Triangulierung ist auch in Abbildung 12c auf Seite 34 dargestellt. Bisher haben wir Voronoi-Polygone für die Symmetriegruppe p1 ermittelt, genauer gesagt für die Ecken der Translationszellen, da wir uns in Gleichung 6.2 auf Seite 33 auf ein von zwei Translationen gebildetes Gitter beschränkt haben. Im Folgenden sollen auch die inneren Symmetrien betrachtet werden, die in Abschnitt 4.3 auf Seite 23 bereits eine Rolle gespielt haben.

Lemma 12: Der Orbit eines Kontrollpunktes p unter einer Symmetriegruppe G besteht aus der Überlagerung mehrerer gegeneinander verschobener Punktegitter mit den Gittervektoren a und b.

Beweis: Berechnet man den Orbit wie in Abschnitt 4 auf Seite 21 beschrieben, so wird der Kontrollpunkt in die Standardzelle verschoben, mit einer inneren Symmetrie abgebildet, und anschließend mit allen ganzzahligen Vielfachen von a und b verschoben. Daher entsteht zu jeder inneren Symmetrie ein entsprechend verschobenes Punktegitter. □

Lemma 13: Bei der Kombination von mehreren gegeneinander verschobenen Punktegittern werden die einzelnen Voronoi-Polygone niemals größer, als sie es bei einem einzelnen Gitter wären.

Beweis: Betrachten wir das Voronoi-Polygon eines Punktes s unter Berücksichtigung aller Gitter. Alle Punkte, die in dem einzelnen Gitter, zu dem s gehört, einem anderen Gitterpunkt als s näher waren, sind dies bei Berücksichtigung zusätzlicher Gitter immer noch. Daher können keine zusätzlichen Punkte zu V_s hinzugekommen sein. Punkte, die zuvor zu s minimalen Abstand hatten, können jedoch zu einem der aus den anderen Gittern hinzugekommenen Gitterpunkte einen noch geringeren Abstand haben. Daher kann das Voronoi-Polygon von s kleiner werden, nicht jedoch größer. □

Satz 14: Alle Punkte eines Orbits, die das Voronoi-Diagramm dieses Orbits in der Standard-Translationszelle beeinflussen können, liegen in dem Parallelogramm

$$\{\lambda_a a + \lambda_b b \mid -m_a \leq \lambda_a \leq m_a + 1 \text{ und } -m_b \leq \lambda_b \leq m_b + 1\} \tag{6.25}$$

mit

$$m_a := \max\{|c_a|, |d_a|, |c_a - d_a|\} \quad \text{und} \quad m_b := \max\{|c_b|, |d_b|, |c_b - d_b|\} \tag{6.26}$$

Beweis: Der Ursprung wird in der in Lemma 11 definierten Triangulierung von 6 Delaunay-Dreiecken umgeben. Die zugehörigen Eckpunkte sind $c, d, d - c, -c, -d$ und $c - d$, wie in Abbildung 12c auf Seite 34 ersichtlich. Dies sind die Punkte, die bei einem einzigen Gitter die Voronoi-Zelle um den Ursprung beeinflussen. Das zu den Gittervektoren a und b parallele umschließende Parallelogramm ist

$$\{\lambda_a a + \lambda_b b \mid -m_a \leq \lambda_a \leq m_a \text{ und } -m_b \leq \lambda_b \leq m_b\} \tag{6.27}$$

Der Bereich aller Punkte, die das Voronoi-Polygon um einen beliebigen, vom Ursprung verschiedenen Punkt $p = p_a a + p_b b$ bestimmen, ist beschränkt durch ein entsprechend verschobenes Parallelogramm.

$$\{(\lambda_a + p_a)a + (\lambda_b + p_b)b \mid -m_a \leq \lambda_a \leq m_a \text{ und } -m_b \leq \lambda_b \leq m_b\} \tag{6.28}$$

Da lediglich die Polygone um Punkte in der Standard-Translationszelle von Bedeutung sind, genügt eine Betrachtung von Verschiebungen im Bereich $0 \leq p_a, p_b \leq 1$. Die Vereinigung all dieser Parallelogramme ist das im Gleichung 6.25 angegebene Parallelogramm, das auch in Abbildung 12d auf Seite 34 dargestellt ist. Dieses liegt punktsymmetrisch um die Translationszelle, während das Parallelogramm von Gleichung 6.27 noch punktsymmetrisch um den Ursprung lag.

Kommen weitere Gitter hinzu, so kann gemäß Lemma 13 das Voronoi-Polygon eines Punktes höchstens schrumpfen. Da die Grenzen des Polygons auf der Mittelsenkrechten der Verbindungskanten zweier benachbarter Punkte liegen, können die für die neuen Grenzen verantwortlichen Punkte nicht außerhalb der Schranken liegen, die für ein einzelnes Gitter gelten. □

Insgesamt wurde bisher aus den Vektoren a und b ein Bereich bestimmt, in dem sich alle für das Voronoi-Diagramm in der Standard-Translationszelle relevanten Punkte befinden.

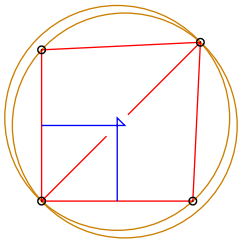
6.3.2 Berechnung der Polygone

Nachdem mit Satz 14 auf der vorherigen Seite Schranken für die Laufvariablen ermittelt worden sind, können durch den in Abschnitt 4 auf Seite 21 beschriebenen Algorithmus alle relevanten Elemente des Orbits des Kontrollpunktes bestimmt werden. Damit liegt nun eine endliche Punktemenge S vor, und man könnte theoretisch mit einem allgemeinen Algorithmus das Voronoi-Diagramm dazu berechnen. Die durch die Symmetrien gegebene hohe Zahl von kollinearen oder kozirkularen Punkten macht jedoch viele Implementierungen instabil. Günstiger ist es, ein einziges Voronoi-Polygon zu berechnen, und dann die Symmetrien auszunutzen, um dieses zu replizieren.

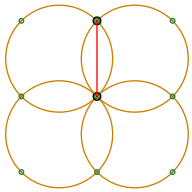
Der Algorithmus berechnet das Voronoi-Polygon um den eigentlichen Kontrollpunkt p, also das Urbild des Orbits: Zunächst werden alle Punkte der Menge $S \setminus \{p\}$ um p herum entsprechend ihres Winkels sortiert. Ein Punkt mit minimalem Abstand kann gemäß Lemma 1 als Endpunkt der ersten Delaunay-Kante verwendet werden.

Nun werden entsprechend der Sortierung weitere Punkte als möglicher dritter Punkt eines Delaunay-Dreiecks in Betracht gezogen, und das Umkreisprädikat geprüft. Ist das Dreieck zulässig, wird der dazu verwendete Nachbarpunkt gespeichert und die Kante zu ihm im nächsten Durchlauf der Schleife als Ausgangskante für die Suche nach dem nächsten Dreieck verwendet. Der Prozess wird so lange fortgesetzt, bis der erste Nachbarpunkt wieder erreicht ist.

Es ist darauf zu achten, dass das Umkreisprädikat aufgrund numerischer Probleme im Allgemeinen nicht exakt ausgewertet werden kann. Es empfiehlt sich, das Prädikat eher etwas großzügiger zu implementieren, also lieber unzulässige Dreiecke anzuerkennen als zulässige abzulehnen. Ein fälschlich anerkanntes unzulässiges Dreieck führt zu einem kleinen, im Allgemeinen nicht wahrnehmbaren, Fehler in der Form des Polygons. Eine zu restriktive Implementierung hingegen kann dazu führen, dass gerade bei regelmäßigen kartesischen Gittern kein einziges Dreieck gefunden werden kann, da immer ein vierter Punkt kozirkular liegt und fälschlich als im Inneren liegend betrachtet werden könnte. Der Unterschied ist in Abbildung 18 illustriert.



(a) Kleine Deformation des Voronoi-Polygons durch zu großzügiges Umkreisprädikat



(b) Keine gültigen Dreiecke bei zu restriktivem Umkreisprädikat

Abbildung 18: Folgen eines unpräzisen Umkreisprädikates

Hat man mit diesem Verfahren die Delaunay-Triangulierung um den Kontrollpunkt herum ermittelt, kann man daraus das erste Voronoi-Polygon berechnen. Die Delaunay-Triangulierung kann nicht unbedingt entsprechend der Symmetriegruppe repliziert werden, da Symmetrien zu Überschneidungen von Dreieckskanten führen können, oder durch die Inexaktheit bei der Berechnung des Umkreis-Prädikates eventuell schon um den ersten Kontrollpunkt herum die Symmetrie verletzt ist. Die Form der Voronoi-Polygone hingegen ist gegen diese Probleme unempfindlich, von vernachlässigbaren numerischen Ungenauigkeiten abgesehen. Das Voronoi-Diagramm kann also als der Orbit des Randes des ersten Voronoi-Polygons gezeichnet werden.

Der Algorithmus benötigt unter Berücksichtigung von n Punkten eine Laufzeit von $O(n^2)$, da zwei ineinander geschachtelte Schleifen jeweils über die gesamte Punktemenge iterieren. Die besten bekannten Algorithmen für die Berechnung von allgemeinen Voronoi-Diagrammen benötigen $O(n \log n)$. Da n selten über 200 liegt, ist der asymptotische Laufzeitgewinn durch eine solche Implementierung eher unbedeutsam, verglichen mit den daraus resultierenden numerischen Problemen. Es ist anzunehmen, dass es für den Spezialfall der Berechnung nur eines Polygons eine geschicktere Implementierung mit geringerer asymptotischer Laufzeit gibt, die die Robustheit der einfachen Implementierung beibehält. Die einfache Implementierung ist jedoch für den vorliegenden Anwendungsfall ausreichend schnell.

7 Export der Zeichnungen

Das Programm bietet die Möglichkeit, gezeichnete Ornamente in verschiedene Raster- und Vektorgrafikformate zu exportieren. Dabei wurden exemplarisch verschiedene Ansätze implementiert.

7.1 Vektorformate

Bei den Vektorformaten werden nicht einfach entsprechend viele primitive Zeichenoperationen wiederholt in die Datei geschrieben, sondern die besonderen Funktionen der zugrunde liegenden Beschreibungssprache für sich wiederholende Muster verwendet.

In der Praxis führt dieser Ansatz leider mitunter zu Inkompatibilitäten, wenn diese sonst recht selten exzessiv genutzten Funktionalitäten entweder gar nicht oder fehlerhaft implementiert sind. Für den kommerziellen Einsatz wäre dieser Ansatz wohl der falsche. Zum Erkenntnisgewinn, zum Vergleich verschiedener Verfahren sowie zum Testen verschiedener Implementierungen hingegen ist die Methode von Vorteil. Außerdem werden durch den musterorientierten Ansatz die erzeugten Dateien kleiner, und die Dateigröße ist unabhängig von der vom Muster überdeckten Fläche.

7.1.1 Portable Document Format (PDF)

PDF[1] stellt Sprachelemente für regelmäßige Muster zur Verfügung. Dabei werden Kopien einer Translationszelle in einem rechtwinkligen Gitter angeordnet. Durch eine optionale Transformationsmatrix kann dieses Muster anschließend noch affin verzerrt werden. Diese Verzerrung wirkt sich jedoch auch auf die Stiftform aus, die beim Zeichnen der Linien gilt. Sie würde also in vielen Gruppen aus einem ursprünglich runden Stift einen elliptischen Stift machen und so die Symmetrie brechen. Eine mögliche Lösung wäre bei der Definition des Musters eine entsprechend inverse Verzerrung des Koordinatensystems zu definieren, so dass sich der Effekt dieser beiden Verzerrungen aufhebt und am Ende nur die Translationszelle, nicht aber der beim Zeichnen verwendete Stift, verzerrt ist.

Ein weiteres Problem entsteht dadurch, dass vom Benutzer gezeichnete Linienzüge sich im Allgemeinen nicht auf eine Translationszelle beschränken. Dadurch können sich die Inhalte mehrerer Kopien der Translationseinheit überlappen. Die PDF-Referenz schreibt jedoch:

> The order in which individual tiles (instances of the cell) are painted is unspecified and unpredictable; it is inadvisable for the figures on adjacent tiles to overlap.

Zur Lösung dieses Problems kommen zwei Ansätze in Frage:

Zerlegung in mehrere Muster. Man könnte bei jedem Farbwechsel ein neues Muster beginnen, und alle so definierten Muster am Ende übereinander zeichnen lassen. Bei aufeinander folgenden Linienzügen mit gleicher Farbe ist die Reihenfolge irrelevant.

Beschränkung auf eine Translationszelle. Dazu wird genau eine Translationszelle dargestellt, was die vom Benutzer eingegebenen Linienzüge gegebenenfalls in mehrere Teile zerlegt.

Die gewählte Implementierung verwendet den Ansatz mit einer einzigen Translationszelle. Allerdings werden durch den PDF-Export keine Linienzüge beschrieben, sondern Flächen. Dadurch entfällt auch das Problem der verzerrten Stiftform, und auf die zusätzliche Transformation kann verzichtet werden.

Zur Berechnung der Flächenbeschreibung wird noch in Java der Umriss jedes Linienzuges beim Zeichnen mit dem jeweils gewählten Stift ermittelt. Eine Translationszelle wird mit allen in Frage kommenden Kopien dieses Linienzuges unter den Symmetrieoperationen überdeckt, und die Vereinigung der entsprechenden Flächenbeschreibungen gebildet. Aufeinander folgende Linienzüge gleicher Farbe werden zu einer einzigen Fläche vereinigt. Das Endergebnis wird auf etwas mehr als eine Translationszelle beschnitten, um Rundungsprobleme zu vermeiden. Das so definierte Muster wird zum Füllen eines die Seite überdeckenden Rechtecks verwendet.

Zur Erzeugung der PDF-Datei wurden Bibliotheken aus meinem Projekt „Writebird“, einer Software für elektronische Tafeln, verwendet[6]. Diese erlauben die Generierung von PDF auf sehr niedriger Ebene und gestatten somit Zugriff auf exotische Features, die von typischen Bibliotheken zur PDF-Erzeugung oft nicht zur Verfügung gestellt werden.

Die erzeugten Dateien können nach bisherigen Erfahrungen recht problemlos von den meisten PDF-verarbeitenden Anwendungen genutzt werden. Bei Mustern mit vielen langen Linienzügen über etliche Translationszellen hinweg dauert die Berechnung der Flächen beim Export mitunter noch etwas lange. Dafür wird die Menge gespeicherter Daten bei ausgemalten Flächen erheblich geringer im Vergleich zur Speicherung entsprechender Linienzüge.

Bei der Darstellung mit Kantenglättung (Antialiasing) sind in den meisten Programmen (wie z.B. dem Adobe Reader 7) die Kanten der Translationszellen als schwach durchscheinende Linien zu erkennen, wenn diese nicht genau horizontal oder vertikal verlaufen. Die Ursache dürfte darin liegen, dass die Kantenglättung vor der Replizierung des Musters ausgeführt wird. Womöglich könnten überlappende Puffer, wie sie in Abschnitt 5.1 auf Seite 26 beschrieben wurden, auch in diesen Anwendungen derartige Probleme vermeiden helfen.

7.1.2 Scaleable Vector Graphics (SVG)

Die Generierung von SVG-Dateien[29] erfolgt im Wesentlichen nach dem gleichen Verfahren wie die von PDF-Dateien. Besonders interessant ist die Möglichkeit, mit frei verfügbaren Editoren das Muster aus der exportierten Datei zu kopieren und zum Füllen beliebiger anderer Objekte zu verwenden.

Allgemein ist die SVG-Unterstützung in vielen Produkten noch unausgereift, und einige unterstützen offenbar noch gar keine Muster. In manchen anderen (z.B. Inkscape 0.45.1) treten Probleme auf, wenn Musterzellen über eine Translationszelle hinausgehen, obwohl dies prinzipiell im Standard vorgesehen ist. Die Beschränkung auf eine einzige Translationszelle ist für SVG also von besonderer Bedeutung.

7.1.3 PostScript

PostScript[2] ist eine vollwertige Programmiersprache. In dieser ist es möglich, den in Abschnitt 4 auf Seite 21 beschriebenen Algorithmus zur Erzeugung aller benötigten Kopien jedes einzelnen Linienzuges zu implementieren. Die exportierte Datei besteht also im Wesentlichen aus einem fast unveränderlichen Kopfteil mit rund 50 Zeilen Programmcode. Dann kommen die den Vektoren a und b entsprechende Matrix und die Matrizen für die inneren Symmetrien der Gruppe. Danach

folgen die Beschreibungen der einzelnen Linienzüge in (a,b)-Koordinaten. Damit entspricht das Speicherformat weitgehend der programminternen Repräsentation des Musters.

Besonders praktisch ist bei PostScript die Möglichkeit, beim Zeichnen von Linien mit dem `ustroke`-Operator eine Matrix anzugeben, die auf den Stift wirkt, nicht jedoch auf den zu zeichnenden Pfad. Dadurch ist ein direktes Zeichnen in (a,b)-Koordinaten möglich, ohne dass die Stiftform dadurch verzerrt wird.

PostScript ist ein Textformat, was mitunter zu verhältnismäßig großen Dateien führt. Um die Dateigröße zu reduzieren, werden die Linienzüge als sogenannte Encoded User Paths[3] gespeichert, in Base85-Codierung. Manche Ausgabegeräte scheinen mit dieser Darstellung jedoch Probleme zu haben.

PostScript besitzt prinzipiell auch Mechanismen zum Zeichnen regelmäßiger Muster, ganz ähnlich zu denen in PDF. Es wurde bewusst eine andere Implementierung gewählt, um diese im Vergleich betrachten zu können. Der Export erfolgt sehr viel schneller als bei PDF, da die Berechnung der Flächenbeschreibungen entfällt. Die Darstellung auf einem Ausgabegerät benötigt hingegen etwas mehr Zeit als bei PDF. Artefakte durch Antialiasing treten mit den PostScript-Dateien nicht auf.

7.2 Rasterformate

Durch die Java-Klasse `javax.imageio.ImageIO` wird das Speichern von Rastergrafiken in gebräuchlichen Speicherformaten ermöglicht. Es wäre also ein Leichtes, einfach den Puffer, der wie in Abschnitt 5 auf Seite 26 beschrieben zum Füllen der Zeichenfläche verwendet wird, direkt in eine Datei zu speichern, oder alternativ eine mit Kopien dieses Puffers überdeckte größere Fläche.

Mit dieser Methode hätte der gespeicherte Ausschnitt der konzeptionell unendlich großen Zeichenfläche eine feste Größe. Zu wünschen wären Grafiken, die sich nahtlos aneinander fügen und somit Flächen beliebiger Größe mit dem Ornament füllen. Bei rechteckigen Translationszellen, deren Kanten exakt horizontal und vertikal verlaufen, ist dies mit dem Export einer einzigen Translationszelle gegeben. Bei anderen Formen der Translationszelle oder bei einer Drehung einer rechteckigen Zelle ist dies jedoch keine Lösung, da die meisten Anwendungen bei der Kachelungen mit Rastergrafiken ein regelmäßiges Rechteckgitter benutzen.

Man kann jedoch unter Verwendung mehrerer Translationszellen einen geeigneten rechteckigen Ausschnitt erhalten. Hinreichende Voraussetzung für eine nahtlose Fortsetzung des Musters ist, dass alle vier Eckpunkte des Rechtecks auf Gitterpunkten des von den Basisvektoren a und b aufgespannten Koordinatensystems liegen, dass diese Punkte also ganzzahlige (a,b)-Koordinaten haben. Für den Ursprung ist dies trivialer Weise der Fall, und wenn zwei weitere Ecken Gitterpunkte sind, so ist die letzte Ecke automatisch auch ein Gitterpunkt.

Da sich die Enden der Basisvektoren im Programm nur pixelweise verschieben lassen und daher immer ganzzahlige Komponenten haben, gibt es auch immer ein Rechteck, dass diese Bedingung erfüllt. Es hat die Breite

$$w = \left| \frac{b_y a_x - a_y b_x}{\mathrm{ggT}(a_y, b_y)} \right| \tag{7.1}$$

und die Höhe

$$h = \left| \frac{b_x a_y - a_x b_y}{\mathrm{ggT}(a_x, b_x)} \right| \tag{7.2}$$

Dieses Rechteck kann jedoch in vielen Fällen unhandlich groß sein, so dass die so entstehende Grafik immense Mengen an Speicherplatz erfordern würde.

Eine bessere Lösung ist es, Gitterpunkte zu suchen, die fast auf den Achsen des (x, y)-Koordinatensystems liegen, und dann das Gitter so zu modifizieren, dass auf jeder dieser Achsen ein solcher Punkt exakt zu liegen kommt. Bei der Auswahl eines geeigneten Punktes sind drei Kriterien gegeneinander abzuwägen.

Winkelfehler. Wird der Winkel zwischen den Gittervektoren geändert, verändert damit die Translationszelle ihre Form. Streng genommen würde das in den meisten Symmetriegruppen die Anforderungen an die Form der Translationszelle verletzen. Der Mensch nimmt Änderungen der Form verhältnismäßig stark wahr. Daher ist dieser Winkelfehler klein zu halten.

Rotationsfehler. Selbst wenn sich der Winkel wenig ändert, kann die Abweichung der Punkte von den jeweiligen Geraden eine Rotation des Ausschnitts als Ganzes bewirken. Bei allgemein gedrehter Translationszelle nimmt der Mensch leichte Abweichungen in deren Ausrichtung kaum wahr. Bei besonderen Ausrichtungen der Translationszelle, beispielsweise exakt 45° gegen die Hauptachsen gedreht, werden sich auch geeignete Gitterpunkte finden, die exakt auf den Achsen liegen und diesen Fehler komplett vermeiden.

Ausschnittsgröße. Wie oben beschrieben ist eine exakte Reproduktion des Musters in einem nahtlos kachelbaren Rechteck in jedem Fall möglich, wenn man nur den Ausschnitt groß genug wählt. Im Allgemeinen ist jedoch eine Verzehnfachung des Speicherbedarfs für eine um ein zehntel Grad exaktere Ausrichtung unverhältnismäßig. Deswegen muss die Größe des durch sie definierten Rechtecks in die Bewertung eines Paares von Gitterpunkten mit einfließen.

7.3 Unterschiedliche Export-Medien

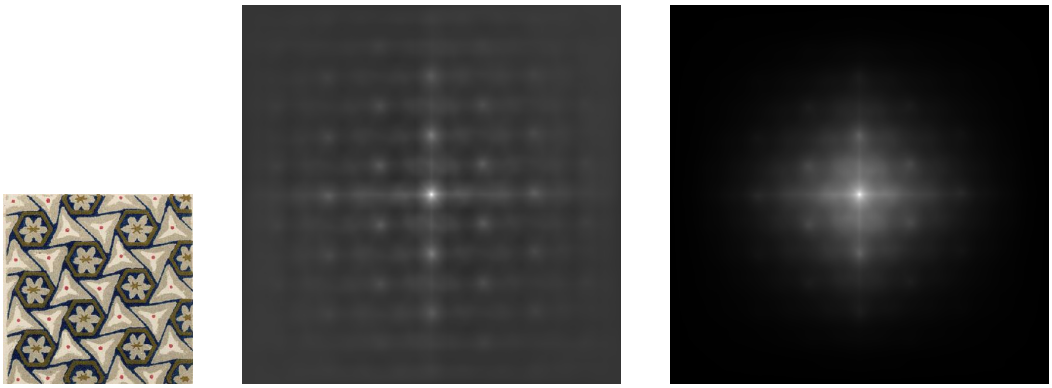
Wird das Programm als eigenständige Java-Anwendung oder als vertrauenswürdiges Applet ausgeführt, so kann der Export ganz normal in eine vom Benutzer ausgewählte Datei erfolgen. In anderen Einsatzszenarien ist diese Möglichkeit jedoch nicht gegeben. Aus diesem Grund wurde der Transport der generierten Daten in eine dem Benutzer verfügbare Datei in eine abstrakte Klasse `ExportMechanism` gekapselt. Verschiedene Implementierungen lassen sich leicht austauschen.

Wird die Anwendung über Java WebStart gestartet und nicht als vertrauenswürdig eingestuft, so ist ein direkter Zugriff auf das Dateisystem nicht möglich. Die JNLP-Umgebung stellt jedoch einen `FileSaveService` zur Verfügung, der dem Benutzer zur Angabe eines Dateinamens auffordert und die Daten dann dort ablegt. Durch diese Benutzerinteraktion wird der unkontrollierte Zugriff böswilliger Programme auf beliebige Dateien verhindert.[9]

Eine besondere Lösung erfordert die Integration des Programms in das Mathematik-Museum ix-quadrat an der TU München. Dort wird die Anwendung als Applet betrieben. Die Benutzer haben weder auf das Dateisystem dieser Computer direkten Zugriff, noch steht ihnen eine Tastatur zur Verfügung. Als Lösung wird der Benutzer per Bildschirmtastatur zur Eingabe einer E-Mail-Adresse aufgefordert. An diese wird die exportierte Datei dann als Anhang einer E-Mail geschickt.

8 Automatische Mustererkennung

Eine der jüngsten Erweiterungen des Programms ist die Möglichkeit, automatisch Bilder symmetrischer Muster zu analysieren, die zugrunde liegende Symmetriegruppe zu erkennen und das Bild als Hintergrundbild für eigene Ornamente zur Verfügung zu stellen. Die Implementierung



(a) Eingabe (b) Symmetrische Werte (c) Asymmetrische Werte

Abbildung 19: Das Eingabebild und seine Autokorrelation

folgt im Wesentlichen der Arbeit von Liu, Collins und Tsin[16]. In einigen Bereichen wurden jedoch andere Strategien verwendet, die in vielen Situationen zu besseren Ergebnissen führen.

8.1 Autokorrelation

Der erste große Schritt auf dem Weg zur automatischen Mustererkennung ist die Identifizierung von Translationssymmetrien. Dazu muss das Bild mit verschobenen Versionen seiner selbst in Beziehung gesetzt und der Grad der Übereinstimmung als Zahl ausgedrückt werden.

Ein geeignetes Maß ist das Skalarprodukt der Bilder, die sogenannte *Autokorrelation*. Die Pixelwerte des ursprünglichen Bildes werden mit denen des verschobenen Bildes multipliziert und die Produkte über den gesamten überlappenden Bereich aufsummiert. Bezeichnen wir den Helligkeitswert eines Bildes der Breite M und Höhe N an der Stelle (x, y) als

$$p_{x,y} \in \mathbb{R} \qquad 0 \leq x < M, 0 \leq y < N \tag{8.1}$$

Vereinbaren wir weiterhin, dass $p_{x,y}$ für Koordinaten außerhalb der Bildabmessungen den Wert 0 haben soll, dann können wir den Autokorrelationswert für eine Verschiebung um (x, y) so formulieren:

$$a_{x,y} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} p_{m,n} \cdot p_{m+x,n+y} \tag{8.2}$$

Der Wertebereich, in dem die Bildinformation dargestellt wird, ist für die Deutlichkeit der Ergebnisse relevant. Liegt dieser Bereich symmetrisch um 0 verteilt, also z.B. von -1 bis $+1$, so erscheinen die Ergebnisse für den Menschen klarer. Der Computer hingegen erzielt aus noch nicht genau geklärten Gründen zumindest mit der im Folgenden besprochenen Implementierung mit einer asymmetrischen Verteilung von 0 bis 1 bessere Ergebnisse. Die Ergebnisse beider Berechnungsweisen sind in Abbildung 19 dargestellt. Der Nullpunkt liegt dabei jeweils in der Mitte der Abbildung, ersichtlich durch das globale Maximum der Autokorrelation an dieser Stelle.

Der Wert der Autokorrelation ist immer dann lokal maximal, wenn Stellen hoher Intensität in originalem und im verschobenem Bild zusammenfallen. Daher ist sie ein gutes Maß für die Selbstähnlichkeit des Musters. Ansätze über die Differenz der Intensitäten oder dergleichen wären zwar etwas anschaulicher, jedoch deutlich schwerer effizient zu berechnen. Auch bei der Autokorrelation wie oben definiert bräuchte ein naiver Ansatz $O(M^2N^2)$ Schritte zur Berechnung aller

Werte. Eine effizientere Implementierung verwendet die schnelle Fourier-Transformation (FFT), eine Implementationsstrategie zur Berechnung der diskreten Fourier-Transformation (DFT). Dadurch benötigt sie nur noch $O(MN\log(MN))$ Schritte.[15]

Die eindimensionale Fourier-Transformierte eines Vektors $f = (f_0, f_1, \ldots, f_{N-1}) \in \mathbb{C}^N$ von N komplexen Zahlen ist wie folgt definiert:

$$F_k = \sum_{j=0}^{N-1} e^{2\pi i \frac{jk}{N}} f_j \tag{8.3}$$

Die dazu passende inverse Transformation (iDFT) ist

$$f_k = \frac{1}{N} \sum_{j=0}^{N-1} e^{-2\pi i \frac{jk}{N}} F_j \tag{8.4}$$

Es gibt in der Literatur einige leicht abweichende Definitionen. Das Vorzeichen in der Exponentialfunktion ist manchmal vertauscht, oder der Faktor $\frac{1}{N}$ als $\frac{1}{\sqrt{N}}$ gleichmäßig auf die beiden Transformationsrichtungen verteilt. Diese feinen Unterschiede sind für die folgende Anwendung irrelevant.

Für die Anwendung auf die Bilddaten wird eine zweidimensionale Fourier-Transformation verwendet. Diese lässt sich berechnen, indem zunächst alle Zeilen und anschließend alle Spalten transformiert werden. Die Berechnung in umgekehrter Reihenfolge führt zum gleichen Ergebnis.

$$P_{x,y} = \sum_{m=0}^{M-1} e^{2\pi i \frac{mx}{M}} \sum_{n=0}^{N-1} e^{2\pi i \frac{ny}{N}} p_{m,n} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} e^{2\pi i \left(\frac{mx}{M} + \frac{ny}{N}\right)} p_{m,n} \tag{8.5}$$

Zur Berechnung der Autokorrelation werden die so erhaltenen Fourier-Koeffizienten $P_{x,y}$ mit ihrem komplex Konjugierten multipliziert.

$$A_{x,y} = \overline{P_{x,y}} \cdot P_{x,y} \tag{8.6}$$

Das Ergebnis dieser komponentenweisen Multiplikation wird anschließend einer zweidimensionalen iDFT unterworfen, um die gesuchten Autokorrelationswerte zu erhalten.

$$a_{x,y} = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} e^{-2\pi i \left(\frac{mx}{M} + \frac{ny}{N}\right)} A_{m,n} \tag{8.7}$$

Lemma 15: Der in Gleichungen 8.5 bis 8.7 berechnete Autokorrelationswert erfüllt die folgende zyklische Variante der in Gleichung 8.2 angegebenen Formel:

$$a_{x,y} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} p_{m,n} \cdot p_{(m+x) \bmod M, (n+y) \bmod N} \tag{8.8}$$

Beweis: Der folgende Beweis beschränkt sich zur Vereinfachung auf den eindimensionalen Fall. Der zweidimensionale ist analog zu behandeln, nur mit doppelt so vielen Summenzeichen und Laufvariablen.

$$\omega := e^{\frac{2\pi i}{N}}$$

$$P_x = \sum_{k=0}^{N-1} \omega^{xk} p_k$$

$$
\begin{aligned}
A_x &= \overline{P_x} \cdot P_x \\
&= \overline{\left(\sum_{m=0}^{N-1} \omega^{xm} p_m\right)} \left(\sum_{n=0}^{N-1} \omega^{xn} p_n\right) \\
&= \left(\sum_{m=0}^{N-1} \omega^{-xm} p_m\right) \left(\sum_{n=0}^{N-1} \omega^{xn} p_n\right) \\
&= \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \omega^{x(n-m)} p_m p_n
\end{aligned}
$$

$$
\begin{aligned}
a_x &= \frac{1}{N} \sum_{k=0}^{N-1} \omega^{-xk} A_k \\
&= \frac{1}{N} \sum_{k=0}^{N-1} \omega^{-xk} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \omega^{k(n-m)} p_m p_n \\
&= \frac{1}{N} \sum_{k=0}^{N-1} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \omega^{k(n-m-x)} p_m p_n \\
&= \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} p_m p_n \sum_{k=0}^{N-1} \left(\omega^{n-m-x}\right)^k \\
&= \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} p_m p_n \begin{cases} N & \text{falls } \omega^{n-m-x} = 1 \\ 0 & \text{sonst} \end{cases} \\
&= \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} p_m p_n \begin{cases} N & \text{falls } n \equiv m + x \pmod{N} \\ 0 & \text{sonst} \end{cases} \\
&= \sum_{m=0}^{N-1} p_m p_{m+x \bmod N}
\end{aligned}
$$

Die Fallunterscheidungen ergeben sich daraus, dass alle Potenzen von 1 immer 1 sind. Andernfalls ist die Formel zur Berechnung geometrischer Reihen anzuwenden, was bei einer solchen Reihe von Einheitswurzeln immer zu 0 führt:

$$
\sum_{k=0}^{N-1} \omega^{jk} = \frac{\left(\omega^j\right)^N - 1}{\omega^j - 1} = \frac{e^{2\pi i j} - 1}{\omega^j - 1} = \frac{1 - 1}{\omega^j - 1} = 0 \quad \text{für } \omega^j \neq 1,\, j \in \mathbb{Z} \qquad \square
$$

Wählt man die Dimensionen M und N für die Fourier-Transformationen mindestens doppelt so groß wie das eigentliche Bild, und initialisiert alle Pixel außerhalb des eigentlichen Bildes mit 0, so kann man die Autokorrelation aus Gleichung 8.8 statt der aus Gleichung 8.2 verwenden. Durch die Modulo-Arithmetik sind auch Verschiebungen in negative Richtungen aus dem Ergebnis abzulesen, indem man einfach entsprechend viele Vielfache von M bzw. N addiert.

Die schnelle Fourier-Transformation lässt sich am einfachsten implementieren, wenn die Zahl der zu bearbeitenden Elemente eine Zweierpotenz ist. Aus diesem Grund werden beide Bilddimensionen zunächst verdoppelt und dann zur nächsten Zweierpotenz aufgerundet.

Der Speicherbedarf für mittelgroße Bilder kann erheblich sein. Da viele Java-Installationen standardmäßig eine Beschränkung auf 64 MiB Arbeitsspeicher konfiguriert haben, und diese für

den Benutzer mitunter nur schwer zu ändern ist, werden die Daten für die Autokorrelationsanalyse in einer temporären Datei gehalten. Durch eine direkte Abbildung von Dateien in den Arbeitsspeicher (Memory Mapping) sowie moderne Caching-Technologien ist die Effizienz dieser Methode mit der eines ausschließlich im Arbeitsspeicher abgelegten Puffers durchaus vergleichbar. Eine Konsequenz dieses Ansatzes ist, dass die Mustererkennung nur bei Zugriffsrechten auf das lokale Dateisystem zur Verfügung steht.

Die naive Speicherung von Bilddaten, nämlich einfach eine Zeile nach der anderen jeweils komplett hintereinander abzulegen, führt zu großen Performanceproblemen, wenn die Daten Spaltenweise traversiert werden sollen. Dies liegt in den Caching-Verfahren begründet. Bei entsprechender Größe des Bildes liegen auf einer Speicherseite nur wenige Pixel dieser Spalte. Es müssen daher mehr Speicherseiten geladen werden, als der Cache fassen kann. Dies führt dazu, dass jede Spalte erneut ihre Daten aus dem Arbeitsspeicher beschaffen muss, während der Cache mit ungenützten Daten überfüllt ist. Die verbreitete Lösungsstrategie für dieses Problem liegt darin, das Bild in Kacheln aufzuteilen, die in etwa die Größe einer Speicherseite haben. Eine Spalte überdeckt dann sehr viel weniger Kacheln, als sie Zeilen überdeckt, weshalb die Zahl der benötigten Speicherseiten kleiner ist, und im Allgemeinen alle diese Seiten im Cache gehalten werden können.

Bisher haben wir die Bilddaten als reelle Zahlen angenommen. In der Realität haben die meisten Bilder jedoch drei bis vier Farbkanäle. Diese werden jeweils einzeln transformiert und die Ergebnisse anschließend addiert. Anders als bei einer Konvertierung in Graustufen vor Beginn der Analyse[18] wird durch diese Herangehensweise auch bei verschiedenen Farben mit ähnlichen Helligkeitswerten noch eine gute Erkennung der Symmetrieeigenschaften gewährleistet.

Die Amplitude der Korrelation nimmt mit zunehmender Länge der Verschiebungsvektoren ab, da sich die beiden Instanzen des Bildes immer weniger überdecken, und somit auch weniger Übereinstimmung bestehen kann. Man kann diesem Effekt entgegenwirken, indem man jeden Wert durch die ihm entsprechende Überlappungsfläche teilt.

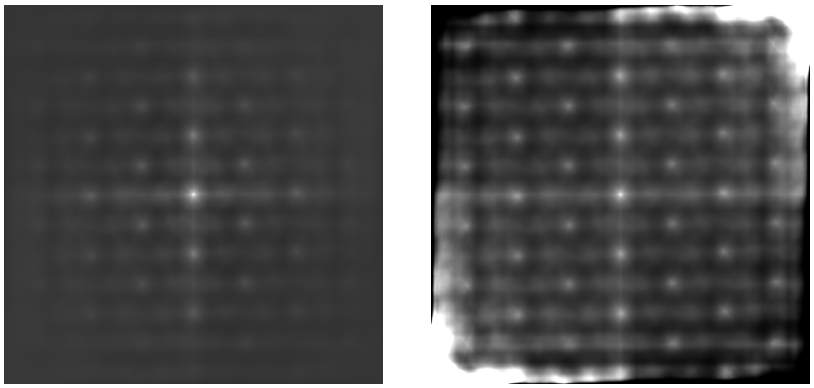
$$a'(x,y) = \frac{a(x,y)}{(M-|x|)(N-|y|)} \tag{8.9}$$

Der Effekt der Normalisierung ist in Abbildung 20 auf der nächsten Seite dargestellt. Versuche mit verschiedenen Bildern haben gezeigt, dass mit dieser Normalisierung die weiter unten beschriebenen Schritte zwar mitunter bessere Zwischenergebnisse, wie beispielsweise vollständigere Gitter, liefern. Die Endergebnisse blieben jedoch hinter denen ohne Normalisierung zurück, weshalb der Programmcode für die Normalisierung deaktiviert wurde.

8.2 Peaks

Die Stellen maximaler Selbstähnlichkeit liegen jetzt als lokale Maxima der Autokorrelationsfunktion vor. Allerdings fallen die Funktionswerte zum Rand hin stark ab, da immer weniger Überlappungsfläche zwischen originalem und verschobenem Bild besteht. Allein aus dem Wert an einer gewissen Stelle lässt sich also schwer entscheiden, ob es sich dabei um einen relevanten Peak handelt.

Speziell für diese Anwendung empfiehlt sich ein Ansatz, der im Englischen mit „Area of Dominance“ bezeichnet wird, zu Deutsch „Dominanzgebiet“. Im Folgenden wird für diese Größe das Symbol d verwendet. Der Grundgedanke dabei ist, dass nicht so sehr die tatsächlichen Werte der Peaks interessieren. Bedeutender ist statt dessen die Fläche, die ein Peak dominiert, in der also kein weiterer Peak mit höherem Wert liegt. Diese Fläche wird kreisförmig gemessen und folglich durch den nächstgelegenen Punkt mit höherem Autokorrelationswert bestimmt. Peaks mit einem großen Dominanzgebiet sind in einem recht großen Raum maximal, während Peaks



(a) ohne Normalisierung (b) mit Normalisierung

Abbildung 20: Effekt der Normalisierung (bei symmetrischem Wertebereich)

mit kleinerer Dominanzgebiet recht nah einen stärkeren Nachbarn haben und im Muster daher eine untergeordnete Rolle spielen.[16]

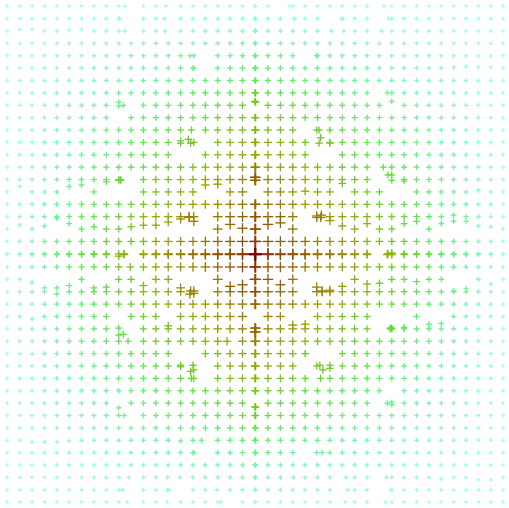
$$d(x,y) = \pi \cdot \min\left\{(x-x')^2 + (y-y')^2 \,|\, a(x',y') > a(x,y)\right\} \tag{8.10}$$

Zunächst werden Peakkandidaten ermittelt. Dazu wird das Bild in kleine Quadrate unterteilt, und in jedem Quadrat der maximale Wert gesucht und als Peakkandidat gespeichert. Die Größe der Quadrate richtet sich nach der Bildgröße, so dass die Zahl der Kandidaten und damit der Rechenaufwand für die folgenden Schritte in etwa unabhängig von der Größe des Bildes ist. Die Idee dahinter ist, dass größere Bilder auch tendenziell größere Strukturen enthalten und daher die gesuchten Peaks weiter auseinander liegen werden. Daher ist die Gefahr, durch zu große Quadrate einen Peak zu übersehen, eher gering. Die in der Literatur angegebene Variante, ein Gleitfenster (Sliding Window) zur Ermittlung der Peakkandidaten zu verwenden[16], erschien den damit verbundenen Programmier- und Rechenaufwand nicht wert, da die festen Quadrate als Fenster hinreichend gute Ergebnisse liefern.

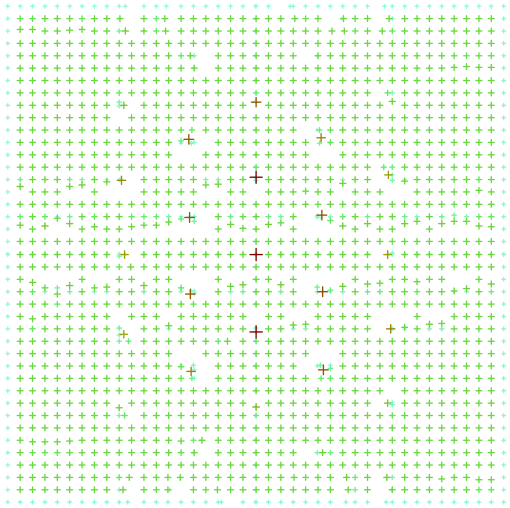
Die Peakkandidaten werden entsprechend ihrem Autokorrelationswert sortiert. Beginnend mit dem zentralen Peak, der zwangsläufig maximalen Autokorrelationswert hat, wird ein *Quadtree* aufgebaut. Darin hat jeder Peakkandidat bis zu vier andere Kandidaten mit geringerem Wert als Kinder, je einen pro Quadrant um die Position dieses Peakkandidaten. Durch diese Datenstruktur kann recht effizient beim Einfügen eines Kandidaten in den Baum der nächstgelegene Kandidat mit höherem Wert ermittelt und zur Berechnung des Dominanzgebietes eingesetzt werden.

Anschließend werden aus den Peakkandidaten diejenigen Peaks ermittelt, die für die folgenden Berechnungen weiter in Betracht gezogen werden sollen. Dazu werden alle Kandidaten erneut sortiert, diesmal nach ihrem Dominanzgebiet. Das Dominanzgebiet des neunten Kandidaten wird mit einem festen Faktor, der zwischen 0 und 1 liegt, multipliziert, und der so erhaltene Grenzwert zum Kürzen der Liste verwendet: nur Kandidaten, deren Dominanzgebiet über diesem Wert liegt, werden weiter betrachtet. Dadurch soll sichergestellt werden, dass der zentrale Peak und acht benachbarte Peaks (in einem rechteckigen Gitter) auf jeden Fall weiter betrachtet werden, und außerdem alle anderen Peaks, die „ähnlich gut“ wie der schwächste dieser neun Peaks sind.

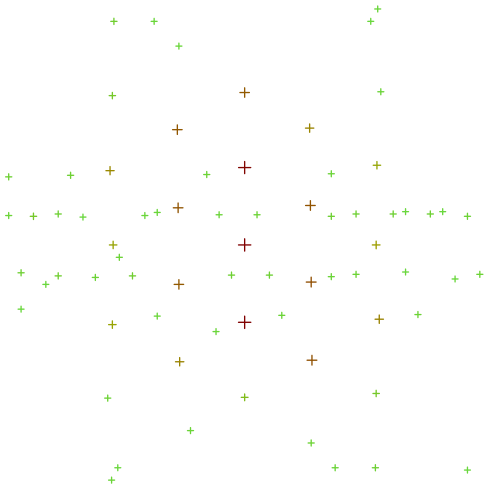
In den Teilbildern a bis c der Abbildung 21 auf der nächsten Seite werden die Schritte der Peakberechnung illustriert, ausgehend von der Autokorrelation aus Abbildung 19c auf Seite 47. Jeder Peakkandidat wird dabei als Kreuz dargestellt, dessen Farbe und Größe sich in Teilbild



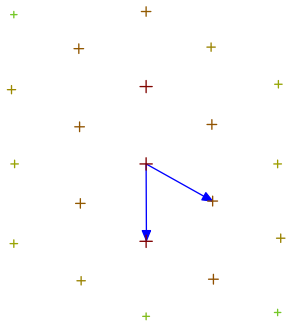
(a) Peakkandidaten nach Wert



(b) Peakkandidaten nach Dominanz



(c) Ausgedünnte Kandidatenliste



(d) Gefundenes Gitter

(e) Skala

Abbildung 21: Die Schritte zur Ermittlung von Peaks

a nach der Stärke des Autokorrelationswertes, in den anderen Teilbildern nach der Größe ihres Dominanzgebietes richtet. In Bereichen, in denen die Autokorrelation einen nahezu gleichmäßigen Gradienten aufweist, liegen alle Peakkandidaten in der gleichen Ecke ihres jeweiligen Fensters, was die regelmäßigen Gitterstrukturen erklärt, die in weiten Bereichen der Teilbilder a und b zu erkennen ist. In der Nähe echter Peaks wird diese Gleichmäßigkeit jedoch gestört, so dass die Position eines Peaks auf den Pixel genau verfügbar ist, und nicht nur das Fenster, in dem der Peak liegt.

8.3 Gittervektoren

Aus den verbleibenden Peaks müssen jetzt zwei geeignete Basisvektoren ermittelt werden, die das Gitter der in der Symmetriegruppe enthaltenen Translationen bilden. Dazu muss jedes Paar potentieller Gittervektoren, die sich aus den Peaks ergeben, bewertet und anschließend das geeignetste Paar gewählt werden. Dabei werden vor allem zwei Ziele verfolgt:

Stimmigkeit. Die vorhandenen Peaks sollen möglichst auf Gitterpunkten liegen.

Vollständigkeit. Auf jedem Gitterpunkt soll nach Möglichkeit ein Peak liegen.

In der Literatur wird häufig hauptsächlich der erste Punkt, die Stimmigkeit des Gitters, betrachtet[15]. Das führt zu einer Bevorzugung von zu kleinen Gittervektoren. Wenn ein regelmäßiges Muster eine teilweise regelmäßige Unterstruktur aufweist, kommt es leicht vor, dass einzelne zu dieser Unterstruktur passende Peaks erhalten geblieben sind und als Gittervektoren in Betracht gezogen werden. Sind die eigentlichen Gittervektoren nun annähernd ganzzahlige Vielfache dieser falschen Peaks, so liegen alle echten Peaks hinreichend nah an Gitterpunkten, um für diese Gittervektoren zu sprechen. Dem menschlichen Betrachter fällt hingegen sofort auf, dass das so ermittelte Gitter voller Löcher und daher unpassend ist.

Statt der Konzentration auf die Stimmigkeit sollten beide Aspekte vergleichbar behandelt werden. Für den im Folgenden vorgestellten Ansatz sind zwei Funktionen wichtig:

Die Bewertung erfasst, wie gut Peak und Gitterpunkt zusammen passen.

Die Gewichtung drückt aus, wie wichtig ein Peak bzw. ein Gitterpunkt für die Gesamtbewertung ist.

Die Bewertung wird für Vollständigkeit und Stimmigkeit auf die gleiche Weise ermittelt. Eingaben in die Bewertungsfunktion sind ein Peak sowie der dazu nächstgelegene Gittervektor. Bei exakter Übereinstimmung soll die Bewertungsfunktion einen maximalen positiven Wert annehmen. Bei zu starker Abweichung kann die Bewertungsfunktion auch negative Werte annehmen. Dies drückt die Tatsache aus, dass fehlende Peaks weniger stark gegen ein mögliches Gitter sprechen als starke Peaks an komplett falschen Stellen. Derzeit ist die Bewertungsfunktion f für einen Peak p mit der Zerlegung in Gittervektoren $p_a a + p_b b$ wie folgt implementiert:

$$f(p) = 10\left(2\left(\left|p_a - \left\lfloor p_a + \frac{1}{2}\right\rfloor\right| - \frac{1}{2}\right)^2 + 2\left(\left|p_b - \left\lfloor p_b + \frac{1}{2}\right\rfloor\right| - \frac{1}{2}\right)^2\right) - 9 \tag{8.11}$$

Die äußere Klammer ermittelt einen entfernungsabhängigen Wert zwischen 0 und 1, während die Teile außerhalb der Klammer die Grenze zwischen negativen und positiven Bewertungen festlegen. Das Ergebnis ist ein Berg mit recht steilem Gipfel und weich auslaufenden Flanken, wie in Abbildung 22 auf der nächsten Seite veranschaulicht. Der positiv bewertete Bereich um einen Gitterpunkt herum ist verhältnismäßig klein.

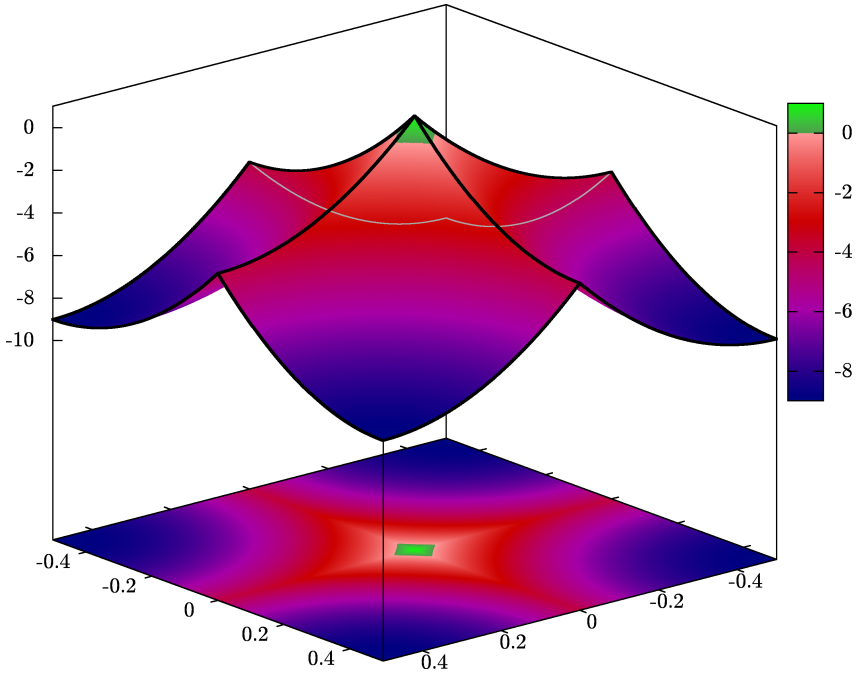


Abbildung 22: Die in Gleichung 8.11 beschriebene Bewertungsfunktion

Diese Formel ist eine Heuristik, die sich empirisch entwickelt hat. Sie funktioniert hinreichend gut, ist aber mit Sicherheit noch verbesserungsfähig. Ein Systematischer Vergleich verschiedener Heuristiken steht noch aus.

Für die Berechnung der Stimmigkeit wird das Dominanzgebiet d als Gewichtungsfunktion verwendet. Je dominanter ein Peak ist, desto wichtiger ist es, diesen Peak möglichst gut zu treffen. Die Berechnung der Vollständigkeit muss die Bedeutung einer Gitterposition ausdrücken. Ein geeignetes Maß hierfür ist die Zahl der überlappenden Pixel bei der dieser Gitterposition entsprechenden Verschiebung. Wenn weit außen bei geringem Überlapp ein Peak verloren geht, so ist dies weniger gravierend, als wenn mitten im Zentrum des Gitters Löcher bestehen. Konkret ist der Überlapp o an der Gitterposition (i, j) eines von den Vektoren a und b aufgespannten Gitters

$$o(i, j) = \max\{0, M - |ia_x + jb_x|\} \cdot \max\{0, N - |ia_y + jb_y|\} \tag{8.12}$$

Die Gewichtungsfunktionen werden unabhängig voneinander mit den Bewertungen der jeweils zugeordneten Peaks multipliziert. Diese Produkte werden jeweils für alle Peaks aufsummiert und anschließend durch eine Division normiert. Der Divisor dieser Normierung ist dabei jeweils verwandt mit der Gewichtungsfunktion: Für die Stimmigkeit der Peaks wird durch die Summe der Dominanzgebiete aller Peaks geteilt. Für die Vollständigkeit des Gitters wird die Summe der Überlappungsflächen aller Gitterpunkte der Ebene als Divisor verwendet. Diese Normalisierung macht die Werte für verschieden lange Gittervektoren vergleichbar.

$$s_{stimmig} = \frac{\sum_{p \in P} d(p) f(p)}{\sum_{p \in P} d(p)} \tag{8.13}$$

$$s_{vollständig} = \frac{\sum_{p \in P'} o(\lfloor p_a + \frac{1}{2} \rfloor, \lfloor p_b + \frac{1}{2} \rfloor) f(p)}{\sum_{i \in \mathbb{Z}} \sum_{j \in \mathbb{Z}} o(i,j)} \tag{8.14}$$

Die in dieser Formel erwähnte Menge P' ist die Menge aller Peaks, die Gitterpunkten zugeordnet sind. Für die Vollständigkeit wird jedem möglichen Gitterpunkt höchstens ein Peak zugeordnet, bei mehreren Kandidaten derjenige mit besserer Bewertung, also größerem f. Daher ist $P' \subseteq P$. Nach der Normierung werden beide Werte multipliziert, um eine einzige Bewertungszahl für das Paar von möglichen Gittervektoren zu erhalten.

$$s = s_{stimmig} \cdot s_{vollständig} \tag{8.15}$$

Diese Multiplikation ist ebenfalls eine etwas willkürlich gewählte Heuristik, und andere Arten der Kombination sind denkbar. Ein systematischer Vergleich verschiedener Heuristiken steht auch hier noch aus. Das Paar mit der besten Bewertung wird schließlich gewählt. Paare, bei denen einer der beiden Werte negativ ist, kommen nicht in Betracht.

Es stellt sich nun noch die Frage, wie viele Peaks bei der Ermittlung geeigneter Gittervektoren betrachtet werden sollten. Im letzten Schritt der Peakermittlung wurde die Zahl der Kandidaten etwas willkürlich reduziert. Diese Reduzierung dient jedoch hauptsächlich dazu, nicht zu viel Zeit mit absolut irrelevanten Kandidaten zu verbringen, und sollte eher zu viele als zu wenige Peaks zulassen. Statt die Zahl der Peaks ein Mal festzulegen, werden die nach der Ausdünnung verbliebenen Peaks der Reihe nach zum Gitter hinzugenommen. Für jeden Peak werden die Bewertungssummen aktualisiert und so eine Beurteilung für die Menge aller Peaks bis zum zuletzt hinzugenommenen errechnet. Die jeweils beste Beurteilung wird gespeichert. Für jedes Paar möglicher Basisvektoren wird erneut über alle Peaks iteriert, und am Ende das Paar mit der insgesamt besten Bewertung verwendet, und die Menge von Peaks, die diese Bewertung erzielt hat. Das Ergebnis einer solchen Wahl von Basisvektoren und passenden Peaks ist in Abbildung 21d auf Seite 52 dargestellt. Durch diesen iterativen Ansatz ist die Wahl der optimalen Anzahl von Peaks nicht vorab zu treffen, sondern ergibt sich bei der Bewertung der Gittervektoren implizit. In Gleichung 8.13 wächst auch der Normalisierungsfaktor, wenn die Zahl der betrachteten Peaks steigt. Die Summe der Überlappungsflächen im Nenner in Gleichung 8.14 hingegen bleibt konstant, da diese unabhängig davon ist, ob eine Gitterstelle besetzt ist oder nicht.

Die bisher gefundenen Gittervektoren erfüllen die Anforderungen von Stimmigkeit und Vollständigkeit recht gut. Die meisten Peaks liegen auf Gitterpunkten, und die meisten wesentlichen Gitterpunkte sind auch besetzt.

Eine weitere wichtige Forderung ist die nach der richtigen Größenordnung des gefundenen Gitters. Eventuelle Obergruppen sollen signifikant schwächer ausgeprägt sein, eventuelle Untergruppen nicht signifikant stärker. Nimmt der Algorithmus beispielsweise das Doppelte der korrekten Gittervektoren, also das Gitter einer Untergruppe, so erhält man im Allgemeinen ein Gitter, das bei wenigen betrachteten Peaks unvollständig und bei vielen betrachteten Peaks unstimmig ist. Kritischer ist die Situation, wenn der Algorithmus das Gitter einer Obergruppe untersucht, also einer größeren Menge von Abbildungen, die das Muster signifikant stark mit sich selbst zur Deckung bringen, wenn auch nicht so gut wie die eigentlich gesuchte Symmetriegruppe. In diesem Fall ist das so gefundene Gitter auch stimmig (entsprechend der Definition auf Seite 53), und in schwierigen Situationen auch hinreichend vollständig, um dem eigentlichen Gitter Konkurrenz zu machen. Daher ist es schwierig, das Gitter einer Obergruppe von dem eigentlich gesuchten Gitter zu unterscheiden.

Es gibt viele Stellen, an denen dieses Verhalten beeinflusst werden kann. Eine davon ist die Reduzierung der betrachteten Peaks vor Beginn der Suche nach Gittervektoren. Dort wären insbesondere Stufen zu erkennen, also lange Reihen von Peaks mit ähnlichem Dominanzgebiet, gefolgt von Peaks mit nur noch etwa dem halben Dominanzgebiet oder weniger. Ein zweiter Faktor ist die Wahl der in Frage kommenden Basisvektoren für das Gitter, und ein dritter die Reihenfolge, in der die Peaks zur Erweiterung des Gitters und Aktualisierung der Bewertung betrachtet werden. Diese beiden Faktoren basieren jeweils auf dem Vergleich von Peaks, und die Auswahl an Vergleichskriterien ist recht groß. Man kann das Dominanzgebiet des Peaks verwenden, den Autokorrelationswert entweder normalisiert (nach Gleichung 8.9 auf Seite 50) oder nicht normalisiert, die Länge des Vektors, oder beliebige berechenbare Kombinationen dieser Vergleichskriterien. Auch eine Normalisierung bei der Berechnung der Dominanzgebiete sowie die Wahl der Bewertungsfunktion f beeinflussen das Verhalten an dieser Stelle.

Die Wahl geeigneter Heuristiken sprengt den Rahmen dieser Arbeit. Daher ist derzeit keine Stufenerkennung implementiert, und sowohl die Auswahl der Vektorkandidaten als auch die Reihenfolge der Gittererweiterung orientiert sich an dem nicht normalisiert berechneten Dominanzgebiet der Peaks. Die ersten neun Peaks werden als Vektorkandidaten in Betracht gezogen.

Noch eine weitere Anforderung besteht darin, möglichst kurze Gittervektoren zu finden, die das gleiche Gitter aufspannen. Dazu können in Abschnitt 6.3.1 auf Seite 33 zur Ermittlung einer Voronoi-Triangulierung vorgestellte Mechanismen verwendet werden. Die in Gleichungen 6.3 und 6.13 definierten Vektoren c und d sind zwei besonders kurze Vektoren, die das gleiche Gitter aufspannen. Bei einer Bewertungsfunktion wie der in Gleichung 8.11 auf Seite 53 angegebenen, die sich an den Gitterkoordinaten und nicht an den kartesischen Koordinaten der Punkte orientiert, ist es wünschenswert, schon bei der Bewertung des Kandidatenpaares auf entsprechende kurze Vektoren zu wechseln.

Abschließend wird die Passgenauigkeit des Gitters optimiert. Dazu werden mit der Methode der kleinsten Fehlerquadrate genauere Gittervektoren ermittelt, die den Unterschied zwischen Peaks und den ihnen zugeordneten Gittervektoren reduzieren. Diese Vektoren haben im Allgemeinen keine ganzzahligen Komponenten mehr. Dadurch sind sie in der Lage, selbst bei extrem genauen Vorlagen noch Verbesserungen gegenüber den ganzzahligen Peakvektoren zu erreichen.

8.4 Drehzentren

Die jetzt vorliegenden Translationsvektoren a und b können verwendet werden, um eine Musterkachel aus dem Bild zu extrahieren. Diese füllt das von den Gittervektoren aufgespannte Parallelogramm aus und repräsentiert jeden Orbit der Vorlage mit einem geeigneten Farbwert. Ein geeignetes Verfahren ist an dieser Stelle die Verwendung des Median aller Punkte des Orbits. Im Unterschied zum arithmetischen Mittel ist der Median resistent gegenüber einzelnen Ausreißern. Ein Loch im Muster, beispielsweise von einer gelochten Vorlage, verschwindet so vollkommen, statt sich schwach durchzudrücken. Ein Nebeneffekt dieses Verfahrens ist, dass unpräzise positionierte Musterelemente auch verschwinden, statt stark verschwommen noch sichtbar zu sein. Dieser Effekt ist in Abbildung 27 auf Seite 62 deutlich zu erkennen. Die Mediankachel dient als Datenbasis für die folgenden Schritte.

Nach der Extraktion dieser Kachel werden Rotationszentren bestimmt. Da nur Rotationen mit Periode zwei, drei, vier oder sechs in den zu erkennenden Gruppen vorkommen können, genügt es, sich auf diese Rotationen zu beschränken. Zum Auffinden der Rotationszentren wird die Mediankachel durch eine affine Abbildung auf ein Quadrat abgebildet, dessen Kanten den Basisvektoren a und b entsprechen. Ein Mal wird die Kachel direkt abgebildet, und ein Mal wird sie um den gesuchten Winkel gedreht und gegebenenfalls wieder in die Standard-Translationszelle abgebildet, bevor die Abbildung in das Quadrat stattfindet. Die affine Abbildung kann zwar die

(a) Ausgangslage (b) Um 60° rotiert (c) Korrelation

Abbildung 23: Korrelation zur Erkennung einer sechszähligen Drehung

Drehsymmetrie augenscheinlich zerstören, aber im Prinzip haben wir es hier nur wieder mit einem schrägen Koordinatensystem zu tun.

Diese beiden Quadrate werden jetzt mittels FFT miteinander korreliert. Diesmal werden die Daten für die Transformation nicht von Nullen umgeben, sondern es wird ganz bewusst die zyklische Natur der Fourier-Transformation ausgenutzt, um die Korrelation gleichsam mit zwei periodisch fortgesetzten und daher unendlichen Mustern auszuführen. Das Maximum dieser Korrelation gibt die Position der möglichen Drehzentren an, und die Stärke dieses Maximums ist ein Hinweis auf die Wahrscheinlichkeit, dass tatsächlich die gesuchte Drehsymmetrie vorliegt. Eingabe und Ergebnis einer solchen Korrelation ist in Abbildung 23 dargestellt.

Zur Bewertung der Stärke eines Peaks kann die Autokorrelation einer Mediankachel mit sich selbst verwendet werden. Das Minimum $a_{\min}$ dieser Autokorrelation und das Maximum $a_{\max}$ definieren zusammen eine Skala, mit der die Maxima der Korrelationen zweier gegeneinander gedrehter Bilder bewertet werden können. Ergibt sich für eine Drehung ein Maximum der Stärke $c_{\max}$, so bestimmt man mittels dieser Skala einen dimensionslosen Wert c'.

$$c' = \frac{c_{\max} - a_{\min}}{a_{\max} - a_{\min}} \tag{8.16}$$

Dieser Zahlenwert kann streng genommen nicht als Wahrscheinlichkeitswert im Sinne der Statistik aufgefasst werden, da kein geeignetes Modell einer Grundgesamtheit aller Muster besteht. Dennoch gilt die Tendenz, dass es sich bei höherem Wert auch wahrscheinlicher um eine echte Drehung handelt. 0,75 hat sich als geeigneter Grenzwert erwiesen, um über die Existenz einer Drehung zu entscheiden.

Wurde bei der Korrelation ein Maximum signifikanter Stärke ermittelt, so muss die diesem Maximum entsprechende Position bestimmt werden. Rufen wir uns dafür die Formel der Korrelation aus Gleichung 8.8 auf Seite 48 ins Gedächtnis, diesmal jedoch bezogen auf zwei unterschiedliche Bilder p und q. Die weiteren Erklärungen sind einfacher zu formulieren, wenn man von einer Schreibweise mit Indizes zu einer funktionsartigen Schreibweise wechselt, bei der später auch ein einzelner Vektor als Argument dienen kann.

$$a(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} p(m, n) \cdot q((m + x) \bmod M, (n + y) \bmod N) \tag{8.17}$$

Aufgrund der Periodizität des Musters gilt diese Gleichung auch ohne die Modulo-Operationen. Im Folgenden wird auf diese verzichtet, da die Formulierungen so leichter fallen. Verdeutlichen

wir noch, wer gegen wen verschoben wird: Die Position (m, n) in p wird verglichen mit der Position $(m + x, n + y)$ in q. Betrachtet man q als feststehend, so wurde p um den Vektor (x, y) dagegen verschoben; umgekehrt betrachtet ändern sich die Vorzeichen des Verschiebungsvektors. Sei $t = (t_x, t_y)$ der Punkt, an dem die Korrelation ihren maximalen Wert erreicht. Dann sind korrespondierende Punkte aus den beiden Bildern einander hinreichend ähnlich.

$$p(m, n) \approx q(m + t_x, n + t_y) \tag{8.18}$$

$$p(r) \approx q(r + t) \tag{8.19}$$

Es sei p das unveränderte Bild, und q die Version, die durch eine Rotation um den Ursprung, beschrieben durch die Drehmatrix R, daraus hervorgegangen ist. Die Bildinformation an jeder Position r in p wird also an die Position Rr in q abgebildet.

$$p(r) = q(Rr) \tag{8.20}$$

Wenn die beiden Bilder miteinander zur Deckung kommen, so wird ein Drehzentrum auf sich selbst abgebildet. Der rotierte Punkt ist also mit dem Verschobenen Punkt identisch. Diese Gleichung kann verwendet werden, um das Rotationszentrum r zu ermitteln, wie auch in Abbildung 24 illustriert ist.

$$Rr = r + t \tag{8.21}$$

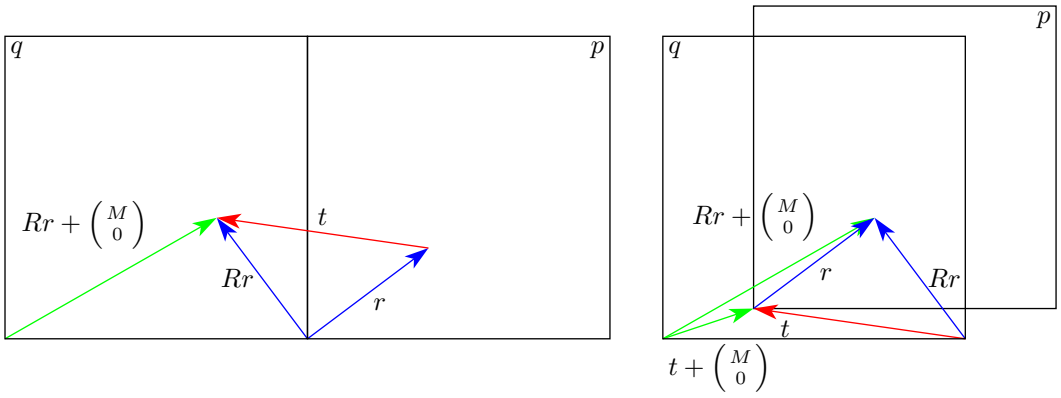
$$Rr - Ir = t \tag{8.22}$$

$$(R - I)r = t \tag{8.23}$$

$$r = (R - I)^{-1}t \tag{8.24}$$

Dabei stellt I die Einheitsmatrix dar. Matrizen mit konkreten Werten für verschiedene Drehwinkel sind aus Tabelle 6 auf Seite 60 zu entnehmen

Streng genommen wird auch das rotierte Bild q in dem Bereich $0 \leq (a, b) < 1$ dargestellt, und nicht direkt an den Koordinaten, die sich aus Rr ergeben. Dieser Sichtweise entsprechen die in Abbildung 24 grün dargestellten Vektoren. Der Unterschied besteht jedoch nur aus der Addition ganzzahliger Vielfacher der Gittervektoren und kann aufgrund der Periodizität des



(a) Die Vektoren in Ausgangslage

(b) Die Vektoren nach Verschiebung

Abbildung 24: Erkennung eines vierzähligen Drehzentrums an der Position r

Musters ignoriert werden. In der Vernachlässigung dieser ganzzahligen Vielfachen liegt auch der Grund, warum in Tabelle 6 auf der nächsten Seite der Bereich möglicher erkannter Drehzentren mitunter außerhalb der Grenzen von sowohl p als auch q liegt, und erst verschobene Kopien ausgedehnte Teile der Standard-Zelle überdecken.

Zusammenfassend kann also aus der Position t des Maximums der Korrelation die Position r eines Rotationszentrums ermittelt werden. Diese Berechnung ist unabhängig vom Koordinatensystem, solange die Rotationsmatrix R entsprechend formuliert ist. Die gesamte Berechnung kann also auch in dem von den beiden Translationsvektoren a und b aufgespannten Koordinatensystem erfolgen.

Es fällt auf, dass wenn man die Fläche einer Translationszelle entsprechend $(R-I)^{-1}$ abbildet, der Bereich möglicher erkannter Drehzentren weniger Fläche einnimmt als eine Translationszelle. Eine Ausnahme von dieser Regel ist die sechszählige Drehung, dort wird genau die Fläche einer Translationszelle abgedeckt. Bei der vierzähligen Drehung ist der Bereich möglicher Rotationszentren auf die halbe Fläche beschränkt, bei der dreizähligen auf ein Drittel, und bei der zweizähligen auf ein Viertel. Dies kommt daher, dass die diesen Drehungen entsprechenden Symmetriegruppen p2, p3, p4 und p6 jeweils vier verschiedene Klassen zweizähliger, drei verschiedene Klassen dreizähliger, zwei Klassen vierzähliger beziehungsweise eine einzige Klasse sechszähliger Drehzentren haben. Dies ist auch aus den Diagrammen in Tabelle 4 auf Seite 20 ersichtlich.

Hat man also ein Drehzentrum gefunden, so sind die Positionen weiterer Drehzentren implizit gegeben. Die gesamte Ebene kann mit kongruenten Parallelogrammen gepflastert werden, wie in Tabelle 6 auf der nächsten Seite dargestellt. Alle diese Parallelogramme enthalten an jeweils der gleichen Position ein Drehzentrum. Daher genügt es, ein einziges Drehzentrum in einem dieser Parallelogramme zu finden, um alle zu kennen. Da eines nicht ohne das andere sein kann, ist der Korrelationsalgorithmus auch nicht in der Lage, einzelne Drehzentren zu erkennen. Das Maximum der Korrelation ist der Tatsache, dass eine Drehung vorliegt, sowie einer Position innerhalb dieser Parallelogramme zugeordnet, nicht jedoch einem konkreten Drehzentrum.

8.5 Spiegelungen

Nach den Translationen sind nun auch die Rotationen erkannt. Es fehlen noch Spiegelungen und Gleitspiegelungen. Der Ansatz hierfür ist prinzipiell der gleiche wie zur Erkennung der Drehzentren: eine unveränderte Elementarzelle wird mit einer gespiegelten Kopie korreliert.

Liegt eine Spiegelung oder Gleitspiegelung vor, so weist die zugehörige Korrelation ein entsprechendes Maximum auf. Die zweidimensionale Position dieses Maximums gibt Auskunft sowohl über die Lage der Spiegelachse (eine Dimension, da die Richtung bereits vorgegeben ist) als auch über die Gleitdistanz einer Gleitspiegelung. Bei einer Distanz von 0 liegt eine einfache Achsspiegelung vor. Diese Distanz wird im Allgemeinen nicht genau 0 sein. Da die Gleitspiegelungen jedoch eine von der Symmetriegruppe vorgegebene Länge haben, lässt sich eine direkte Spiegelung leicht von einer Gleitspiegelung unterscheiden.

Abbildung 25 auf Seite 61 gibt einen Überblick, wie der Reihe nach Eigenschaften des Musters durch Korrelation überprüft werden, um die Symmetriegruppe zu ermitteln. Die Matrizen R_i für Drehungen um Winkel $\frac{2\pi}{i}$ wurden bereits in Tabelle 6 auf der nächsten Seite angegeben. Für die Spiegelungen genügen weitere vier Matrizen. Die erste beschreibt eine Spiegelung an der Winkelhalbierenden der Basisvektoren, und vertauscht dabei deren Rolle.

$$M_{a+b} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{8.25}$$

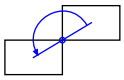
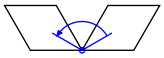
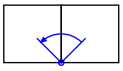
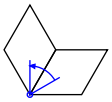
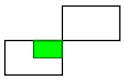
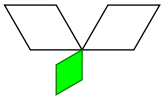
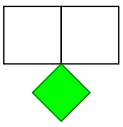
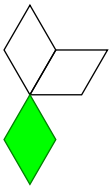
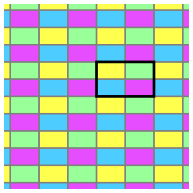
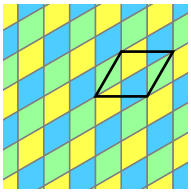
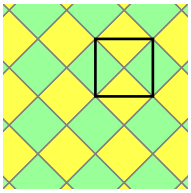
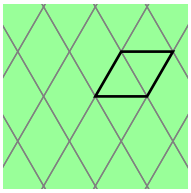
Drehwinkel	$\frac{2\pi}{2}$	$\frac{2\pi}{3}$	$\frac{2\pi}{4}$	$\frac{2\pi}{6}$
Drehmatrix R in (a,b)-Koordinaten	$\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$	$\begin{pmatrix} -1 & -1 \\ 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix}$
Die Zelle und ihr Abbild				
$(R-I)^{-1}$ in (a,b)-Koordinaten	$\frac{1}{2}\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$	$\frac{1}{3}\begin{pmatrix} -1 & 1 \\ -1 & -2 \end{pmatrix}$	$\frac{1}{2}\begin{pmatrix} -1 & 1 \\ -1 & -1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ -1 & -1 \end{pmatrix}$
Mögliche Positionen erkannter Drehzentren				
Positionen innerhalb einer Translationszelle				
Pflasterung mit kongruenten Bereichen				

Tabelle 6: Erkennung verschiedener Rotationszentren

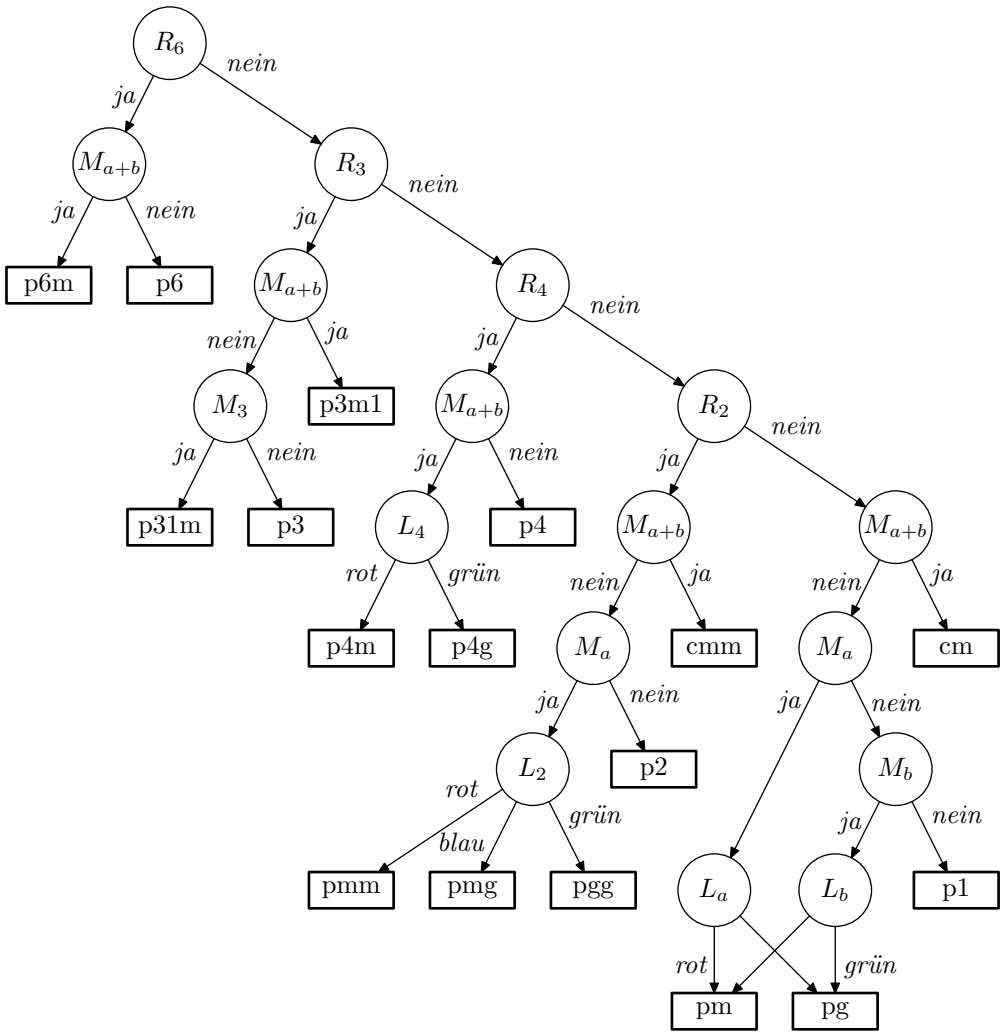
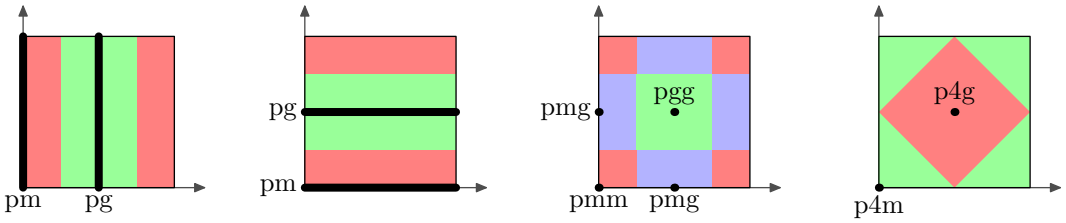


Abbildung 25: Entscheidungsbaum zur Erkennung der Symmetriegruppe



(a) L_a (b) L_b (c) L_2 (d) L_4

Abbildung 26: Positionen der Korrelationspeaks für einzelne Gruppen

Die Spiegelung an einer durch einen der Basisvektoren a oder b beschriebenen Achse kann bei rechtwinkligen Koordinaten durch einen einfachen Vorzeichenwechsel realisiert werden:

$$M_a = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{8.26}$$

$$M_b = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \tag{8.27}$$

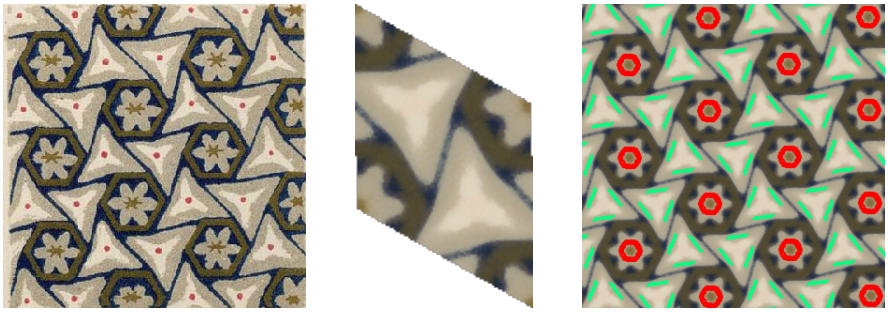
Bei den Gruppen mit dreizähliger Symmetrie muss die schiefe Lage der Basisvektoren berücksichtigt werden, wodurch eine Spiegelung an der durch a beschriebenen Achse die folgende Form hat:

$$M_3 = \begin{pmatrix} 1 & 1 \\ 0 & -1 \end{pmatrix} \tag{8.28}$$

In vielen Fällen genügt allein die Existenz einer hinreichend starken Korrelation, um für oder gegen eine bestimmte Menge von Gruppen zu entscheiden. In anderen Fällen ist es erforderlich, die Position des entsprechenden Peaks zu betrachten, im Entscheidungsbaum durch ein L gekennzeichnet. Kennt man die Basisvektoren sowie ein Drehzentrum mit minimalem Drehwinkel, so lassen sich daraus die Positionen, an denen die einzelnen Spiegelungen auftreten, leicht ermitteln. Der gefundene Peak der unmittelbar vorangegangenen Korrelation wird einfach dem nächstgelegenen dieser Punkte zugeordnet, entsprechend Abbildung 26 auf der vorherigen Seite. Dabei wird das gefundene Drehzentrum als Ursprung betrachtet, was sich entweder durch eine entsprechend neu berechnete Mediankachel oder durch eine geeignete Koordinatentransformation erreichen lässt.

8.6 Hintergrundbild

Nachdem mithilfe der Rotationen und Spiegelungen die Gruppe eindeutig bestimmt wurde, wird erneut eine Mediankachel berechnet. Diesmal wird der Median über alle Pixel eines Orbits gebildet. Das Bild, das man so erhält, erfüllt alle Symmetrien der erkannten Gruppe. Es wird als Hintergrundbild in der Anwendung dargestellt, und kann durch eigene Zeichnungen abgepaust oder ergänzt werden. Abbildung 27 zeigt eine solche abschließende Mediankachel sowie ein im Programm ergänztes Ornament.



(a) Eingabe (b) Mediankachel (c) Erkanntes Ornament und Handzeichnung

Abbildung 27: Ergebnis der Analyse

Teil III
Anwendung und Ausblick

9 Einsatzmöglichkeiten

9.1 Einsatz in der Lehre

9.1.1 Grundschule

Ein möglicher Einsatzbereich des Programms liegt in der Lehre. Dies beginnt auf Grundschulniveau. Durch das Spielen mit dem Programm kann Begeisterung geweckt werden, wie junge Besucher des Mathematik-Museums ix-quadrat immer wieder beweisen. Diese Begeisterung kann dann genutzt werden, um spielerisch Konzepte wie beispielsweise Spiegelsymmetrie zu erklären. Die Schüler können auch aufgefordert werden, eigenständig die zugrunde liegende Systematik zu untersuchen und zu beschreiben.[13]

9.1.2 Sekundarstufe

Mit den mathematischen Werkzeugen der Sekundarstufe können die Symmetrien systematischer untersucht werden. So ist beispielsweise an bayerischen Gymnasien in der siebten Klasse ein ganzer Themenbereich zu Symmetrie und Kongruenz vorgesehen.[14]

Auch im Bereich der Kunsterziehung kann das Programm eingesetzt werden. Zum einen um eigene Muster zu entwerfen, und sie womöglich anschließend mit anderen Werkzeugen auszugestalten. Zum anderen aber auch, um die Konzepte hinter bestehenden Kunstwerken, insbesondere in Architektur oder Grafik, verstehen zu können. Wer ein Ornament in einer der Symmetriegruppen eigenhändig skizzenhaft reproduziert hat, erkennt seine Bestandteile und versteht deren Zusammenspiel.

9.1.3 Hochschule

Dass das Programm bis in Hochschulen hinein interessant ist, zeigt meine eigene Begegnung mit der Vorgängerversion. Es kann einen anschaulichen Zugang zu den sonst eher trockenen Axiomen der Gruppentheorie liefern und schon allein durch die interessante Optik die Aufmerksamkeit des Auditoriums wecken. Auch als Experimentierwerkzeug für die Lernenden ist das Programm hervorragend geeignet.

Durch die enge Verwandschaft zu Refexionsgruppen und Coxetergruppen bieten sich auch bei fortgeschrittenen Themen der Gruppentheorie noch lehrreiche Anwendungsmöglichkeiten.

Der Einsatz ist jedoch nicht auf die Mathematik und Informatik beschränkt. Bei kristallographischen Themen in der Physik und Chemie kann das Programm helfen, Konzepte im zweidimensionalen leichter zu verstehen, die sich anschließend auf dreidimensionale Fälle übertragen lassen.

9.2 Professioneller Einsatz

Auch Künstlern bieten sich mit diesem Programm vielfältige Einsatzmöglichkeiten. Die wenigsten werden wohl komplette Kunstwerke mit meinem Programm entwerfen, ausdrucken und anschließend unverändert verkaufen. Ich hatte jedoch bereits mit zwei Künstlern Kontakt, die mein Programm in der Entwurfsphase für eigene Werke einsetzen. Gemessen an der Gesamtzahl von

Rückmeldungen zu diesem Programm scheinen die Künstler einen ausgesprochen hohen Anteil der Nutzergemeinde auszumachen.

Der Einsatzbereich ist dabei nicht auf Malerei und Grafik beschränkt. So wird in einem Fall das Programm verwendet, um die exportierten Daten mittels einer CNC-Fräse zu Stempeln zu verarbeiten, die dann zum Prägen handgebundener Bücher verwendet werden. Auch bei der Arbeit mit Textilien bieten sich Anwendungsmöglichkeiten, und zwar nicht nur beim Bedrucken von Stoffen. So bietet gerade die Unterteilung in veränderbare Fundamentalzellen eine inspirierende Vorlage für Patchwork und Quiltarbeiten.

Neben Künstlern, bei denen Ornamente Hauptelemente ihrer Arbeit ausmachen, gibt es noch einige weitere Anwendungsfelder, bei denen Ornamente eine untergeordnete Rolle spielen. So können exportierte kachelbare Rastergrafiken beispielsweise von Webdesignern als Hintergrundbilder verwendet werden.

9.3 Heimgebrauch

Doch man muss nicht unbedingt professionell Kunstschaffender sein, um sich das Programm im Alltag zunutze zu machen. Die Gestaltung ansprechender Muster kann entspannend und inspirierend wirken und die Kreativität der Nutzer fördern. Doch auch ganz abgesehen von diesem objektiven Nutzen: das Zeichnen von Ornamenten macht einfach Spaß, und da Computer für viele Leute mehr ein Unterhaltungs- denn ein Arbeitsgerät sind, wäre sogar dieser Spaßfaktor allein Daseinsberechtigung genug.

Für ein breites, wissenschaftlich interessiertes Publikum hat das Programm durch einen Link in dem Magazin P.M. kürzlich einige Bekanntheit erlangt[21].

9.4 Ausstellungen

Auch Ausstellungen bringen die Mathematik der Ornamente einem breiten Publikum näher. Dazu gehört natürlich die Dauerausstellung im ix-quadrat in Garching. Anlässlich des Jahrs der Mathematik 2008 ist das Programm als Teil der Wanderausstellung „Imaginary“ des Mathematischen Forschungsinstituts Oberwolfach unterwegs durch Deutschland[17]. Außerdem laufen Planungen, das Programm eventuell in das Mineralienmuseum in Oberwolfach zu integrieren.

10 Ausblick

10.1 Weiterentwicklung

Das Ornamente-Programm ist nicht vorrangig Bestandteil dieser Diplomarbeit, der zusammen mit dem Abschluss der Arbeit abgeschlossen wäre. Vielmehr ist es ein lebendiges Software-Projekt, das sich ständig im Wachstum befindet. Schon jetzt bestehen einige Ideen, in welche Richtung sich das Programm weiterentwickeln sollte.

Für die unmittelbare Zukunft ist einige Arbeit an der noch jungen Implementierung der Mustererkennung geplant. Wie im Text mehrfach angesprochen gilt es, verschiedene Heuristiken systematisch zu vergleichen. Außerdem muss die Funktionalität noch besser mit dem Rest des Programms, insbesondere den verschiedenen Exportformaten, verknüpft werden.

Die nächste größere Neuerung, die geplant ist, ist die Ausweitung auf Fries- und Rosettengruppen. Insbesondere für die Rosettengruppen ist in weiten Teilen eine neue Benutzeroberfläche erforderlich, wenn man eine große Zahl möglicher Drehwinkel zulassen will. Das interne Konzept mit den in Abschnitt 5.1 auf Seite 26 beschriebenen Puffern ist in diesen Fällen zumindest nicht

mehr unverändert anzuwenden, wodurch größere Änderungen in vielen Programmteilen erforderlich werden.

Ein weiteres Ziel ist eine Automatisierung der Mustergenerierung. Dem Menschen, und sei er künstlerisch noch so unbegabt, fällt es nicht schwer, in Gruppen hoher Symmetrie ästhetisch reizvolle Muster zustande zu bringen. Das sollte der Computer auch lernen. Mögliche Einsatzbereiche wären die Projektion an die Außenwand des ix-quadrats außerhalb der Öffnungszeiten, aber auch die Verwendung auf dem heimischen PC als Bildschirmschoner.

10.2 Folgeprojekte

Neben diesen Plänen für die Fortentwicklung des Programms für euklidische Elemente ist ein eigenständiges Programm zum Zeichnen in den Gruppen der hyperbolischen Geometrie in Entwicklung. Das Zeichnen in Echtzeit ist bereits implementiert, und ein dadurch entstandenes Bild in Abbildung 28 dargestellt. Als besonderes Problem zeigt sich die Navigation der möglichen Gruppen, da deren Zahl unendlich ist, im Unterschied zu den 17 Gruppen der euklidischen Ebene. Während andere existierende Programme auf dem Bereich der hyperbolischen Geometrie an dieser Stelle oft die Eingabe unintuitiver symbolischer Beschreibungen erfordern, ist in dem geplanten Projekt eine intuitivere Navigation vorgesehen. Insgesamt ist dieses Projekt eines Zeichenprogramms für hyperbolische Ornamente als Thema meiner Dissertation geplant.

Vielleicht ist es auch möglich, die Erkennung euklidischer Muster mit dem Zeichnen in hyperbolischen Symmetriegruppen zu kombinieren. Aufgrund der unterschiedlichen Oberflächenkrümmung kann der Fundamentalbereich eines euklidischen Musters niemals genau der Fundamentalbereich eines hyperbolischen Ornamentes sein. Eventuell ist es jedoch möglich, die Abweichung so zu gestalten, dass sie sich harmonisch in das Muster einfügt und nicht störend wahrgenommen wird.

Abbildung 28: Ein hyperbolisches Ornament, erzeugt mit dem in Entwicklung befindlichen Nachfolgeprojekt

11 Verzeichnisse

11.1 Literatur

[1] Adobe Systems, Inc. (Hrsg.): *PDF Reference*. Sixth edition, version 1.7, November 2006. http://www.adobe.com/devnet/acrobat/pdfs/pdf_reference_1-7.pdf

[2] Adobe Systems, Inc. (Hrsg.): *PostScript language reference*. Third edition. Addison-Wesley, 1999 http://www.adobe.com/products/postscript/pdfs/PLRM.pdf. – ISBN 0–201–37922–8

[3] *Kapitel* 4.6.2: Encoded User Paths. In: [2], 200-202

[4] Conway, John H. ; Huson, Daniel H.: The Orbifold Notation for Two-Dimensional Groups. In: *Structural Chemistry* 13 (2002), August, Nr. 3-4, S. 247–257. – DOI 10.1023/A:1015851621002

[5] Fortune, Steven: A sweepline algorithm for voronoi diagrams. In: *Algorithmica* 2 (1987), März, Nr. 1, S. 153–174. – DOI 10.1007/BF01840357

[6] Gagern, Martin von: *WirteBird*. http://martin.von-gagern.net/projects/writebird/

[7] Grünbaum, Branko ; Shephard, G. C.: *Tilings and Patterns*. W. H. Freeman and Company, 1987. – ISBN 0–7167–1193–1

[8] Hahn, Theo (Hrsg.): *International Tables for Crystallography*. Bd. A: *Space Group Symmetry*. 5th. Springer and IUCr, 2005. – DOI 10.1107/97809553602060000100. ISBN 978–0–7923–6590–7

[9] Sun Microsystems, Inc. (Hrsg.): *Java Web Start 1.4.2 Developer Guide*. http://java.sun.com/j2se/1.4.2/docs/guide/jws/developersguide/contents.html

[10] Jones, Owen: *The Grammar of Ornament*. Folio edition. Bernard Quaritch, 1910 http://digital.library.wisc.edu/1711.dl/DLDecArts.GramOrnJones

[11] Krausse, Rüdiger: *Ornament*. http://www.math.tu-dresden.de/~ganter/ornament.html

[12] Krimmel, Bernd (Hrsg.): *Symmetrie in Kunst, Natur und Wissenschaft*. Mathildenhöhe Darmstadt, 1986

[13] Kultus, Bayerisches Staatsministerium für Unterricht und (Hrsg.): *Lehrplan für die bayerische Grundschule*. 2000 http://www.isb.bayern.de/isb/index.asp?QNav=4&STyp=1

[14] Kultus, Bayerisches Staatsministerium für Unterricht und (Hrsg.): *Lehrplan für das bayerische Gymnasium*. 2003 http://www.isb.bayern.de/isb/index.asp?QNav=4&STyp=14. – ISBN 3–93708200–X

[15] Lin, Hsin-Chih ; Wang, Ling-Ling ; Yang, Shi-Nine: Extracting periodicity of a regular texture based on autocorrelation functions. In: *Pattern Recognition Letters* 18 (1997), S. 433–443

[16] Liu, Yanxi ; Collins, Robert T. ; Tsin, Yanghai: A computational model for periodic patterns. In: *IEEE transactions on pattern analysis and machine intelligence* 26 (2004), März, Nr. 3, S. 354–371

[17] Oberwolfach, Mathematisches Forschungsinstitut: *Imaginary.* http://www.imaginary2008.de/

[18] Reddy, Sravana: On improving the performance of the wallpaper symmetry group classification algorithm. Robotics Institute. Carnegie Mellon University. Pittsburgh, Pennsylvania. Oktober 2005 (CMU-RI-TR-05-49)

[19] Rehm, Klaus: Im Spiegel der Symmetrie. In: *BerliNews*, 13.11.2002. http://www.berlinews.de/artikel.php?14701

[20] Richter-Gebert, Jürgen ; Kortenkamp, Ulrich: *Cinderella.* http://cinderella.de/

[21] Schirawski, Nicolai: Symmetrie. In: *P.M.* (2008), Nr. 2, S. 14–20. – ISSN 1863–9313

[22] Schoeni, Nicolas ; Hardaker, Wes ; Chapuis, Gervais: *Escher Web Sketch.* http://escher.epfl.ch/escher/

[23] Shepperd, Michael: *crystallographic classification system for border patterns.* http://michaelshepperd.tripod.com/resources/groups.html

[24] Weeks, Jeff: *Kali.* http://www.geometrygames.org/Kali/

[25] Weisstein, Eric W.: *MathWorld – A Wolfram Web Resource.* Version: 2007. http://mathworld.wolfram.com/

[26] *Kapitel* Fundamental Domain. In: [25]

[27] Wikimedia Foundation, Inc. (Hrsg.): *Wikipedia.* Deutsch. http://de.wikipedia.org/

[28] *Kapitel* Erweiterter euklidischer Algorithmus. In: [27]

[29] World Wide Web Consortium (Hrsg.): *Scalable Vector Graphics (SVG) 1.1 Specification.* Januar 2003. http://www.w3.org/TR/SVG11/

11.2 Definitionen

Definition 1 Isometrie 10
Definition 2 Automorphismus 11
Definition 3 Symmetriegruppe 11
Definition 4 Generator 12
Definition 5 Orbit 13
Definition 6 Häufungspunkt 13
Definition 7 diskrete Gruppe 13
Definition 8 Äquivalente Gruppen 13
Definition 9 Struktur 14
Definition 10 Translationszelle 21
Definition 11 Fundamentalbereich 29
Definition 12 Voronoi-Polygon 31
Definition 13 Delaunay-Dreieck 32
Definition 14 Delaunay-Triangulierung 32

11.3 Abbildungsverzeichnis

1 Screenshot der Anwendung . . . 6
2 Kinder beim Zeichnen von Ornamenten im ix-quadrat . . . 8
3 Die vier Symmetriearten . . . 11
4 Ein symmetrisches Ornament . . . 12
5 Diskrete und nicht diskrete Symmetriegruppen . . . 14
6 Schritte der Orbitberechnung . . . 22
7 Algorithmus zur Erzeugung des Orbits eines Punktes . . . 24
8 Translationszellen und überlappende Puffer . . . 26
9 Schritte zur Bestimmung von Überdeckungen . . . 28
10 Verschiedene Fundamentalzellen der Gruppe p3 . . . 30
11 Voronoi-Zellen und Delaunay-Triangulierung . . . 33
12 Schritte zur Beschränkung des Einflussbereiches . . . 34
13 Illustration zum Beweis von Lemma 1 . . . 34
14 Erweiterter euklidischer Algorithmus . . . 36
15 Illustration zum Beweis von Lemma 7 . . . 38
16 Illustration zum Beweis von Lemma 8 . . . 39
17 Illustration zum Fall 2 des Beweises von Satz 9 . . . 39
18 Folgen eines unpräzisen Umkreisprädikates . . . 42
19 Das Eingabebild und seine Autokorrelation . . . 47
20 Effekt der Normalisierung (bei symmetrischem Wertebereich) . . . 51
21 Die Schritte zur Ermittlung von Peaks . . . 52
22 Die in Gleichung 8.11 beschriebene Bewertungsfunktion . . . 54
23 Korrelation zur Erkennung einer sechszähligen Drehung . . . 57
24 Erkennung eines vierzähligen Drehzentrums an der Position r . . . 58
25 Entscheidungsbaum zur Erkennung der Symmetriegruppe . . . 61
26 Positionen der Korrelationspeaks für einzelne Gruppen . . . 61
27 Ergebnis der Analyse . . . 62
28 Ein hyperbolisches Ornament . . . 65

11.4 Tabellenverzeichnis

1 Beispiele von Ornamenten mit Rosettengruppen . . . 17
2 Beispiele von Ornamenten mit Friesgruppen . . . 17
3 Beispiele von Ornamenten mit kristallographischen Gruppen . . . 19
4 Übersicht über die ebenen kristallographischen Gruppen . . . 20
5 Innere Symmetrien in (a,b)-Koordinaten . . . 25
6 Erkennung verschiedener Rotationszentren . . . 60

11.5 Index

1, 18
2, 18

(a,b)-Koordinaten, 21
Abbildung, 10, 56
Abstand, *siehe* Distanz
Achse, 26, 46
Äquivalente Gruppen, 13
Äquivalenzklasse, 13, 14
affine Abbildung, 13, *siehe* affine Transformation
affine Transformation, 15, 56
Amplitude, 50
Antialiasing, *siehe* Kantenglättung
Anwendung, 10
Anwendungsbereich, 7
Applet, 7, 46
Area of Dominance, *siehe* Dominanzgebiet
arithmetisches Mittel, 56
assoziativ, 12
Atari, 7
Ausschnitt, 24, 33, 45, 46
Ausstellung, 7, 64
Autokorrelation, 47
automatische Mustererkennung, *siehe* Mustererkennung
Automorphismus, 11

Bandornamentgruppe, *siehe* Friesgruppe
Basisvektor, 21, 45, 53
Baum, *siehe* Quadtree
Benennung, *siehe* Bezeichnung
Berg, 53
Bewegung, *siehe* Isometrie
Bewertung, 53, 55
Bezeichnung, 14, 16, 18
Bild, 11
Bitmapformat, *siehe* Rasterformat
Bruch der Symmetrie, *siehe* Symmetriebruch

Cache, 50

Datei, 43, 46, 50
Delaunay-Dreieck, 32
Delaunay-Triangulierung, 32
DFT, *siehe* Fourier-Transformation
Diedergruppe, 16
Differenz, 47
Dimension, 10
dimensionslos, 57
diskrete Gruppe, 13
Distanz, 11, 33, 59
Dominanzgebiet, 50, 54, 56
DOS, 7
Drehgruppe, *siehe* zyklische Gruppe
Drehsinn, 10
Drehung, *siehe* Rotation
Drehzentrum, 11
Dreiecksungleichung, 34

E-Mail, 46
Ebene, 6, 10
Ecke, 27, 31, 33, 45
Einheitswurzel, 49
einrasten, 31
Entfernung, 53
erweiterter euklidischer Algorithmus, 35
euklidisch, 6, 10
-er Algorithmus, *siehe* erweiterter euklidischer Algorithmus
Exponat, 7
Exponentialfunktion, 48

Farbe, 18, 50, 56
Feature, 7
Fenster, 51, 53
FFT, *siehe* Fourier-Transformation
Fläche, 44, 50, 59
Form, 7, 10, 21, 23
Fourier-Transformation, 48, 57
Freiheitsgrad, 31
Friesgruppe, 18
Fundamentalbereich, 21, 29
Fundamentalzelle, *siehe* Fundamentalbereich

g, 18
Generator, 12
generieren, *siehe* Generator
geometrische Reihe, 49
Gewichtung, 53, 54
Gitter, 7, 33, 45, 50, 53

Gitterpunkt, 45, 53
Gittervektor, 46, 53
Gleitfenster, *siehe* Fenster
Gleitspiegelung, 11, 59
Gradient, 53
Grafiktablett, 7
Graustufen, 50
Grenzwert, 32
Gruppe, 12
Größe, 7, 26
Größenordnung, 55

Hauptachse, *siehe* Achse
Hermann-Mauguin-Symbol, 18
Heuristik, 54–56
Hilfskoordinaten, 21
Hilfslinie, 7
Hintereinanderausführung, *siehe* Verkettung
Hintergrundbild, 46, 62
homogene Koordinaten, 14
hyperbolisch, 65
Häufungspunkt, 13

identische Abbildung, *siehe* Identität
Identität, 11, 12, 16
innere Abbildung, *siehe* innere Symmetrie
innere Symmetrie, 23, 44
intuitiv, 7, 31
Invarianz, 10
inverse Abbildung, *siehe* Umkehrabbildung
Isometrie, 10
Isomorphismus, 13
iterativ, 55
ix-quadrat, 63

JAR, 8
Java, 7, 23

Kandidat, *siehe* Peakkandidat
Kantenglättung, 44
kartesische Koordinaten, 14, 21
Kind, 51
kleinste Fehlerquadrate, 56
Knopf, 7
kollinear, 32
Kompatibilität, 43
komplexe Zahlen, 48
Kongruenzabbildung, *siehe* Isometrie
Kongruenzbewegung, *siehe* Isometrie
Konjugation, 13
  komplexe, 48
Konstruktion, 9
Kontrollpunkt, 7, 31, 33
konvex, 31
Koordinatensystem, 21, 23, 45, 57, 59
Korrelation, 47, 57
kozirkular, 32
Kreis, 32
Kristallographie, 18
kristallographische Gruppe, 18
kurz, 56

Least Squares, *siehe* kleinste Fehlerquadrate
Linearkombination, 21
Länge, 10, 54
Längenmaß, 10

m, 18
Mannigfaltigkeit, 10
Matrix, 15
Maximum, 47
Maß, *siehe* Längenmaß
Median, 56, 62
Menü, 7
Mittel, *siehe* arithmetisches Mittel
Modulo, 48, 49, 57
Motiv, *siehe* Fundamentalbereich
Museum, 7
Muster, 11
Mustererkennung, 46

Name, *siehe* Bezeichnung
Normalisierung, 50
Normierung, 54

Obergruppe, 13, 55
Orbifold-Notation, 18
Orbit, 6, 13, 21, 62

Parallelogramm, 21
Parkettierung, 29
PDF, 43
Peak, 50, 53
Peakkandidat, 51, 55
periodisch, 57
Pixel, 26, 47, 53
Portierung, 7
Position, 7, 57, 59

PostScript, 44
Programmiersprache, 16, 44
Puffer, 26
Punktemenge, 13
Punktgruppe, *siehe* Rosettengruppe
Punktspiegelung, 11, 18

Quadrat, 51, 56
Quadtree, 51

Raster, 26
Rasterformat, 45
Raum, 10
Rechteck, 26, 27, 44, 45
Reflexion, 11, 14, 16, 59
Reihenfolge, 18, 56
Ressource, 7
Rosettengruppe, 16
Rotation, 11, 14, 46, 56
runden, 22, 29, 37

Scherung, 15
Schleife, 23, 26, 42, 43
Schranke, 13, 33
Selbstähnlichkeit, 47
Skalarprodukt, 47
Sliding Window, *siehe* Fenster
Speicher, 46, 49
Spiegelachse, 11
Spiegelung, *siehe* Reflexion
Spline, 9
Standard-Translationszelle, 22, 33, 56, 59
Stelle, 53, 55
Stift, 7, 43, 45
Stimmigkeit, 53, 54
Streckung, 15
Struktur, 13, 23
Strukturdiagramm, 16
Stufe, 56
SVG, 44
Sweepline-Algorithmus, 32
Symmetrie, 10
Symmetriebruch, 18
Symmetriegruppe, 11

Teiler, 35
Teilmenge, 15
Transformation, 14
    Fourier, *siehe* Fourier-Transformation
Translation, 11, 14, 16, 47, 53
Translationszelle, 7, 21, 26

Überlappung, 26, 27, 47, 54
Umgebung, 13
Umkehrabbildung, 10, 12
Umkreis, 32
Umriss, 44
Untergruppe, 55
Unterstruktur, 53
unverändert, 11

Vektor, 14
Vektorformat, 43
Vergleich, 50, 56
Verkettung, 10, 12, 15
Verschiebung, *siehe* Translation
Vielfache, 49, 53, 58
Vollständigkeit, 53, 54
Voronoi-Diagramm, 31, 32
Voronoi-Polygon, 31
Voronoi-Zelle, *siehe* Voronoi-Polygon

Wahrscheinlichkeit, 57
WebStart, 46
Wert, 50
Wertebereich, 47
Winkel, 10, 11, 37, 46

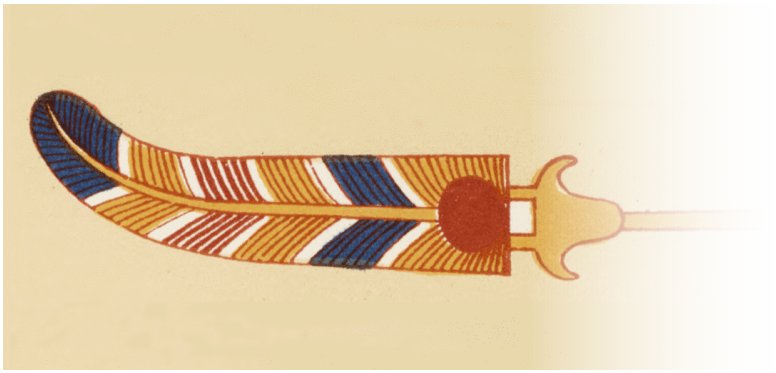
Zeichenebene, *siehe* Ebene
Zerlegung, 29
Zweierpotenz, 49
zyklisch, 48, 57
zyklische Gruppe, 16
n-zählig, 11

Teil IV
Anhang

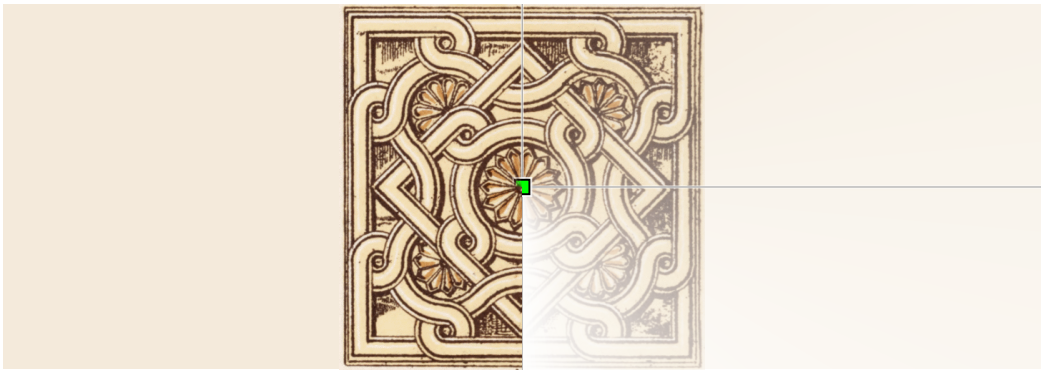
12 Beispielornamente

12.1 Rosettengruppen

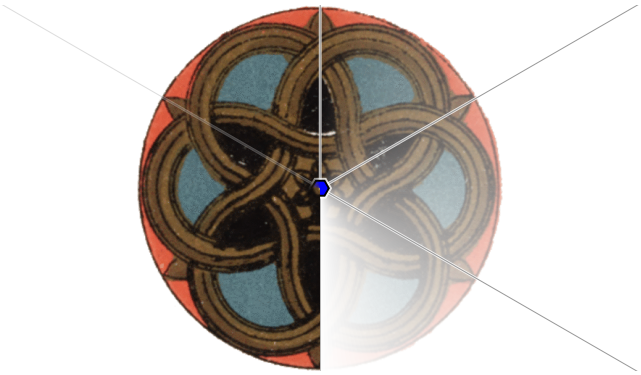
C_1



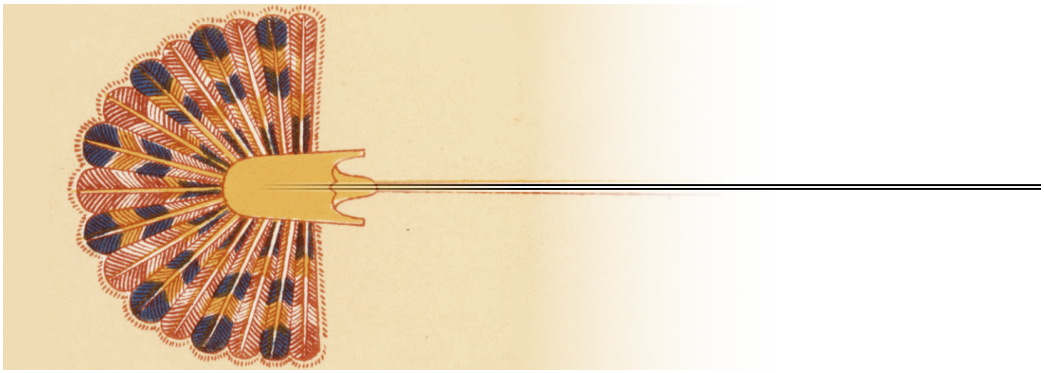
C_4



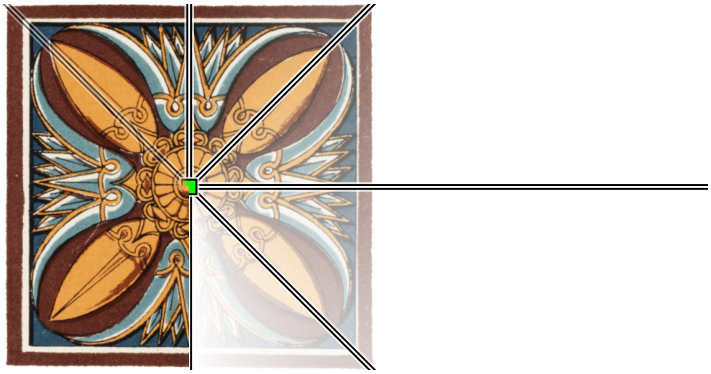
C_6



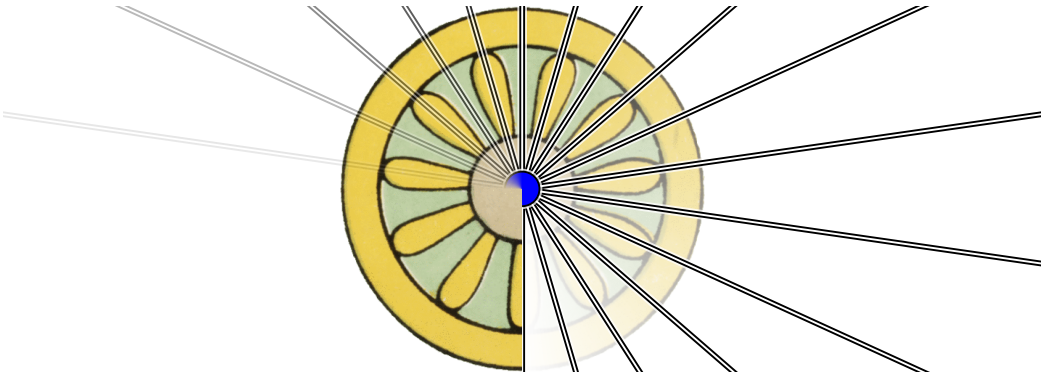
D_1



D_4

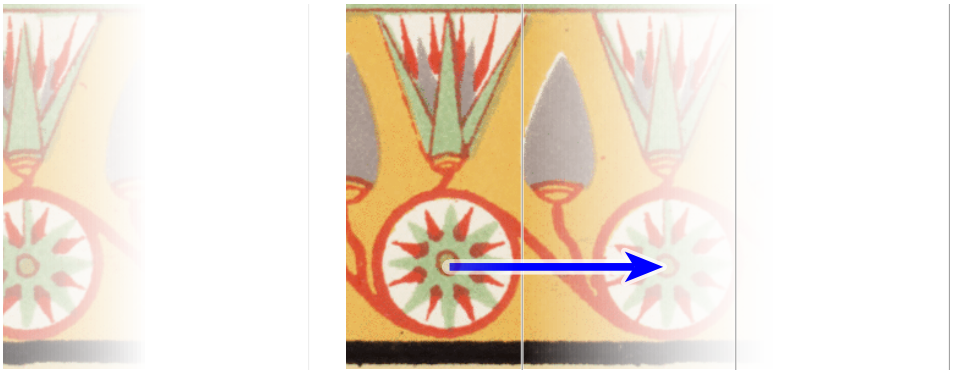


D_{11}

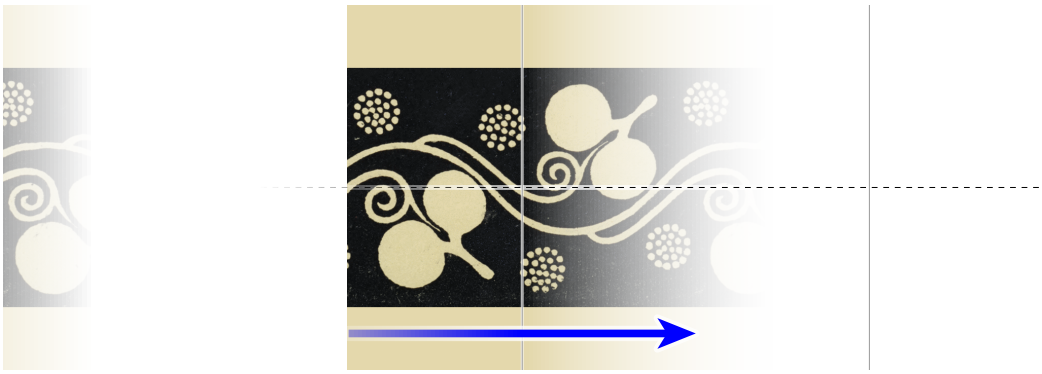


12.2 Friesgruppen

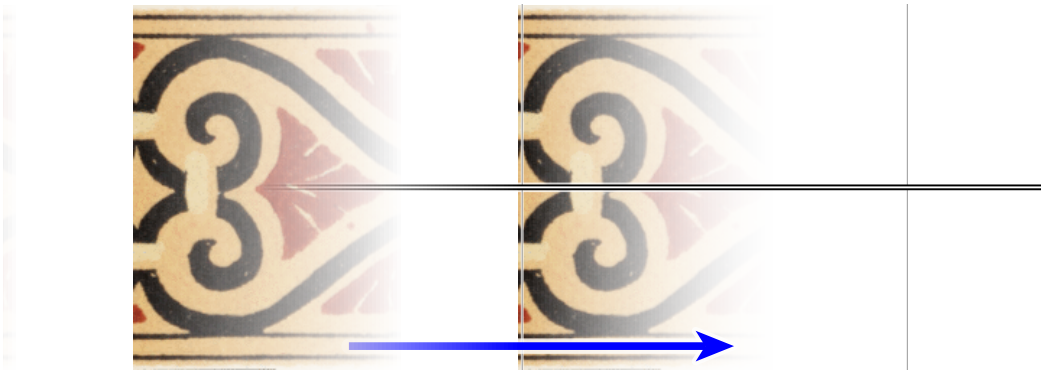
11



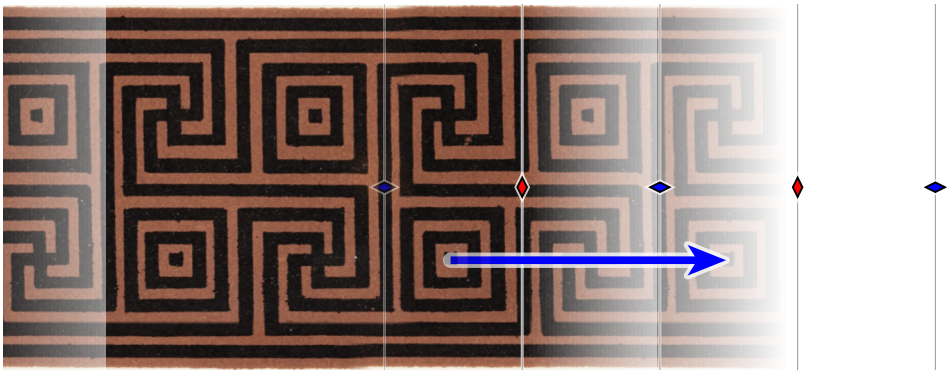
1g



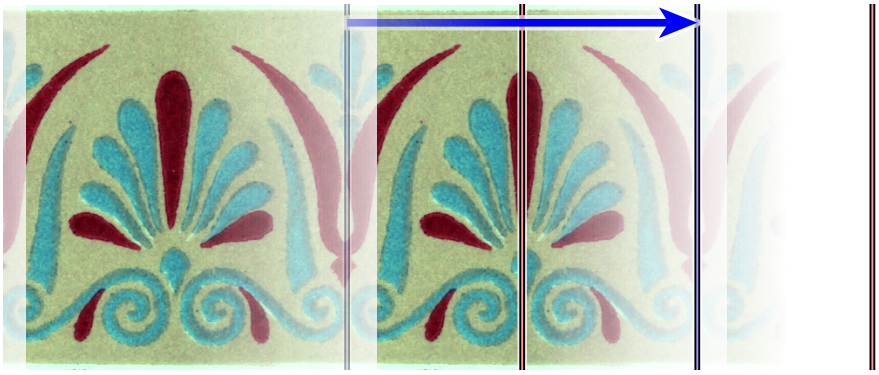
1m



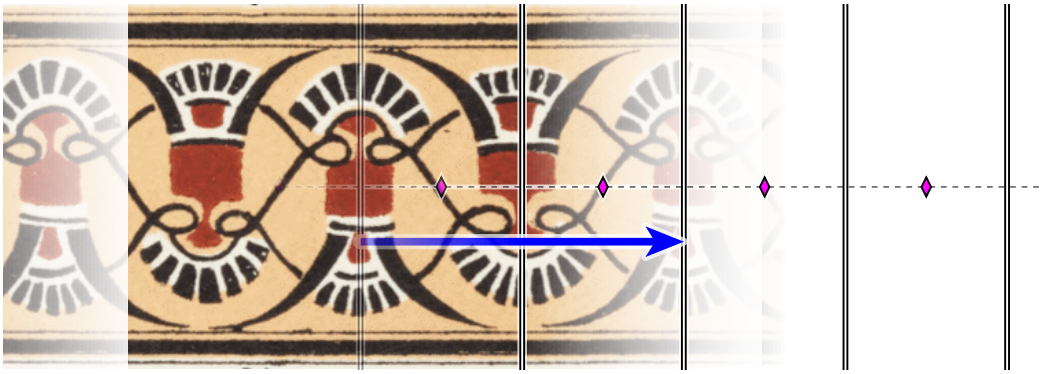
12



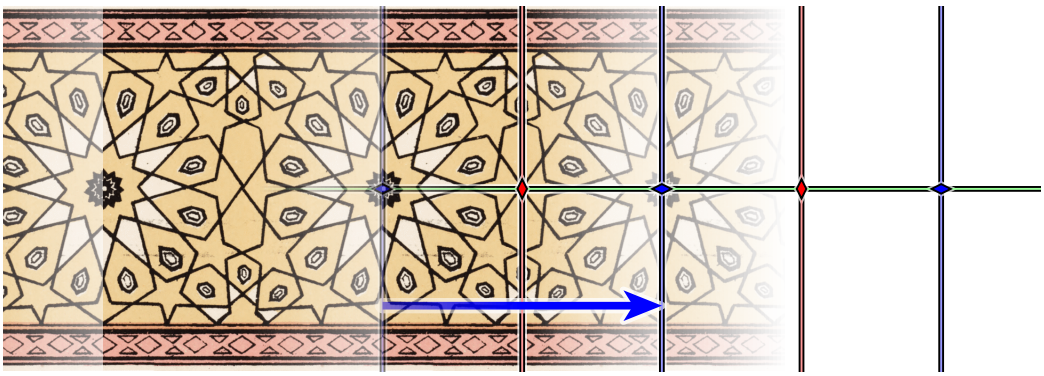
m1



mg

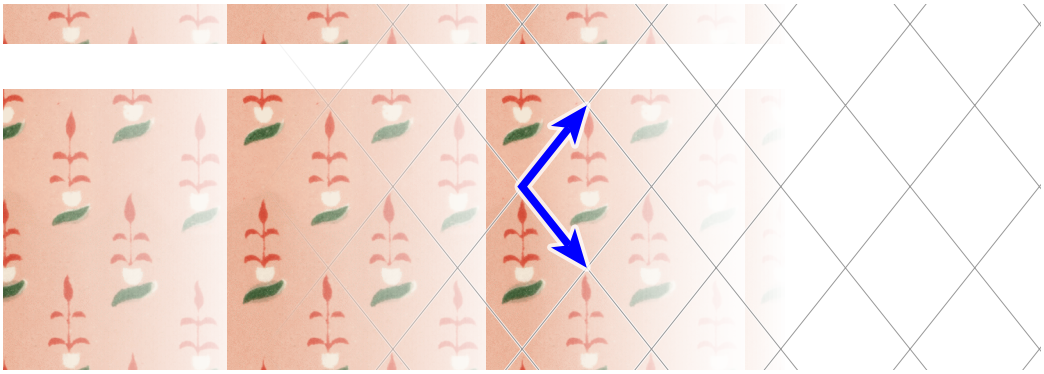


mm

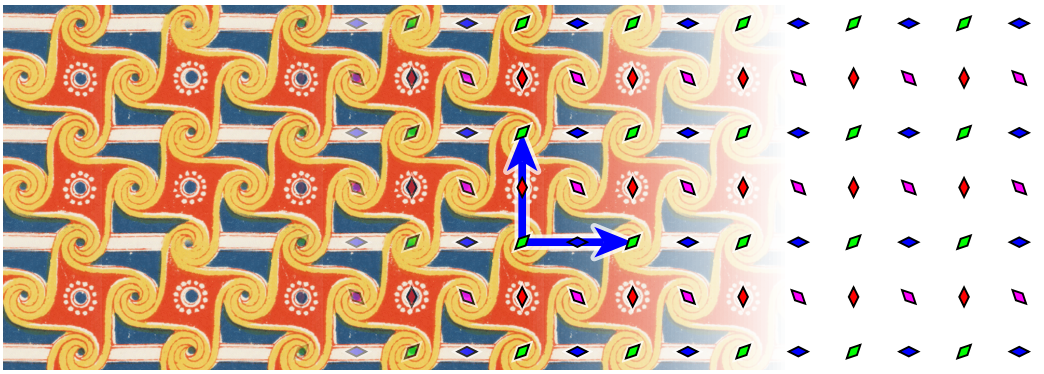


12.3 Kristallographische Gruppen

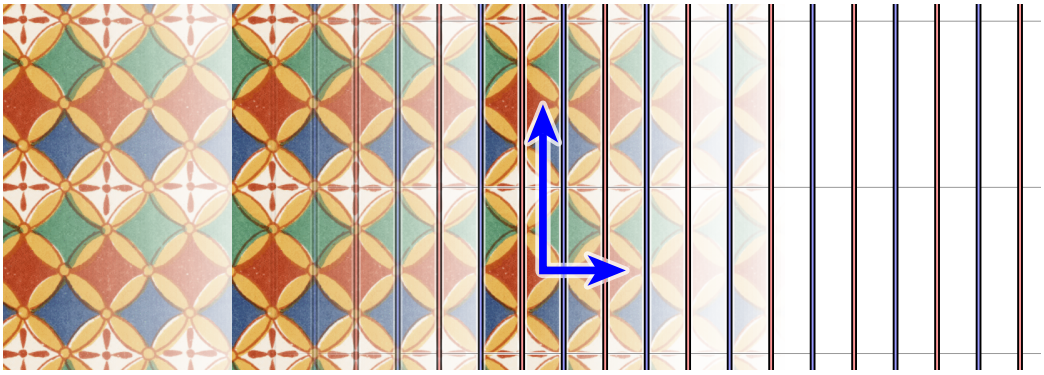
p1



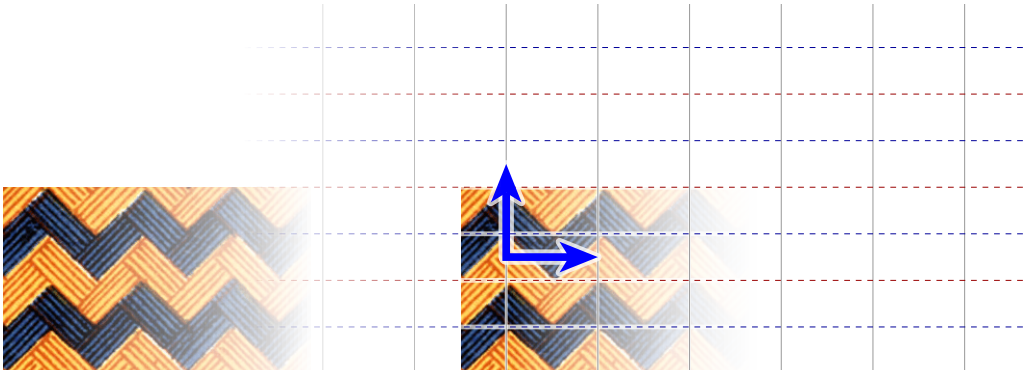
p2



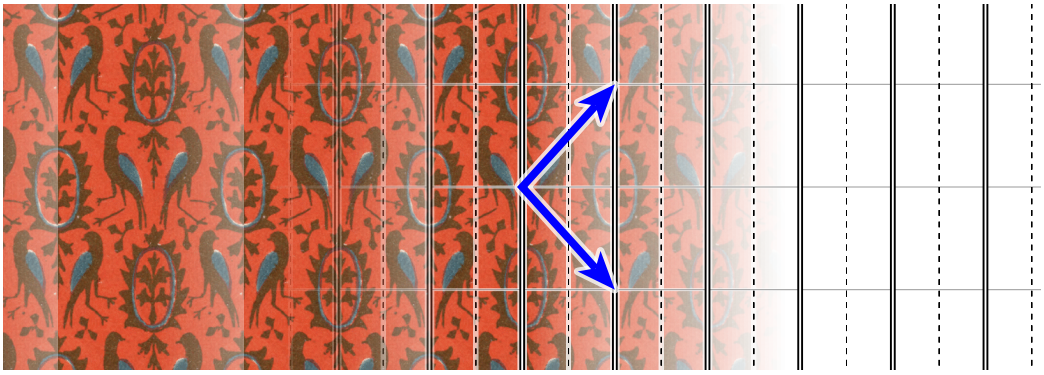
pm



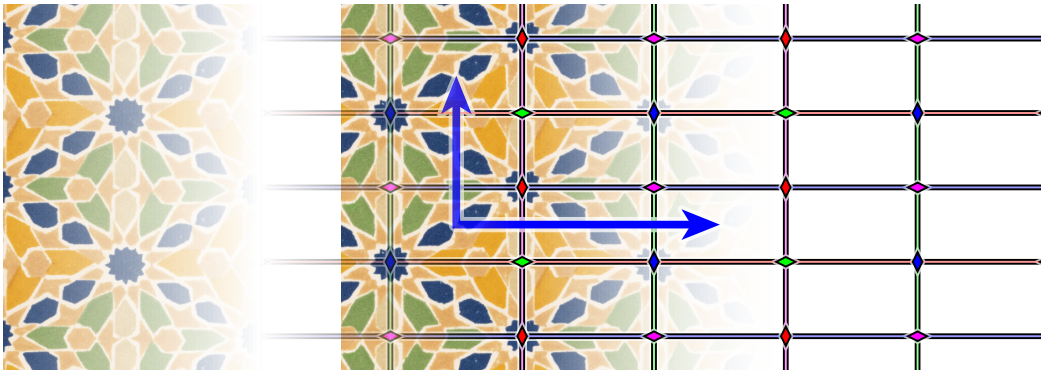
pg



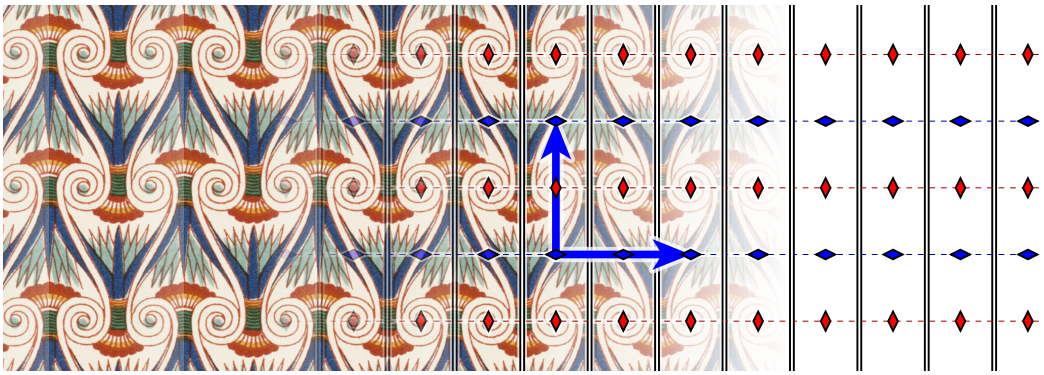
cm



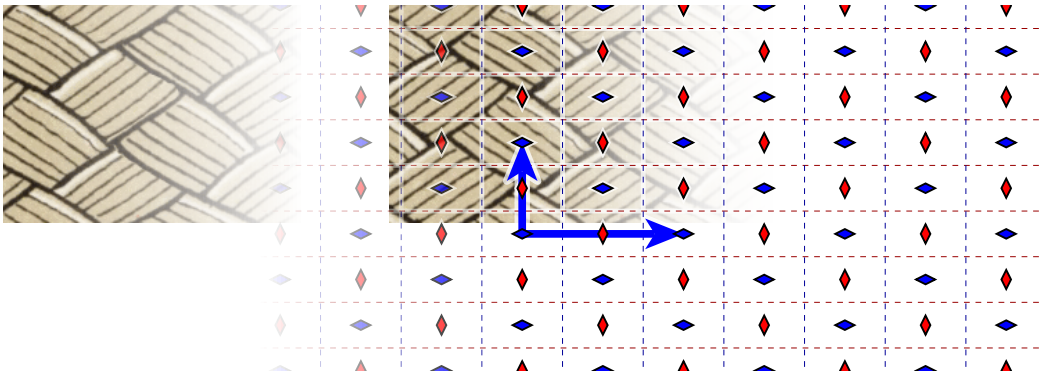
pmm



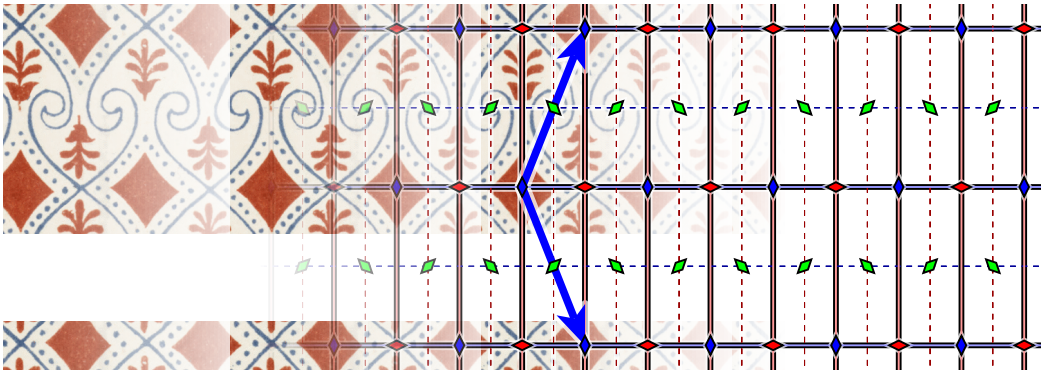
pmg



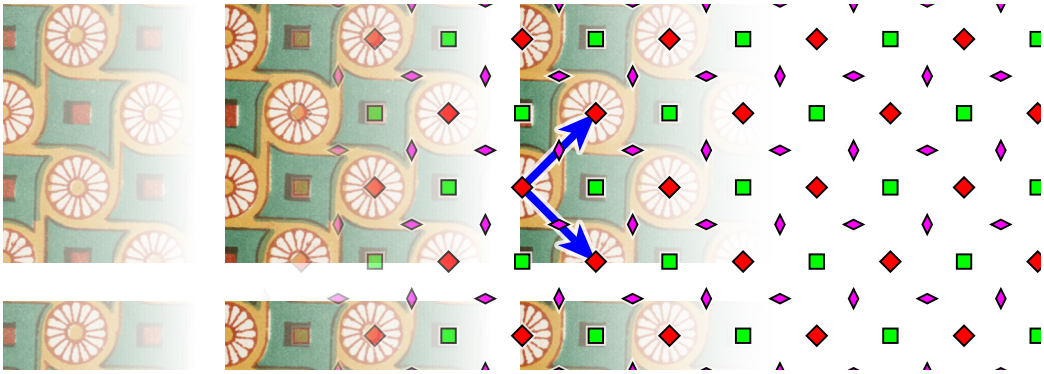
pgg



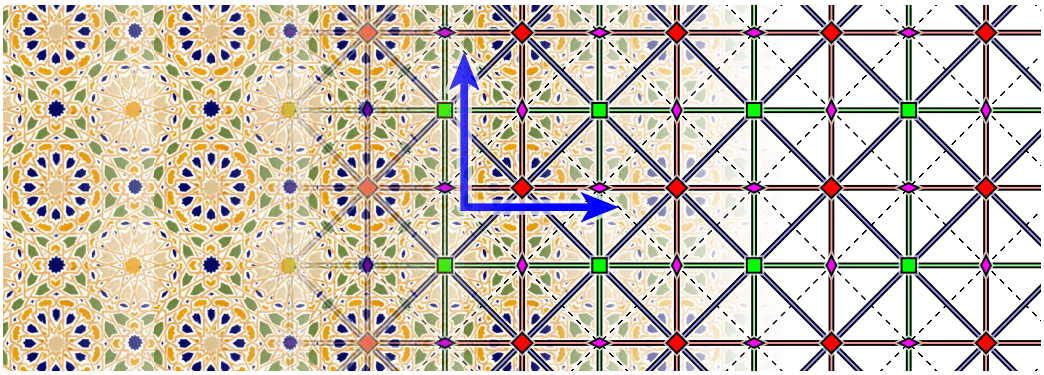
cmm



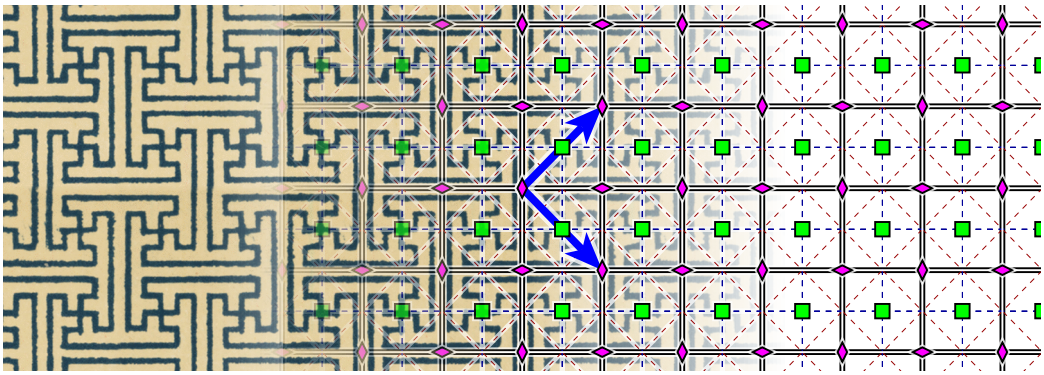
p4



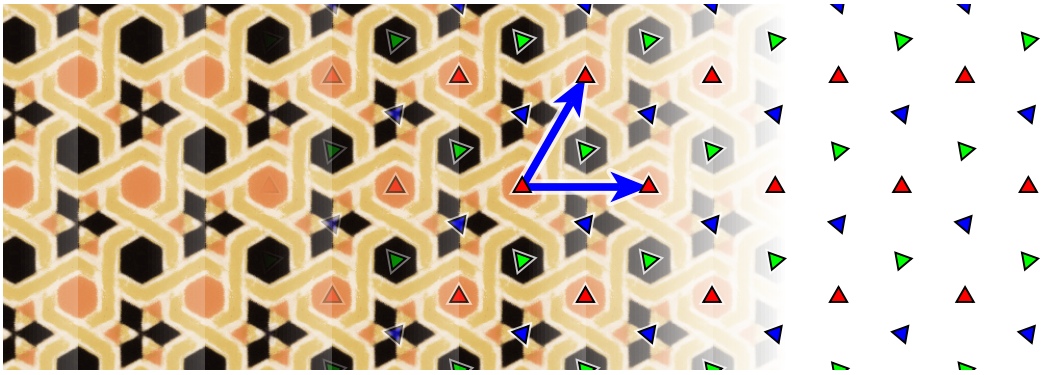
p4m



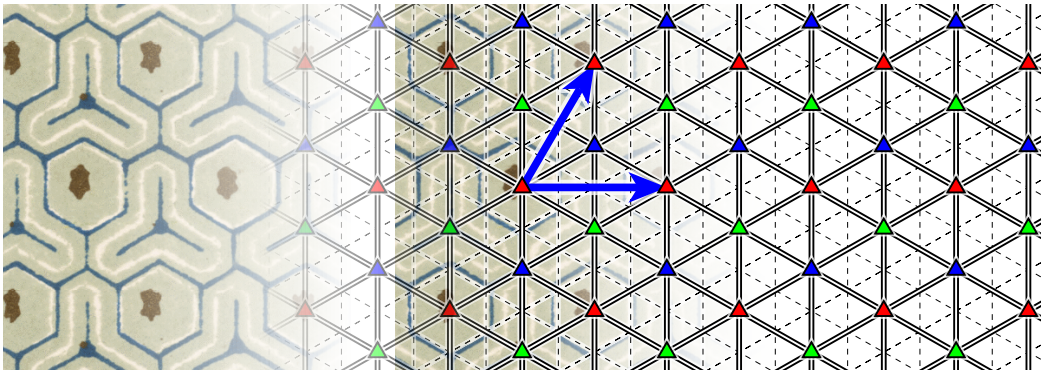
p4g



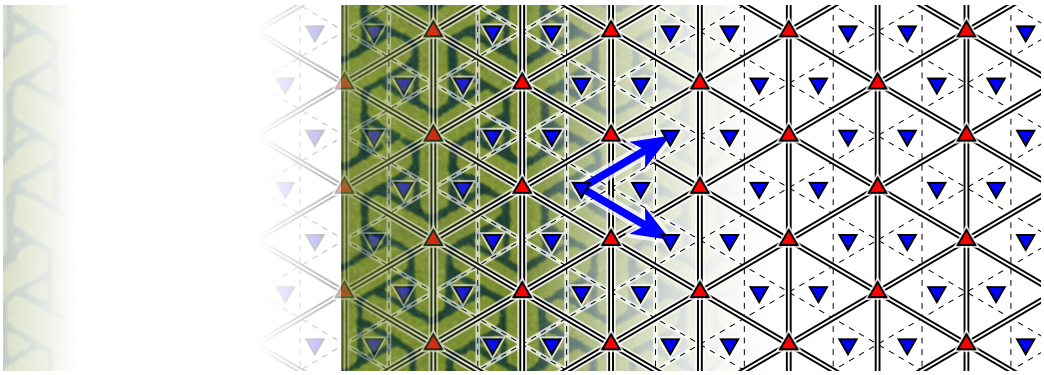
p3



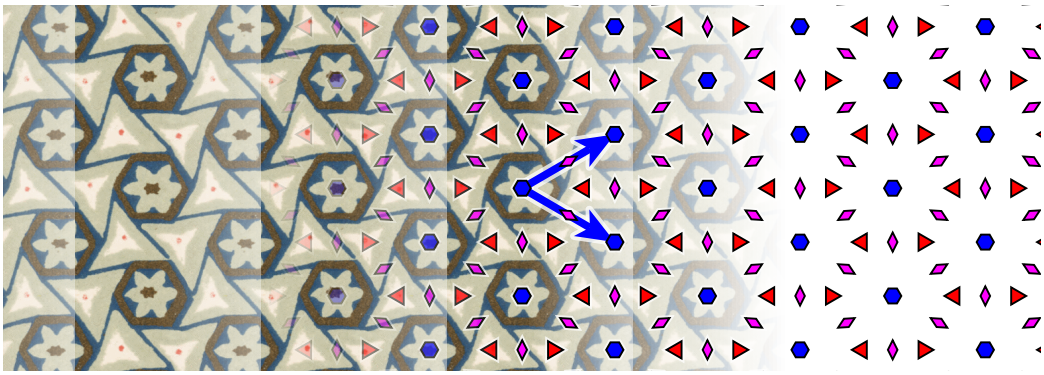
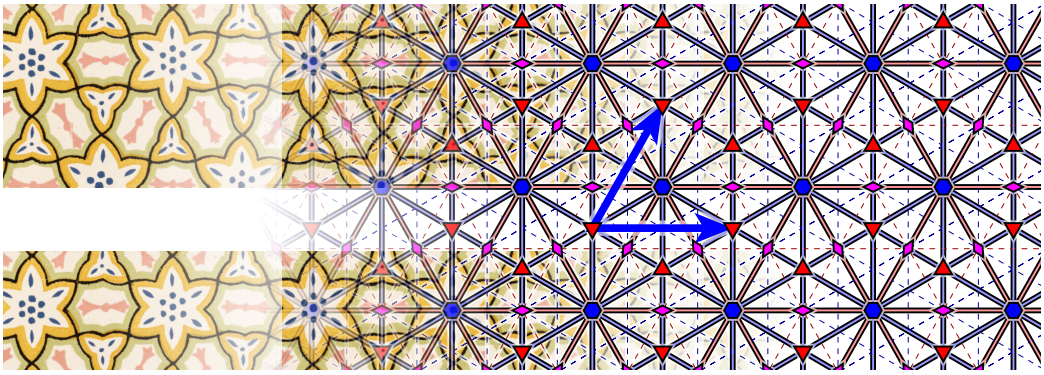
p3m1



p31m



p6



p6m