Selbstkorrigierende Kodierung im Ternärsystem

Einführung

Wir sind es uns bisher gewohnt, Informationen im Binär- oder in davon abgeleiteten Stellenwertsystemen (wie bspw. das Hexadezimalsystem) darzustellen. Dabei kann es manchmal durchaus sinnvoll sein, ein Zahlensystem zu wählen, dem eine ungerade Basis zugrunde liegt!

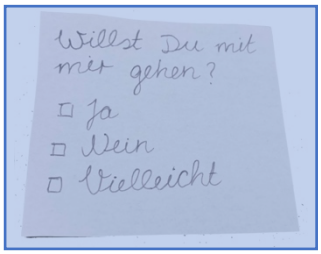
Ein Beispiel für solch ein Stellensystem ist das «**Ternärsystem**» (auch Trinär-, Dreier- oder 3-adisches System genannt), das die **Basis 3** verwendet. D.h. eine einzelne, ternäre Ziffer codiert **drei verschiedene Zustände**.

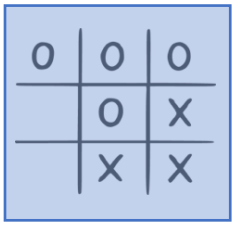
In dieser Unterrichtseinheit wird es darum gehen, wie man geschickt selbstkorrigierenden Code für das Ternärsystem gestalten kann. Bevor wir uns aber hierzu genauer Gedanken machen, werden wir zunächst das Ternärsystem selbst etwas näher kennenlernen.

Aufgabe

Betrachten Sie die nachfolgenden Bilder. Für welche Beispiele erachten Sie den Einsatz des Ternärsystems als geeignet?

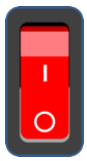
Ternäre Logik





Könnte man ggf. gelten lassen – ein Zustand wird einfach durch 2 ternäre Ziffern charakterisiert!

x, y, z Bewegung des 3D-Druckkopfs



Überlegen Sie sich ein eigenes Beispiel, für welches man Information sinnvollerweise ternär darstellen könnte:

Mögliche Nennungen: Zuordnung von Passagieren zu einer Flugklasse (Economy, Business, First), Aggregatszustände, Vier-Gewinnt-Spiel, 3G (geimpft, genesen oder getestet – wobei inzwischen das eine das andere nicht mehr ausschliessen muss...), usw.

Die Funktionsweise des Ternärsystems

Bei den Dezimalzahlen haben wir zehn Ziffern (`0` bis `9`) zur Verfügung, bei den Binärzahlen nur deren zwei (`0` und `1`). Beim Ternärsystem, das auf der Basis 3 beruht, verwenden wir **drei Ziffern**, üblicherweise:

`0, 1, 2`

Für gewisse Anwendungen bevorzugt man die Ziffern -1, 0, 1, die wir aber hier nicht weiter benutzen werden. Eine einzelne, ternäre Ziffer wird – ganz analog zum Bit – ein «**Trit**» genannt, und sechs Trits ergeben – analog zum Byte – ein «**Tryte**».

Um zu verstehen, wie das Ternärsystem genau funktioniert, schauen wir uns am besten ein Beispiel an. Wir werden feststellen, dass es sich um ein ganz gewöhnliches Stellenwertsystem handelt und deshalb völlig analog zum Dezimal- bzw. zum Binärsystem aufgebaut ist – nur, dass wir hier mit 3er-Potenzen statt mit 10-er bzw. 2er Potenzen arbeiten.

Beispiel

① Betrachten wir die ternäre Zahl **1210** und schreiben (von rechts nach links in aufsteigender Reihenfolge) die Potenzen der Basis (d.h. 3^0, 3^1, 3^2, 3^3, ...) daneben. Dann ergänzen wir die Faktoren 0 bis 2 vor den Dreierpotenzen – je nachdem, was im Ternärcode angegeben ist.

② Als nächstes rechnen wir alle Dreierpotenzen explizit aus.

③ Zu guter Letzt rechnen wir alles zusammen, unter Berücksichtigung der Faktoren. Wir erhalten als Ergebnis die Zahl in Dezimaldarstellung – in diesem Beispiel wäre das **48**.

①	1210_3 =	1 * 3^3 +	2 * 3^2 +	1 * 3^1 +	0 * 3^0
②		1 * 27 +	2 * 9 +	1 * 3 +	0 * 1
③	48_{10} =	27 +	18 +	3 +	0

Möchte man umgekehrt eine Dezimalzahl in eine Ternärzahl umwandeln, so muss man den dezimalen Wert mit Scheinen à Dreierpotenzen (1, 3, 9, 81 etc.) «bezahlen» und dabei möglichst wenig Scheine verwenden. Algorithmisch kann man das so bewerkstelligen: Man teilt die Dezimalzahl durch die Basis 3 und notiert sich jeweils den Rest. Man fährt damit so lange fort, bis der Quotient 0 ergibt, dann stoppt man. Die Reste ergeben – rückwärts gelesen – die ternäre Darstellung der Zahl.

Beispiele

Dezimalzahl:	48_{10}	*Dezimalzahl:*	121_{10}
48 / 3 = 16	Rest: 0	121 / 3 = 40	Rest: 1
16 / 3 = 5	Rest: 1	40 / 3 = 13	Rest: 1
5 / 3 = 1	Rest: 2	13 / 3 = 4	Rest: 1
1 / 3 = 0	Rest: 1	4 / 3 = 1	Rest: 1
		1 / 3 = 0	Rest: 1
Ternärzahl:	1210_3	*Ternärzahl:*	11111_3

Ein paar kleine Übungsaufgaben

1) Für welche Dezimalzahlen stehen die folgenden ternären Codes?

b) 20_3

$0 \cdot 3^0 + 2 \cdot 3^1 = 6_{10}$

c) 1120_3

$0 \cdot 3^0 + 2 \cdot 3^1 + 1 \cdot 3^2 + 1 \cdot 3^3 = 42_{10}$

d) 10110_3

$0 \cdot 3^0 + 1 \cdot 3^1 + 1 \cdot 3^2 + 0 \cdot 3^3 + 1 \cdot 3^4 = 93_{10}$

e) 212121_3

$1 \cdot 3^0 + 2 \cdot 3^1 + 1 \cdot 3^2 + 2 \cdot 3^3 + 1 \cdot 3^4 + 2 \cdot 3^5 = 637_{10}$

2) Rechnen Sie die folgenden Dezimalzahlen in Ternärzahlen um:

a) 16_{10}

$16/3 = 5 \quad R: 1$
$5/3 = 1 \quad R: 2$
$1/3 = 0 \quad R: 1$

121_3

b) 98_{10}

$98/3 = 32 \quad R: 2$
$32/3 = 10 \quad R: 2$
$10/3 = 3 \quad R:$
$3/3 = 1 \quad R: 0$
$1/3 = 0 \quad R: 1$

10122_3

c) 111_{10}

$111/3 = 37 \quad R: 0$
$37/3 = 12 \quad R: 1$
$12/3 = 4 \quad R: 0$
$4/3 = 1 \quad R: 1$
$1/3 = 0 \quad R: 1$

11010_3

d) 2021_{10}

$2021/3 = 673 \quad R: 2$
$6573/3 = 224 \quad R: 1$
$224/3 = 74 \quad R: 2$
$74/3 = 24 \quad R: 2$
$24/3 = 8 \quad R: 0$
$8/3 = 2 \quad R: 2$
$2/3 = 0 \quad R: 2$

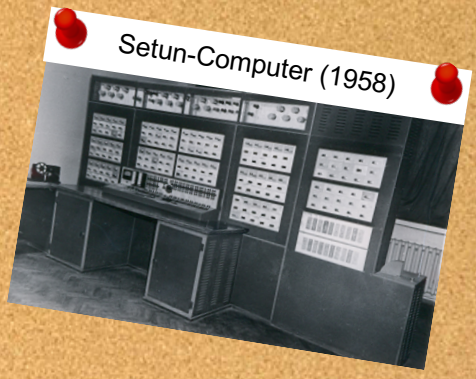
2202212_3

Kurzlösungen
1a) 6 b) 42 c) 93 d) 637 2a) 121 b) 10122 c) 11010 d) 2202212

Funfacts zum Ternärsystem

- Das ternäre Zahlensystem weist die höchste «Informationsdichte» unter den ganzzahlig basierten Systemen auf. Das bedeutet, dass man im Dreiersystem Zahlen am effizientesten (= platzsparendsten) speichern kann, weil man damit das bestmögliche Verhältnis zwischen Anzahl Stellen und Anzahl verschiedener Ziffern pro Stelle erreicht.
  Unter uns LP: Noch effizienter wäre die Euler'sche Zahl als Basis ☺ Aber die ist natürlich nicht ganzzahlig...

- Tatsächlich wurde in den 1960er-Jahren in der Sowjetunion ein Computer gebaut, der auf dem ternären System (mit den Ziffern -1, 0, 1) basierte: der Setun-Computer. Dieser wurde für Lehr- und wissenschaftliche Zwecke eingesetzt. Allerdings war bereits anfangs der 1970er-Jahre die binäre Technologie so weit fortgeschritten, dass man das ternäre System effizient auf binären Rechnern emulieren konnte. Die Entwicklung und Produktion ternärer Rechner wurde deshalb eingestellt.



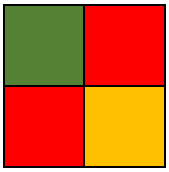
Bildquelle: Earl T. Campbell: «The Setun Computer»
https://earltcampbell.com/2014/12/29/the-setun-computer/ (abgerufen am 18.1.22)

Selbstkorrigierender Code im Ternärsystem

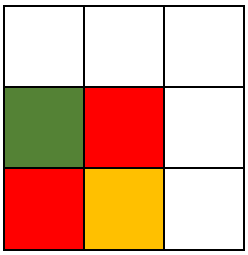
Wir haben verstanden, wie das Ternärsystem aufgebaut ist. Jetzt geht es darum, herauszufinden. mit welchen Kniffen man ternären Code erstellen kann, bei dem sich Fehler, die z.B. bei der Übertragung passieren, eruieren und ggf. korrigieren lassen.

Lernaufgabe 1: Ein Kartentrick mit drei Farben

Zwei Magier führen einen Kartentrick vor. Um den Zaubertrick vorzuführen, benötigen sie Karten mit drei verschiedenen Farben: grün, gelb und rot. Ein Magier verlässt den Raum und sein Kollege fordert einen Freiwilligen dazu auf, vier beliebige Karten auszuwählen und damit ein 2x2-Quadrat zu bilden. Der verbleibende Magier fügt anschliessend 5 weitere Karten hinzu, um das 2x2-Quadrat zu einem 3x3-Quadrat zu ergänzen.



2x2-Quadrat des Freiwilligen



Der Magier ergänzt die 5 leeren Felder

Der Freiwillige wird dann aufgefordert, eine der neun Karten durch eine Karte anderer Farbe zu ersetzen. Der Magier, der den Raum zuvor verlassen hat, wird wieder zurückgerufen. Er wird jetzt in jedem Fall genau bestimmen können, welche Karte verändert wurde und welche Farbe die ursprüngliche Karte hatte.

a) Überlegen Sie sich eine Strategie, wie der im Raum verbliebene Magier die 5 Karten legen muss, so dass sein Komplize in jedem Fall die vertauschte Karte finden und korrigieren kann.

b) Der Zaubertrick kann auch durchgeführt werden, wenn der Freiwillige ein 5x5-Quadrat legt, dass anschliessend vom Magier zu einem 6x6-Quadrat ergänzt wird. Wie muss der Magier im Raum in diesem Fall die Karten legen, damit sein Komplize in jedem Fall den Fehler findet?

c) Der Trick ist so aufgebaut, dass der Magier in jedem Fall **einen** Fehler erkennen **und** korrigieren kann. Vielleicht ist aber noch mehr möglich! Finden Sie die jeweils grösste Anzahl an Karten, die verändert werden können, damit der Magier...

   i) ... erkennen kann, dass so viele Karten vertauscht wurden.

   ii) ... zusätzlich erkennen kann, wo die vertauschen Karten liegen und welche Farben sie hatten.

d) Beweisen Sie, dass der Magier immer eine passende Karte in der Ecke des 3x3- bzw. 6x6-Quadrats anfügen kann, ohne die in a) und b) definierten Regeln zu verletzen.

e) Die Regeln in a) und b) gelten nur für Rechtecke mit speziellen Seitenlängen. Können Sie eine Regel finden, wie der Magier die Karten für ein beliebiges Quadrat mit Seitenlängen a und b legen muss, damit sein Komplize den Fehler erkennt?

Lösung Lernaufgabe 1

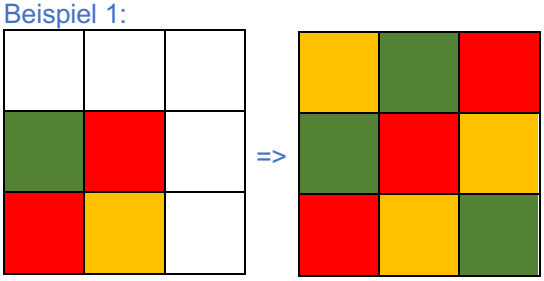
a) Wenn wir zwei Karten in einer Reihe oder Spalte betrachten, dann haben sie entweder die gleiche Farbe oder unterschiedliche Farben. Je nachdem, welcher Fall eintritt, muss der Magier eine andere Karte platzieren.

1. Fall: Falls die beiden Karten die gleiche Farbe haben, dann fügt der Magier eine Karte von der gleichen Farbe hinzu.

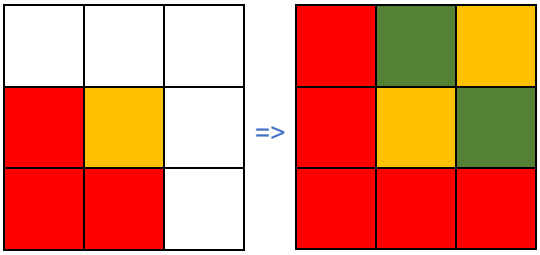
2. Fall: Falls die beiden Karten unterschiedliche Farben haben, dann fügt der Magier eine Karte von der dritten Farbe hinzu.

Das führt dazu, dass nach dem Hinzufügen der 5 Karten in jeder Zeile und jeder Spalte entweder drei Karten der gleichen Farbe oder drei Karten von unterschiedlicher Farbe liegen.

Beispiel 1:

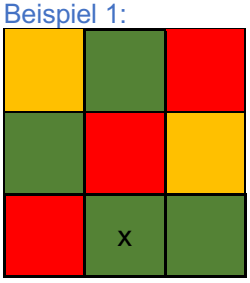


Beispiel 2:

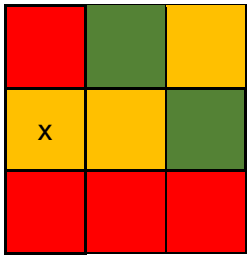


Wird nun eine der neun Karten durch eine Karte einer anderen Farbe ersetzt, so wird die Regel, dass alle Karten in jeder Reihe/Spalte die gleichen oder unterschiedlichen Farben haben, in genau einer Reihe und einer Spalte verletzt sein. Das Feld, das auf der Kreuzung dieser Zeile und dieser Spalte liegt, ist das Feld, auf dem die Karte vertauscht wurde.

Beispiel 1:



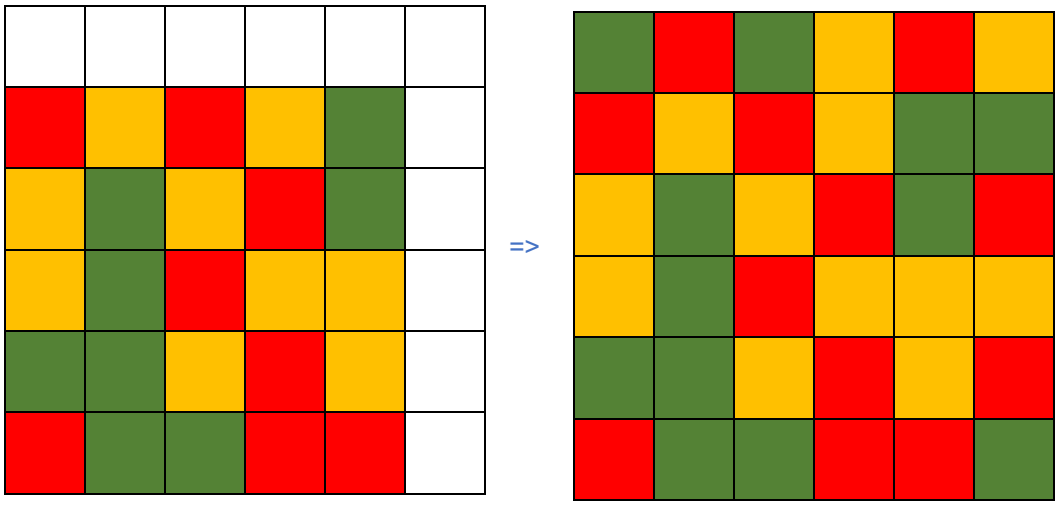
Beispiel 2:



Wir können auch direkt erkennen, was die ursprüngliche Farbe der Karte war. Wenn die beiden anderen Karten in der Zeile oder Spalte die gleiche Farbe haben, so war die falsche Karte ebenfalls von dieser Farbe. Ist die Farbe der beiden anderen Karten unterschiedlich, so muss die falsche Karte ursprünglich eine Karte der dritten Farbe gewesen sein.

In Beispiel 1 sind die anderen Karten in der zweiten Spalte und der dritten Zeile rot und grün, deshalb muss die ausgetauschte Karte gelb gewesen sein. In Beispiel 2 sind die beiden Karten in der ersten Spalte rot, also muss die ausgetauschte Karte auch rot gewesen sein.

b) Der Magier kann sehr ähnlich vorgehen wie beim 2x2-Quadrat. In jeder Reihe und jeder Spalte des 5x5-Quadrats muss es entweder drei Karten geben, die alle die gleiche Farbe haben, oder drei Karten, die alle verschiedene Farben haben. Diese drei Karten kann der Magier mental ignorieren und sich auf die zwei verbleibenden Karten konzentrieren. Er kann dann das Quadrat mit der gleichen Regel wie in a), angewendet auf die zwei verbleibenden Karten, ergänzen.

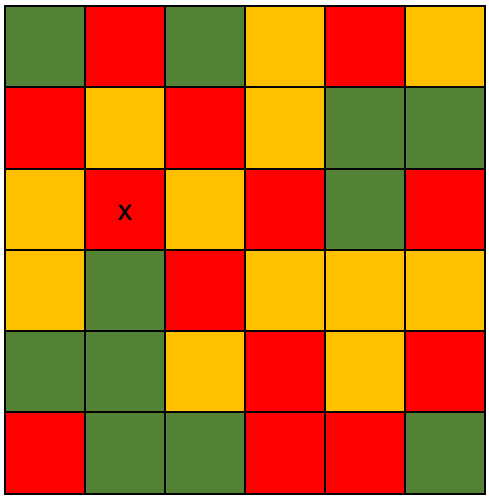


Wir erklären das Vorgehen anhand von drei Beispielen

- Oberste Zeile: Die drei letzten Karten haben unterschiedliche Farben. Wenn wir diese ignorieren, bleiben die beiden ersten übrig, die rot und gelb sind. Die zu ergänzende Karte muss also grün sein.
- Mittlere Zeile: In dieser Zeile liegen drei gelbe Karten, die anderen zwei sind rot und grün. Somit muss die zu ergänzende Karte gelb sein.
- Vierte Spalte von rechts: Die Spalte enthält drei rote Karten. Wenn wir diese wegdenken, bleiben zwei gelbe Karten übrig, die zu ergänzende Karte ist also auch gelb.

Wir bemerken, dass es für die sechs Karten in jeder Zeile oder Spalte nur vier mögliche Muster gibt:
Muster 1: Alle sechs Karten haben die gleiche Farbe.
Muster 2: Von jeder Farbe hat es genau zwei Karten.
Muster 3: Es hat von zwei Farben je drei Karten.
Muster 4: Von einer Farbe hat es vier Karten und von den anderen je eine.

Wenn nun eine Karte ausgetauscht wird, so wird es genau eine Zeile und genau eine Spalte geben, die nicht einem dieser Muster entsprechen. Die Karte auf der Kreuzung dieser Zeile und Spalte ist die vertauschte Karte.
Die Farbe der ursprünglichen Karte kann wie in a) auch eindeutig identifiziert werden, da es immer genau eine Möglichkeit gibt, die entsprechende Zeile und Spalte wieder auf eines der vier Muster zu bringen.
Wir illustrieren das Korrekturverfahren an einem Beispiel.



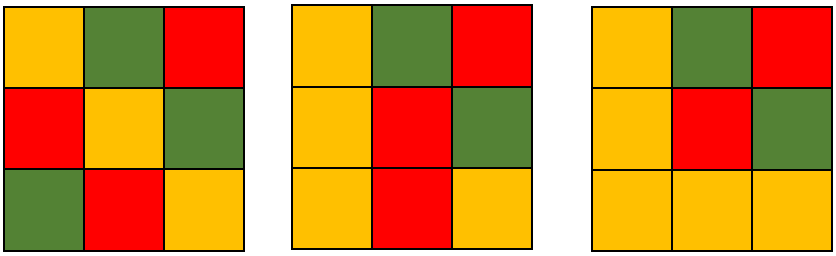
In der dritten Zeile hat es drei rote Karten, die anderen drei sind aber weder von der gleichen Farbe noch alle verschieden. In der zweiten Spalte hat es drei grüne Karten und auch hier sind die drei verbleibenden Karten nicht gleich oder verschieden. Die vertauschte Karte muss also an der mit x markierten Stelle liegen.

In der betroffenen Zeile hat es neben der falschen Karte je zwei rote, zwei gelbe und eine grüne Karte. Das impliziert, dass die falsche Karte ursprünglich grün war. In der betroffenen Spalte liegen neben der falschen Karte drei grüne Karten und je eine gelbe und eine rote. Auch hier ist grün die einzige Möglichkeit, wieder ein korrektes Muster herzustellen. Daraus können wir sicher folgern, dass die vertauschte Karte ursprünglich eine grüne Karte war.

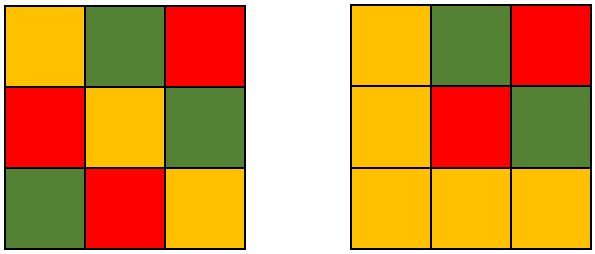
c) Es reicht aus, die Beweise für die 3x3-Quadrate zu führen, der Beweis für die 6x6-Quadrate (oder alle Quadrate, deren Seitenlänge durch 3 teilbar ist), geht genau gleich.

i) Wenn wir in einer Zeile oder einer Spalte eine Karte austauschen, dann wird immer ein Fehler erkannt. Auch wenn zwei Karten verändert werden, wird man das erkennen können, weil in den zwei betroffenen Spalten bzw. Zeilen Unstimmigkeiten auftreten werden. Allerdings lässt sich nicht mehr genau sagen, wie das ursprüngliche Quadrat ausgesehen haben muss (siehe auch Aufgabe 1c) ii)).

Werden drei Karten vertauscht, so müssen zwangsläufig zwei Karten in zwei verschiedenen Spalten oder zwei verschiedenen Zeilen verändert worden sein. Das generiert grundsätzlich zwei Unstimmigkeiten in zwei verschiedenen Spalten bzw. Zeilen, wovon aber höchstens eine durch das Verändern einer dritten Karte verborgen werden kann. Man bemerke: Bei drei ausgetauschten Karten (im unten gezeigten Beispiel die ersten zwei der mittleren Zeile und die erste der letzten Zeile) kann man zwar feststellen, dass das Quadrat manipuliert wurde, man kann aber nicht mit Sicherheit sagen, wie viele Karten tatsächlich vertauscht wurden. Beim Quadrat rechts bspw. sind die fehlerhafte, mittlere Spalte und die letzte Zeile durch das Verändern einer einzigen Karte behoben worden.

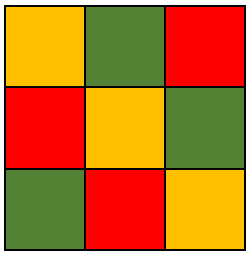


Bei vier Fehlern ist es möglich, Fehler vollständig zu verbergen, wie man anhand des folgenden Beispiels sieht:

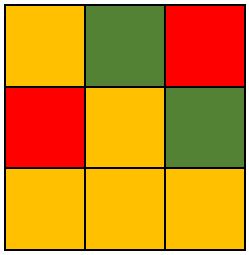


Beide dieser 2x2-Quadrate haben die gleichen Kontrollbits, sind aber sonst vollständig verschieden. Da die Quadrate aber alle Regeln erfüllen, wird der Fehler nicht erkannt. Es folgt daraus, dass das Vertauschen von vier Karten im Allgemeinen unerkannt bleibt.

ii) Wenn die zwei vertauschten Karten in der gleichen Zeile oder Spalte liegen, so kann ein Fehler zwar erkannt werden, jedoch ist es nicht in immer möglich, die falschen Karten eindeutig zu identifizieren.

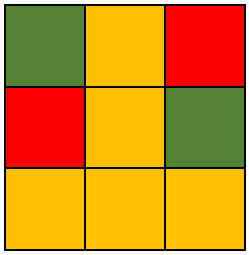


Feld vor Vertauschen

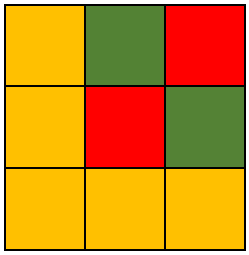


Feld nach Vertauschen

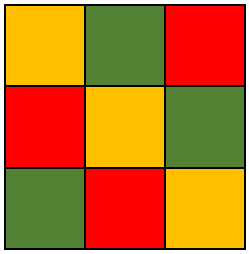
Statt den zwei ersten Karten in der untersten Reihe könnten wir auch die ersten zwei Karten in der obersten Reihe austauschen, um wieder ein korrektes 3x3-Quadrat zu erhalten. Aus diesem Gegenbeispiel folgt bereits, dass diese Codierung nicht 2-Fehlerkorrigierend ist. Nachfolgend alle Korrekturvarianten:



Feld nach falscher Korrektur (Variante 1)



Feld nach falscher Korrektur (Variante 2)

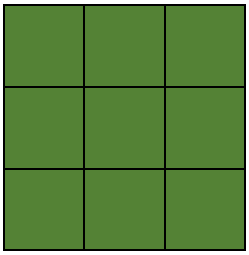


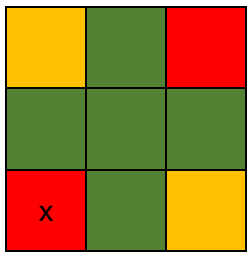
Feld nach richtiger Korrektur (Variante 3)

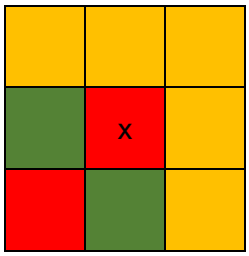
d) Wir zeigen, dass wir beim Ändern von einer Karte im ursprünglichen 2x2-Quadrat immer drei der Kontrolltrits anpassen können, damit wieder ein korrektes 3x3-Quadrat entsteht. Die drei Kontrolltrits sind die zwei Trits, die der Zeile und Spalte der veränderten Karte entsprechen, sowie das Kontrolltrit in der Ecke.

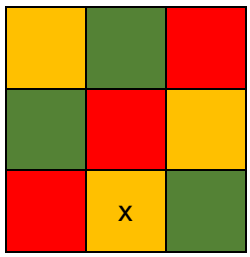
Damit unser Beweis vollständig ist brauchen wir zuerst einen Startzustand, bei dem die Karte in der Ecke korrekt gelegt werden kann. Die einfachste Möglichkeit ist der Zustand, bei dem alle Karten die gleiche Farbe haben. Die Karte in der Ecke muss dann auch die gleiche Farbe haben.

Ausgehend von diesem Startzustand können wir jedes beliebige 2x2-Quadrat herstellen und stellen fest, dass die Karte in der Ecke immer passend gewählt werden kann. Als Beispiel stellen wir das 2x2-Quadrat aus Beispiel 1 in der Lösung von a) her.









e) Um eine universell anwendbare Codierung für beliebige Rechtecke zu erhalten, müssen wir leider einen kleinen Teil der Schönheit dieses Problems aufgeben. Bis jetzt war es unwichtig, welche Farbe die einzelnen Karten haben, es zählt nur, dass jeweils drei Karten zusammen entweder gleichfarbig oder verschieden sind. Dieser Ansatz funktioniert aber nur, wenn die Seitenlänge des Rechtecks durch 3 teilbar ist. Für allgemeine Rechtecke brauchen wir einen anderen Ansatz.

Wir ordnen jeder Zahl eine Nummer zwischen 0 und 2 zu. Es ist nicht wichtig, wie wir diese Zahlen zuordnen, solange beide Magier die gleiche Zuordnung verwenden. Wir verwenden die Zuordnung 0 = grün, 1 = gelb und 2 = rot.

Mit dieser Zuordnung wählen wir nun die Kontrolltrits, so dass die Summe aller Zahlen in jeder Zeile und jeder Spalte ein Vielfaches von 3 ergibt. Als Beispiel betrachten wir eine Erweiterung eines 3x4- auf ein 4x5-Rechteck.

2	1	2	1	
0	0	2	1	
1	2	1	0	

0	0	1	1	1
2	1	2	1	0
0	0	2	1	0
1	2	1	0	2

Der zweite Magier muss jetzt nur Zählen, in welchen Reihen und welcher Spalte die Summe der Zahlen nicht durch 3 teilbar ist, um die ausgetauschte Karte zu identifizieren. Indem er die Summe der anderen Zahlen in der entsprechenden Zeile addiert und auf das nächste Vielfache von 3 ergänzt, kann er auch die ursprüngliche Farbe finden.

Lernaufgabe 2: Effiziente 1-korrigierende Codes für Trinärnachrichten

Für ein Binärwort der Länge 4 haben kennen wir bereits eine effiziente Methode, durch Hinzufügen von 4 Prüfbits den Code 1-Fehlerkorrigierend zu machen. Diese Methode kann mit einer Hamming-Matrix dargestellt werden.

c1	c2	
a1	a2	c3
a3	a4	c4

Tabellendarstellung

	a1	a2	a3	a4	c1	c2	c3	c4
c1	x		x		x			
c2		x		x		x		
c3	x	x					x	
c4			x	x				x

Hamming-Matrix

Die Spalten der Matrix beschreiben, wie sich die Prüfbits verändern, wenn eines der Nachrichtenbits verändert wird. Bei Trinärwörtern reicht es aber nicht aus, das Bit zu finden, das verändert wurde. Mit einem 1-Fehlerkorrigierenden Code wollen wir auch erkennen können, welchen Wert das fehlerhafte Bit ursprünglich hatte.

a) Beschreiben Sie eine Methode, mit der Sie anhand der veränderten Prüfbits erkennen können, welches Nachrichtenbit eines Trinärworts verändert wurden **und** welchen Wert es vor der Veränderung hatte.

b) Statt ein neues System für Trinärzahlen zu entwickeln könnte man auch alle Trinärzahlen in zweistellige Binärzahlen umwandeln und die bekannten Codierungsverfahren darauf anwenden. Begründen Sie, warum die in a) entwickelte Hamming-Matrix für Trinärcodes weniger Speicherplatz einnimmt als eine entsprechende Hamming-Matrix für zweistellige Binärzahlen.

c) Ist es möglich mit weniger als vier Prüfbits auszukommen? Begründen Sie Ihre Antwort und geben Sie gegebenenfalls die Hamming-Matrix an.

Lösung Lernaufgabe 2

a) Die Kontrollstruktur ist gleich wie bei einem 1-Korrigierenden Binärcode. Insbesondere werden die Prüfbits c_1, c_2, c_3 und c_4 durch diese Gleichungen festgelegt.

$c_1 = a_1 \oplus_3 a_3$ $c_2 = a_2 \oplus_3 a_4$ $c_3 = a_1 \oplus_3 a_2$ $c_4 = a_3 \oplus_3 a_4$

Da wir aber nicht mit Binärzahlen, sondern mit Trinärzahlen rechnen, können die Nachrichtenbits und die Kontrollbits drei Werte annehmen (0, 1 und 2), somit muss das Zeichen $\oplus$ anders definiert werden. Dieses neue Zeichen bezeichnen wir mit $\oplus_3$ und definieren es über die folgenden Regeln:

$0 \oplus_3 0 = 0,$ $0 \oplus_3 1 = 2,$ $0 \oplus_3 2 = 1,$ $1 \oplus_3 1 = 1,$ $1 \oplus_3 2 = 0,$ $2 \oplus_3 2 = 2$

Allgemein ausgedrückt bedeutet dies, dass das Resultat der $\oplus_3$-Operation so gewählt werden muss, dass die Summe der drei Zahlen Modulo 3 genau 0 ergibt. Man beachte, dass diese Operation kommutativ ist, weswegen wir im Folgenden nicht alle Möglichkeiten für die Argumente auflisten werden. Mit obiger Definition beobachten wir nun, was passiert, wenn ein Nachrichtenbit verändert wird.

2	0			2	2	
1	1	1	=>	1	2	0
0	2	1		0	2	1

Das Nachrichtenbit a_2 wurde von 1 auf 2 geändert. Wenn wir jetzt für diese neue die Kontrollbits berechnen, dann würde $c_2 = 2$ und $c_3 = 0$ folgen. Wie erwartet verändern haben sich die beiden Kontrollbits, die a_2 überwachen, verändert. Insbesondere verändern sie sich um genau $-1 \; mod \; 3$. Wir folgern daraus eine erste Regel:

Regel 1: Unterscheiden sich zwei oder mehr Kontrollbits um $-1 \; mod \; 3$ von ihrem berechneten Wert, so wurde das Nachrichtenbit, dass von ihnen überwacht wird, um $+1 \; mod \; 3$ verändert.

2	0			2	1	
1	1	1	=>	1	0	2
0	2	1		0	2	1

Das Nachrichtenbit a_2 wurde von 1 auf 0 geändert. Auch diesmal hat das einen direkten Einfluss auf die Kontrollbits c_2 und c_3, jedoch verändern sie sich diesmal um $+1 \; mod \; 3$. Auch damit formulieren wir eine Regel:

Regel 2: Unterscheiden sich zwei oder mehr Kontrollbits um $+1 \; mod \; 3$ von ihrem berechneten Wert, so wurde das Nachrichtenbit, dass von ihnen überwacht wird, um $-1 \; mod \; 3$ verändert.

Mit diesen beiden Regeln kann somit immer ein Fehler in einem Trinärwort erkannt und direkt korrigiert werden.

Dieses Prinzip kann nun in eine neue Version der Hamming-Matrix übertragen werden. Es reicht nicht aus, nur zu vermerken, welche Stelle verändert wurde, mir müssen zusätzlich vermerken, welche Art der Änderung durchgeführt wurde. Wir können das lösen, indem wir die Zeilenanzahl der Matrix verdoppeln und zwei Zeichen verwenden.

In der Matrix bedeutet +, dass sich das Trit um $+1 \ mod \ 3$ verändert, das Zeichen – bedeutet, dass sich das Trit um $-1 \ mod \ 3$ verändert. Damit stellen wir diese Matrix auf.

	a_1+	a_2+	a_3+	a_4+	a_1-	a_2-	a_3-	a_4-	c_1+	c_2+	c_3+	c_4+	c_1-	c_2-	c_3-	c_4-
c_1	-		-		+		+		+				-			
c_2		-		-		+		+		+				-		
c_3	-	-			+	+					+				-	
c_4			-	-			+	+				+				-

Das ist natürlich wieder eine Hamming-Matrix, da keine zwei Zeilen identisch sind. Wir können aus dieser neuen Matrix nun ablesen, wie aus einer Veränderung der Trits der Fehler gefunden werden kann. Wenn z.B. die beiden Trits c_2 und c_3 um $+1 \ mod \ 3$ verändert wurden, so suchen wir die Spalte, in der bei c_2 und c_3 ein + steht. Das ist die Spalte a_2-, das bedeutet also, dass das Nachrichtentrit a_2 um $-1 \ mod \ 3$ verändert wurde.

Mit dieser Methode kann jede beliebige Hamming-Matrix für Binärnachrichten umgewandelt werden in eine entsprechende Hamming-Matrix für Trinärnachrichten.

b) Wir betrachten eine Tritcodierung mit 4 Nachrichtentrits und 4 Kontrolltrits. Wir können diese Nachricht umwandeln in eine Binärcodierung mit 8 Nachrichtenbits und 8 Kontrollbits. Wir berechnen nun den Speicherplatz der beiden Hamming-Matrizen.

Tritcodierung: Wir brauchen pro Nachrichtentrit 2 Spalten und pro Kontrolltrit 2 Spalten und 1 Zeile. Wie in a) gezeigt resultiert daraus eine Matrix der Dimension $4 \times (2 \times 4 + 2 \times 4) = 4 \times 16$. Die Matrix hat somit 64 Einträge.

Bitcodierung: Wir brauchen pro Nachrichtenbit 1 Spalte und pro Kontrollbit je eine 1 Zeile und 1 Spalte. Daraus resultiert eine Matrix der Dimension $8 \times (8 + 8) = 8 \times 16$. Diese Matrix hat 128 Einträge und braucht somit doppelt so viel Speicherplatz wie die Matrix der Trinärcodierung.

Wir können auch allgemein begründen, dass die Matrix der Tritcodierung gleich viele Spalten hat, jedoch nur halb so viele Zeilen, deshalb wird der Speicherplatz einer tritcodierten Nachricht immer halb so gross sein.

c) Es ist möglich mit nur drei Kontrollbits auszukommen. Die folgende Hamming-Matrix zeigt ein Beispiel.

	a_1+	a_2+	a_3+	a_4+	a_1-	a_2-	a_3-	a_4-	c_1+	c_2+	c_3+	c_1-	c_2-	c_3-
c_1	-			-	+			+	+			-		
c_2		-	-	-		+	+	+		+			-	
c_3	-	-	-	-	+	+	+	+			+			-

Lernaufgabe 3: Visualisierung der Abstände mit drei Zuständen (Distanzmatrix)

a) Zeichnen Sie einen möglichst übersichtlichen eindimensionalen Hyperwürfel mit drei Zuständen (0 , 1 , 2).
(D.h. listen Sie alle möglichen Folgen von (0 , 1 , 2) der Länge 1 auf und verbinden Sie alle, welche Abstand 1 haben mit einer Kante.)

b) Erweitern Sie Ihren eindimensionalen Hyperwürfel von Aufgabe 2a) zu einem möglichst übersichtlichen zweidimensionalen Hyperwürfel.
(D.h. listen Sie alle möglichen Folgen von (0 , 1 , 2) der Länge 2 auf und verbinden Sie alle, welche Abstand 1 haben mit einer Kante.)
Tipp: Ergänzen Sie die Folgen von a) mit jeweils 0 , 1 oder 2 . Sie können dazu Ihren eindimensionalen Hyperwürfel also zuerst dreimal zeichnen und dann ergänzen. So erhalten Sie alle möglichen Folgen von der Länge 2. Verbinden Sie nun alle Folgen, welche Abstand 1 haben.

c) Aus wie vielen Bitfolgen und Kanten besteht der dreidimensionale Hyperwürfel mit drei Zuständen?
Zusatz: Wie viele Kanten hat ein hat der n-dimensionale Hyperwürfel mit drei Zuständen?

d) *Knobelaufgabe: Wie könnte der dreidimensionale Hyperwürfel übersichtlich dargestellt werden?

Lösung Lernaufgabe 3

a) Alle möglichen Folgen von (0 , 1 , 2) der Länge 1 sind 0 , 1 und 2 . Mit nur einem Trit ist es schnell einsehbar, dass alle den Abstand 1 zueinander haben. So ist zwischen allen Trit eine Kante.

Abbildung 1: eindimensionaler Hyperwürfel

b) Um alle möglichen Folgen von (0 , 1 , 2) der Länge 2 zu bestimmen, verdreifachen wir die den eindimensionalen Hyperwürfel und hängen bei jeder Kopie eine 0 , 1 oder 2 vor die Tritfolgen. Auf diese Weise erhalten wir ungefähr das die rechtsstehende Abbildung 2.
Anschliessend verbinden wir die neuen Tritfolgen, die Abstand 1 haben. Da nur das erste Trit neu ist, müssen wir nur diese untersuchen und stellen fest, dass wir jede «kopierte» Stelle miteinander verbinden müssen. Dies ist in der Abbilung 3 festgehalten.

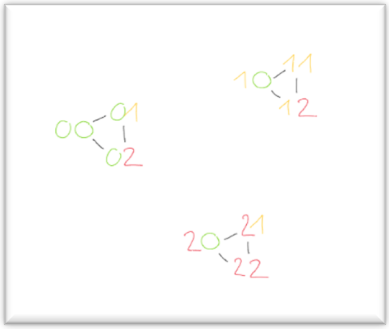


Abbildung 2: Verdreifachung des eindimensionalen Hyperwürfels

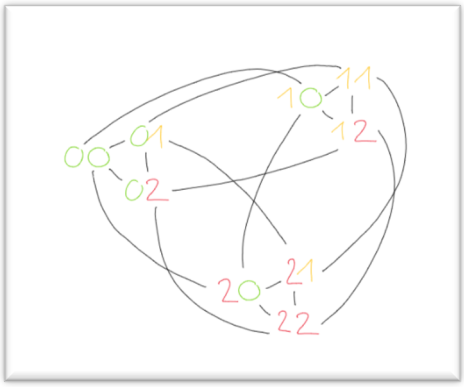


Abbildung 3: zweidimensionaler Hyperwürfel (ungeordnet)

Nun möchten wir die Tritfolgen etwas übersichtlicher anordnen. Inspiriert vom zweidimensionalen Koordinatensystem könnte dies aussehen wie in der Abbildung 4 dargestellt. (Jede Tritfolge xy steht am Punkt (x|y) des kartesischen zweidimensionalen Koordinatensystems.)

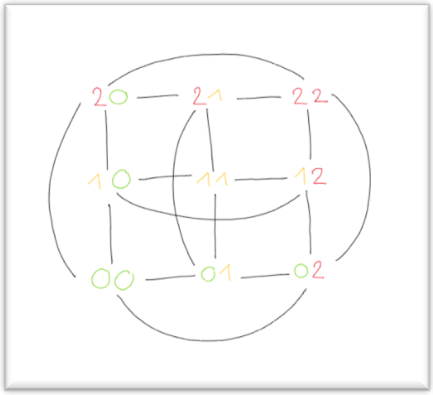


Abbildung 4: zweidimensionaler Hyperwürfel (geordnet)

c) Machen wir uns eine tabellarische Übersicht und ein paar Gedanken dazu, wie wir den dreidimensionalen Hyperwürfel aus dem zweidimensionalen Hyperwürfel konstruieren würden.

Dimension	1	2	3	n
Anzahl Tritfolgen	3	9		
Anzahl Kanten	3	18		

Dass der zweidimensionale Hyperwürfel 18 Kanten hat, könnten wir im Bild zählen. Wir können es uns aber auch folgendermassen überlegen. Jede Tritfolge ist durch 4 Kanten mit einer andern Tritfolge verbunden. So erhalten wir vorerst $9 \cdot 4 = 36$. Da aber jede Kante zu zwei Tritfolgen gehört und somit jede Kante doppelt gezählt würde, teilen wir unser vorläufiges Ergebnis noch durch 2 und erhalten $9 \cdot 4 : 2 = 18$.
Alternativ könnte man auch überlegen, wie viele Kanten neu dazukommen. Im eindimensionalem Hyperwürfel waren dies 3. Beim Konstruieren des zweidimensionalen Hyperwürfels haben wir den eindimensionalen verdreifacht. Die macht bereits $3 \cdot 3 = 9$ *Kanten. Dann wurde jede Tritfolge mit seinen zwei entsprechenden Kopien verbunden. Dies macht insgesamt (unter Berücksichtigung die Kanten nicht doppelt zu zählen) ebenfalls* $3 \cdot 3 + 9 \cdot 2 : 2 = 18$.

Dimension	1	2	3	n
Anzahl Tritfolgen	3	9	27	3^n
Anzahl Kanten	3	18		

Schnell sehen wir, dass durch die Verdreifachung für jede höhere Dimension wir im dreidimensionalen Hyperwürfel 27 Tritfolgen und im n-dimensionalen 3^n Tritfolgen haben.

Dimension	1	2	3	n
Anzahl Tritfolgen	3	9	27	3^n
Anzahl Kanten	3	18	81	

Entweder überlegen wir uns wieder, mit wie vielen Tritfolgen jede Tritfolge verunden ist (also Abstand 1 hat) oder wir überlegen, wie viele neue Kanten dazukommen.
Damit zwei Tritfolgen den Abstand 1 haben, haben sie genau ein unterschiedliches Trit. In einer Tritfolge der Länge drei, haben wir also drei Möglichkeiten jeweils ein Trit durch eines der zwei anderen zu ersetzen. Dies bedeutet, dass jede Tritfolge sechs Kanten (zu sechs Tritfolgen mit Abstand 1) hat. Unter der Berücksichtigung, dass wir keine Kante doppelt zählen, erhalten wir folglich $27 \cdot 6 : 2 = 81$ Kanten.
Die alternative Überlegung verläuft ähnlich wie im zweidimensionalen Fall.
Im zweidimensionalem Hyperwürfel waren dies 18 Kanten. Beim Konstruieren des dreidimensionalen Hyperwürfels haben wir den zweidimensionalen verdreifacht. Die macht bereits $18 \cdot 3 = 54$ *Kanten. Dann wurde jede Tritfolge mit seinen zwei entsprechenden Kopien verbunden. Dies macht insgesamt (unter Berücksichtigung die Kanten nicht doppelt zu zählen) ebenfalls* $18 \cdot 3 + 27 \cdot 2 : 2 = 81$.

Dimension	1	2	3	n
Anzahl Tritfolgen	3	9	27	3^n
Anzahl Kanten	3	18	81	$3^n \cdot n$

Auch für den allgemeinen Fall der Dimension n können wir dieselben Überlegungen anstellen. Eine Tritfolge der Länge n hat n Möglichkeiten, ein einzelnes Trit gegen zwei andere auszutauschen und zu diesen neuen Tritfolgen den Abstand 1 zu haben. Dies bedeutet, dass jede Tritfolge $2n$ Kanten (zu $2n$ Tritfolgen mit Abstand 1) hat. Unter der Berücksichtigung, dass wir keine Kante doppelt zählen, erhalten wir folglich $3^n \cdot 2n : 2 = 3^n \cdot n$ Kanten.

d) Mit 81 Kanten ist es sehr schwierig den dreidimensionalen Hyperwürfel übersichtlich darzustellen. Eine naheliegende Idee ist es, die Bitfolgen xyz wie im dreidimensionalen, kartesischen Koordinatensystem anzuordnen und die Kanten einzuzeichnen. In Abbildung 5 ist dies zu sehen, auch wenn auf die «sichtbaren» Tritfolgen und deren Kanten fokussiert wurde und nicht alle 81 Kanten eingezeichnet sind.

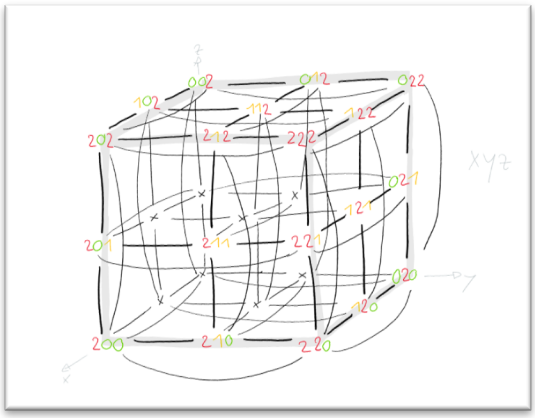


Abbildung 5: dreidimensionaler Hyperwürfel (im Koordinatensystem)

Eine andere Idee ist es, die Abbildung 4 zu verdreifachen und so zu verbinden, dass es die Form eines Torus (oder Donut) ähnelt (vgl. Abbildung 6). Hierbei kann man die Tritfolgen xyz auf teilweise gekrümmten Ebenen verstehen. Abbildung 7 versucht diese Ebenen im nicht-kartesischen Koordinatensystem aufzuzeigen. Grün bedeutet 0, gelb 1 und rot 2.

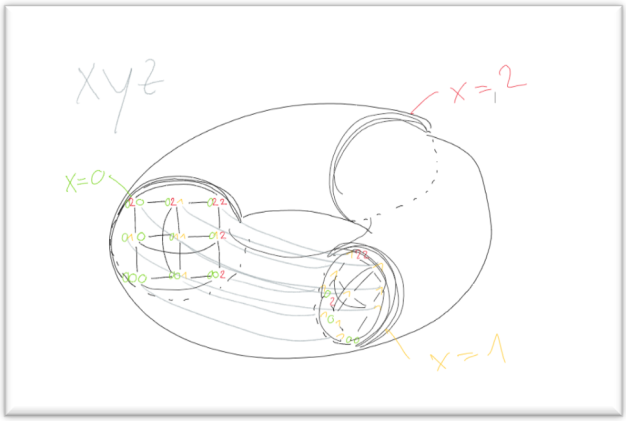


Abbildung 6: dreidimensionaler Hyperwürfel (Torus)

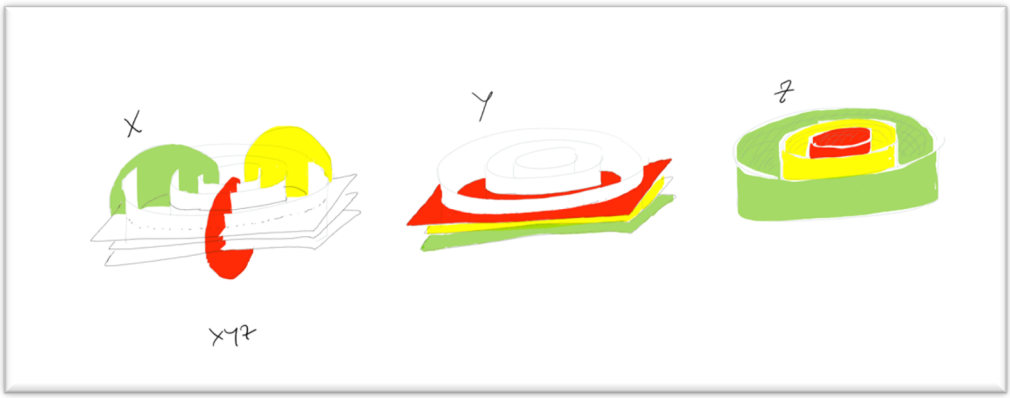


Abbildung 7: Ebenen des dreidimensionalen Hyperwürfels in Torusform