Hans Walser, [20080124a]

Hypercubus

1 Worum es geht

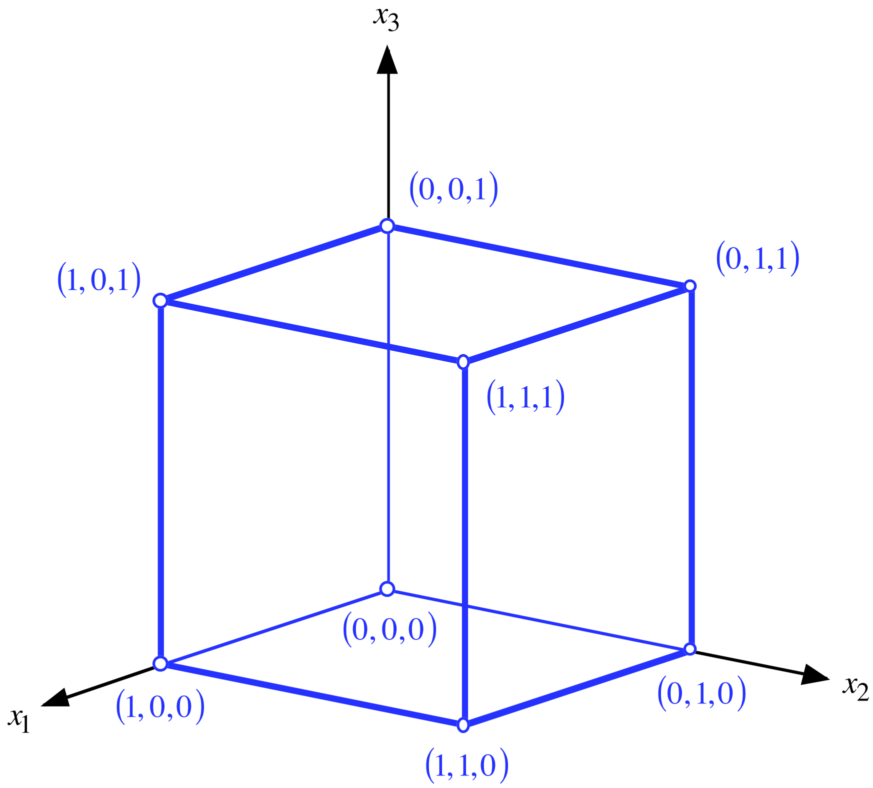
Es werden n-dimensionale Hypercubi oder Hyperwürfel in isometrischer Projektion vorgestellt.

2 Hintergrund

Wir schreiben die Zahlen 0 bis $2^n - 1$ in n-stelliger binärer Form. Für $n = 3$ zum Beispiel heißt das:

Dezimal	Binär
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

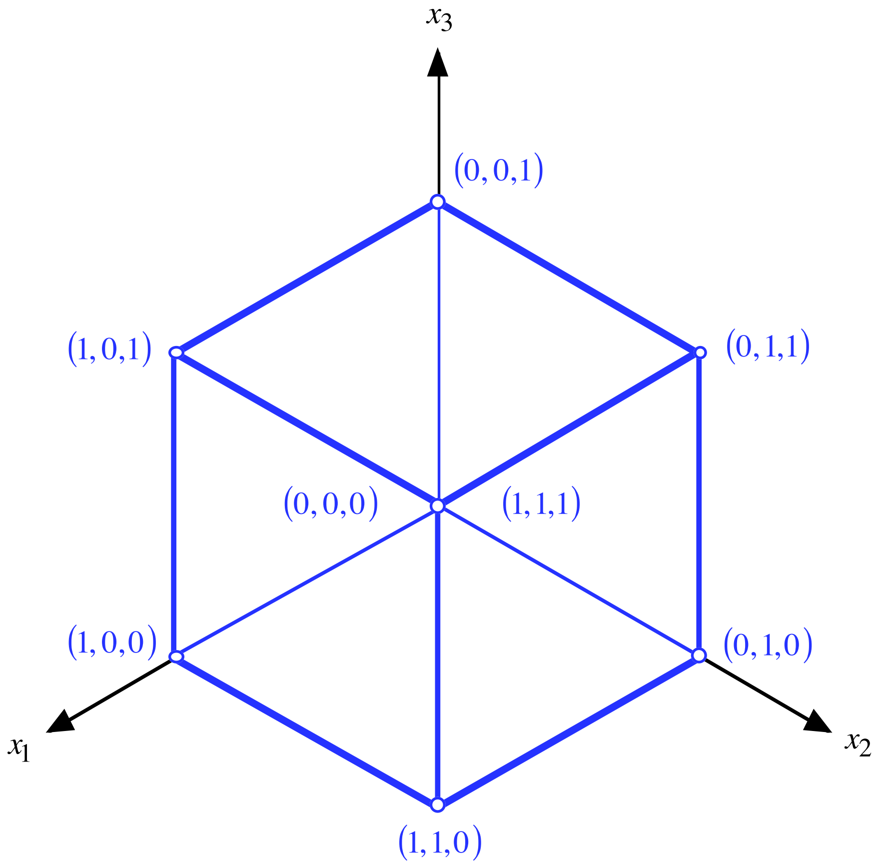
Die Ziffern der binären Form interpretieren wir als kartesische Koordinaten im n-dimensionalen Raum. Für $n = 3$ ergeben sich die Eckpunkt des dreidimensionalen Würfels.



Würfel

Zwei Punkte, deren Koordinaten sich in genau einer Stelle unterscheiden (Hamming-Dinstanz 1) sind durch eine Kante verbunden. Zwei Punkte, deren Koordinaten sich in genau zwei Stellen unterscheiden (Hamming-Distanz 2), können durch eine Seitenflächendiagonale verbunden werden, und zwei Punkte, deren Koordinaten sich in allen drei Stellen unterscheiden (Hamming-Distanz 3), sind diametral und können durch eine Raumdiagonale verbunden werden.

Eine besonders symmetrische Darstellung erhalten wir bei Normalprojektion längs einer Raumdiagonalen.



Isometrische Darstellung

Hier sind alle Kanten gleichermaßen verkürzt, wir sprechen daher von einer *isometrischen* Projektion. Die Winkel zwischen den Kantenprojektionen sind Vielfache von $\frac{\pi}{n}$.

Andere Strecken, wie zum Beispiel die Seitenflächendiagonalen oder die Körperdiagonalen, werden allerdings unterschiedlich verkürzt.

3 Liste

Im Folgenden einige n-dimensionale Hyperwürfel in isometrischer Projektion dargestellt. Zunächst werden nur die Eckpunkte gezeichnet, anschließend nur die Kanten (Hamming-Distanz 1), dann nur die Seitenflächendiagonalen (Hamming-Distanz 2) und so weiter bis zu den Verbindungen diametraler Punkte (Hamming-Distanz n).

3.0 Punkt

Am Anfang war der Punkt

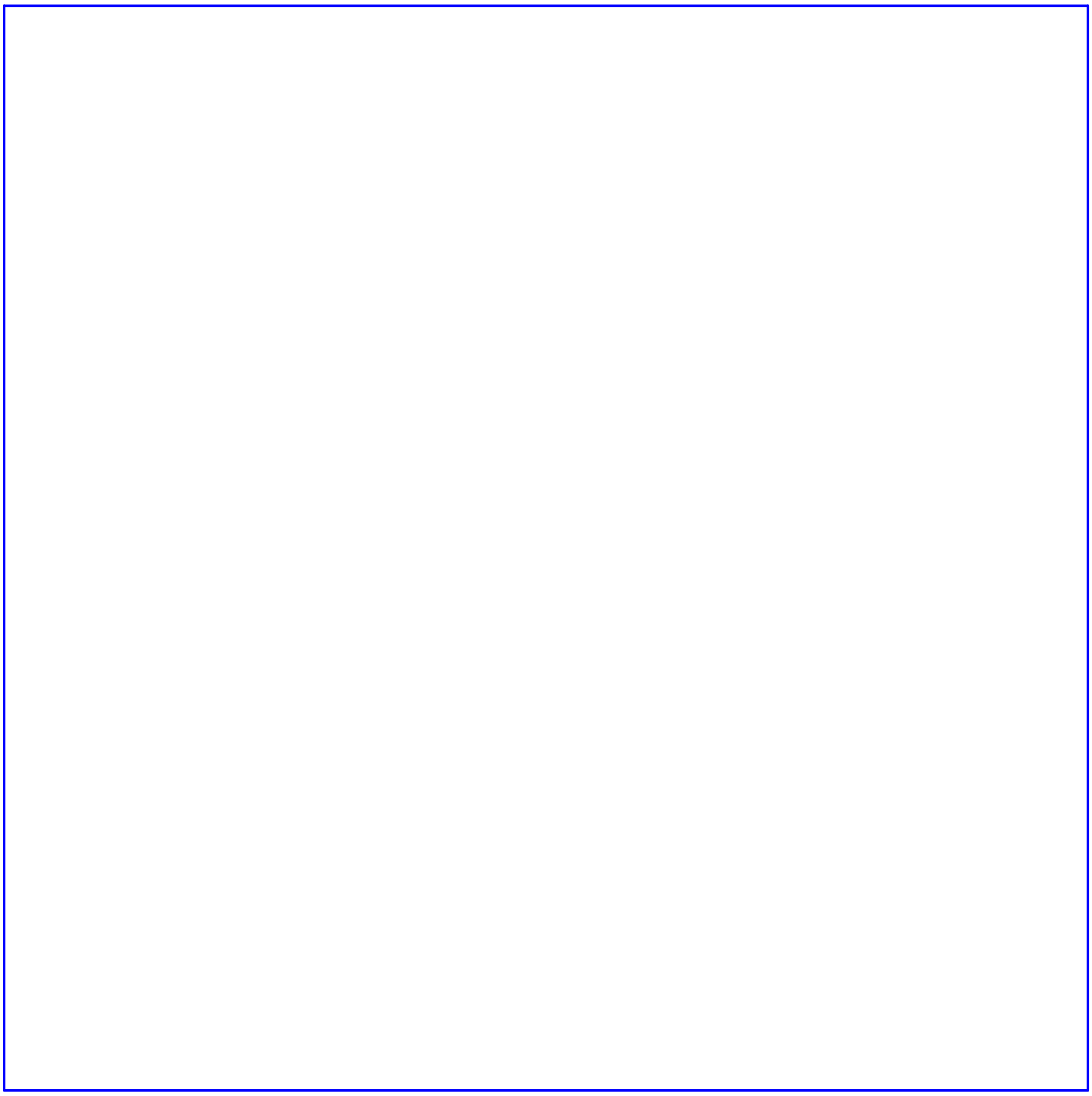
3.1 Strecke

Endpunkte einer Strecke

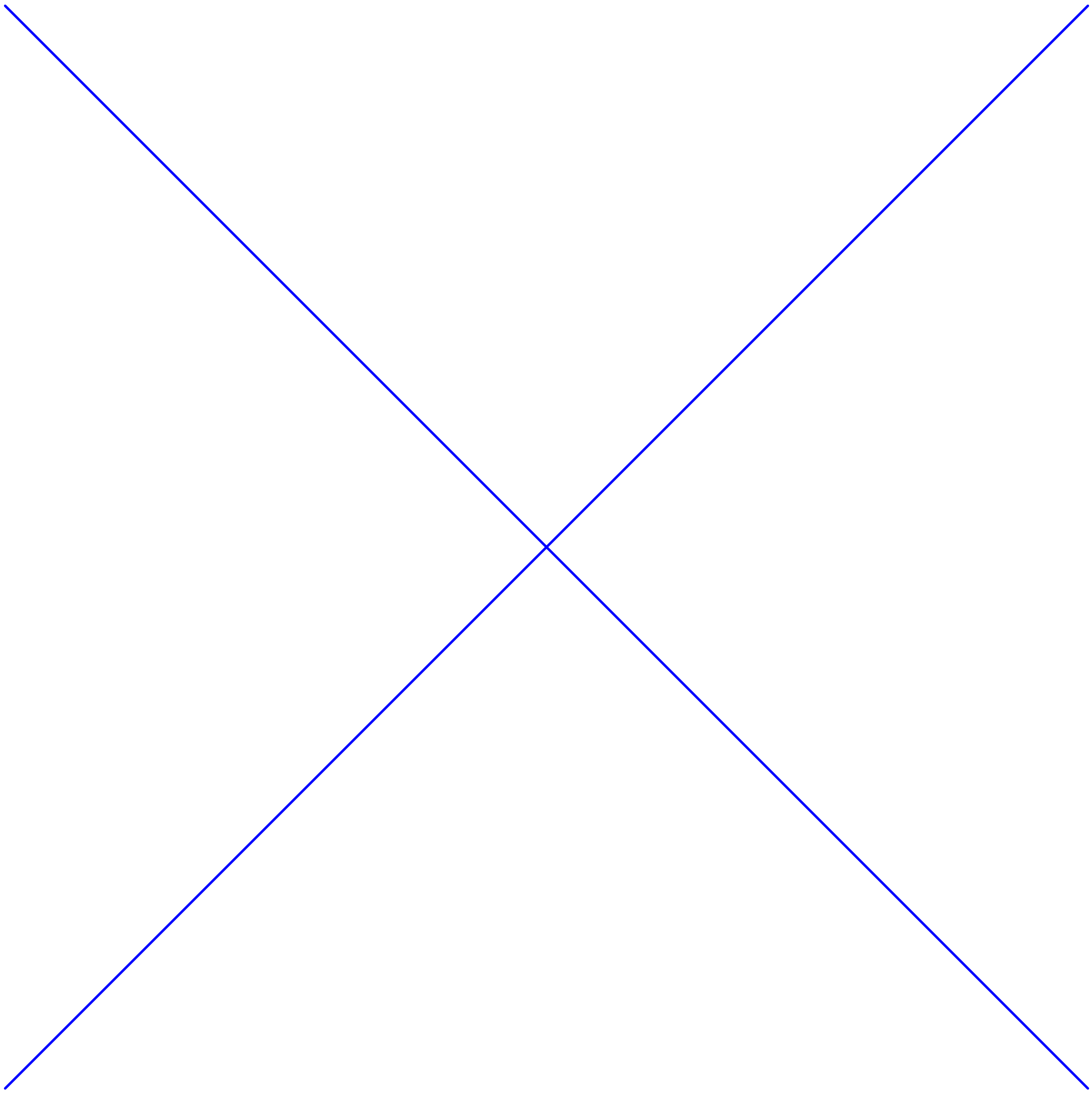
Strecke

3.2 Quadrat

Eckpunkte des Quadrates

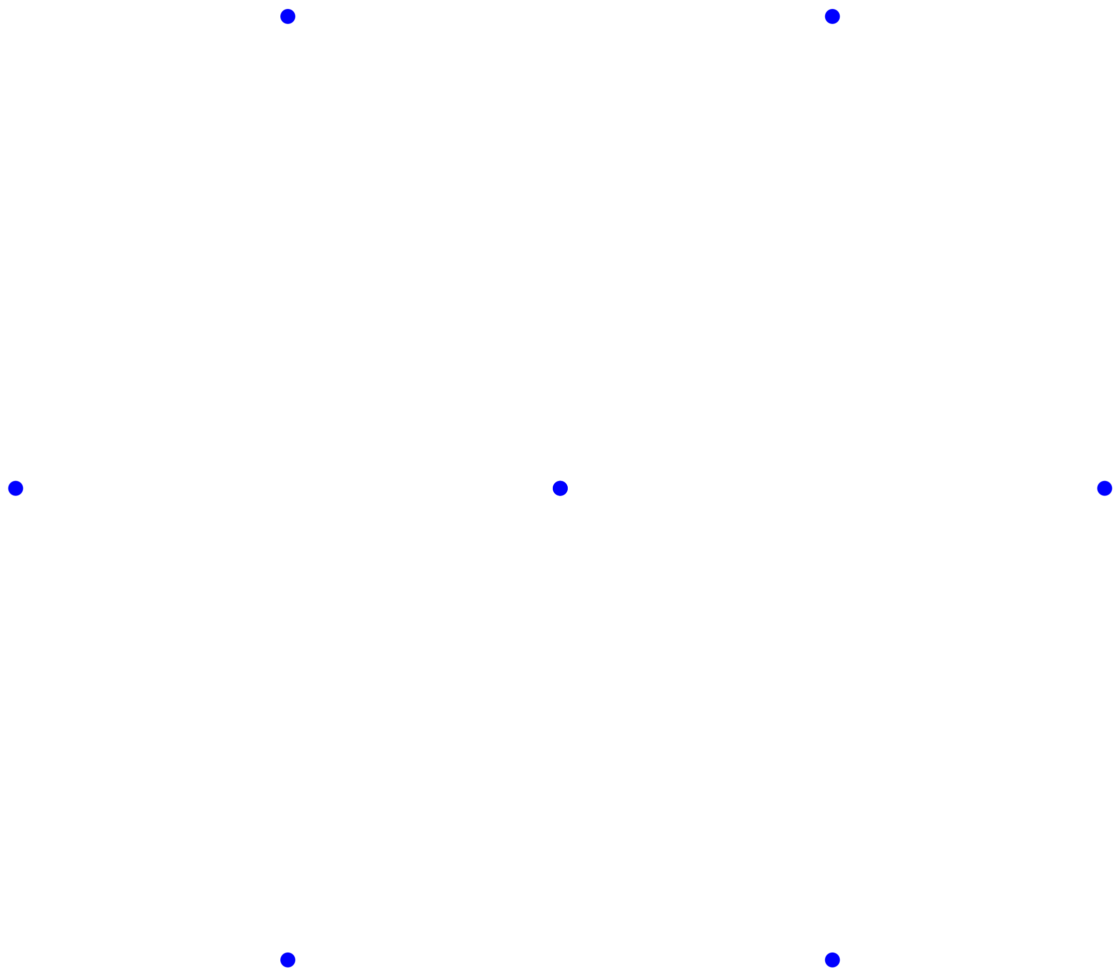


Quadrat

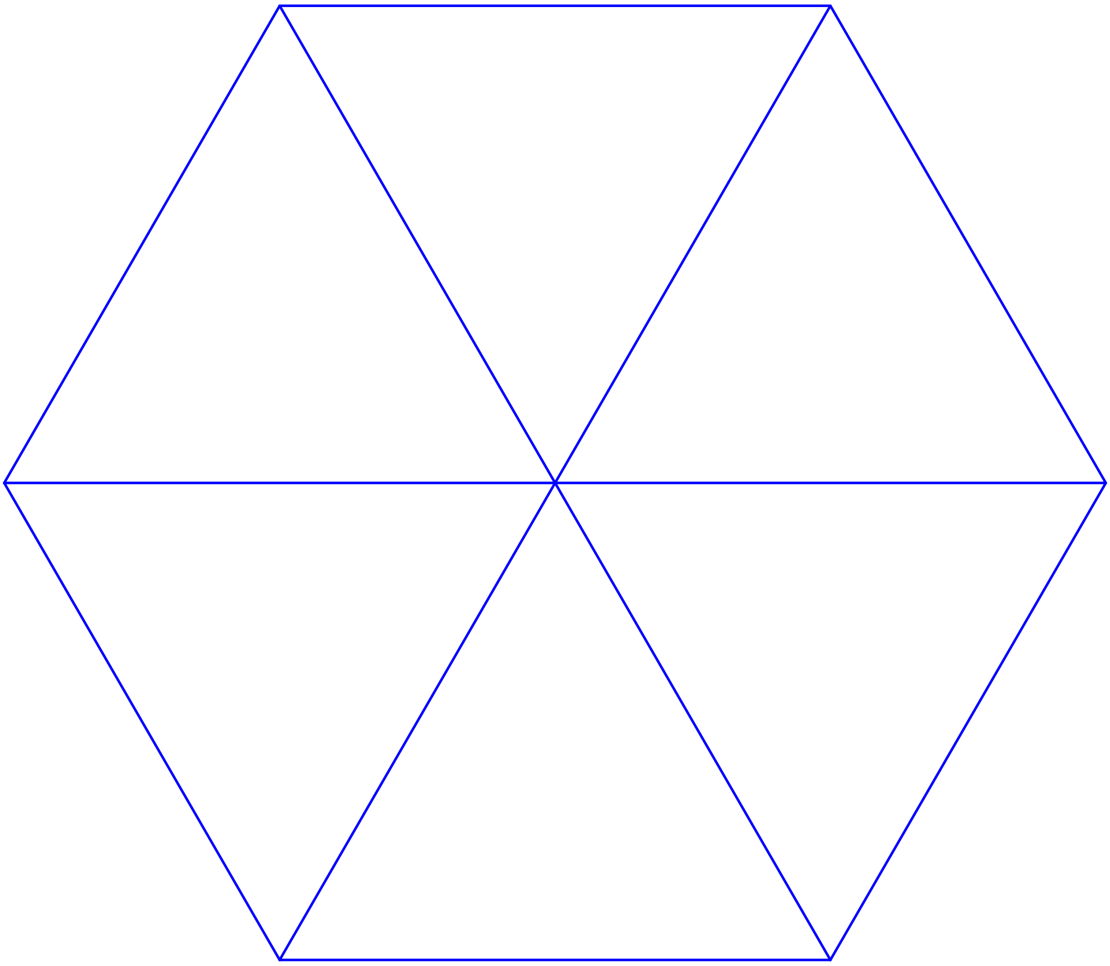


Quadratdiagonalen

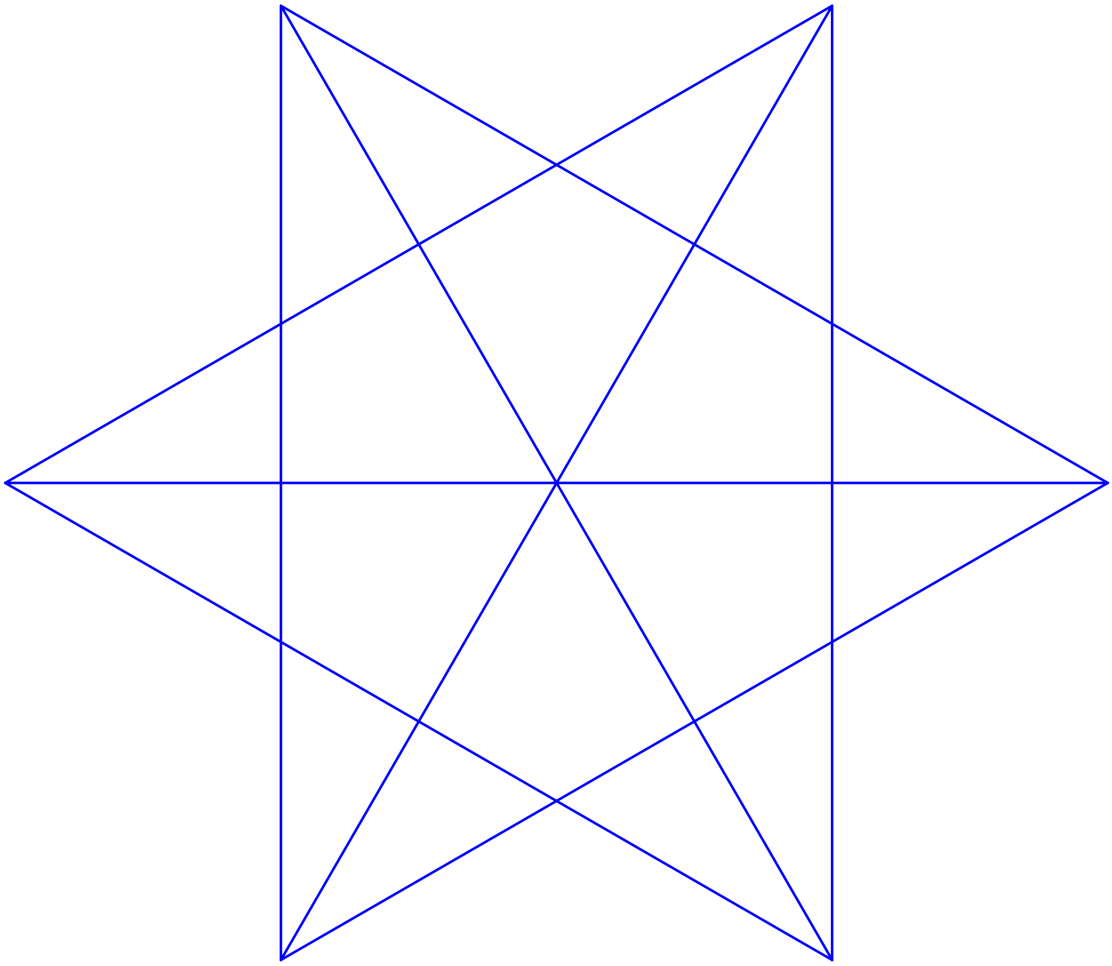
3.3 Würfel



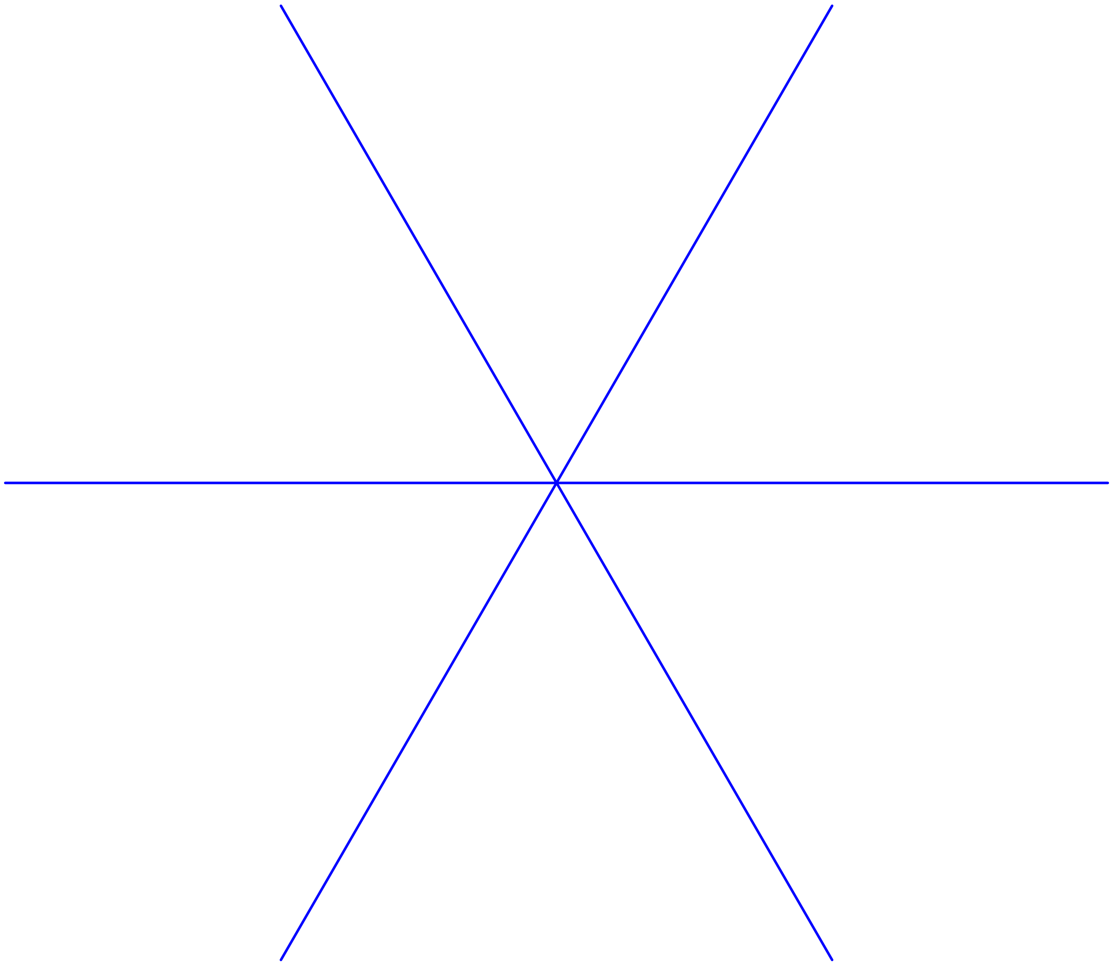
Würfelecken



Würfel

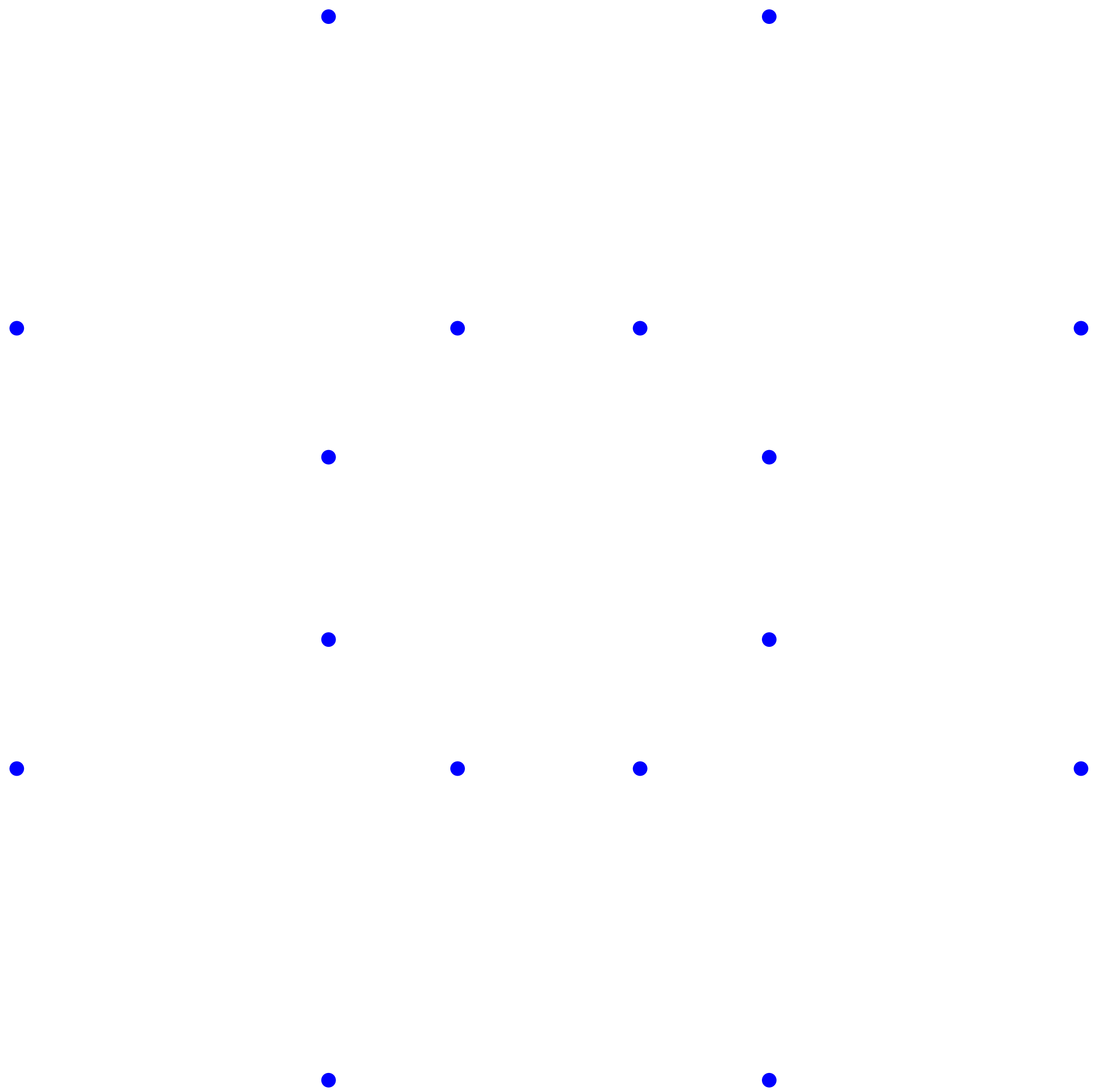


Seitenflächendiagonalen des Würfels

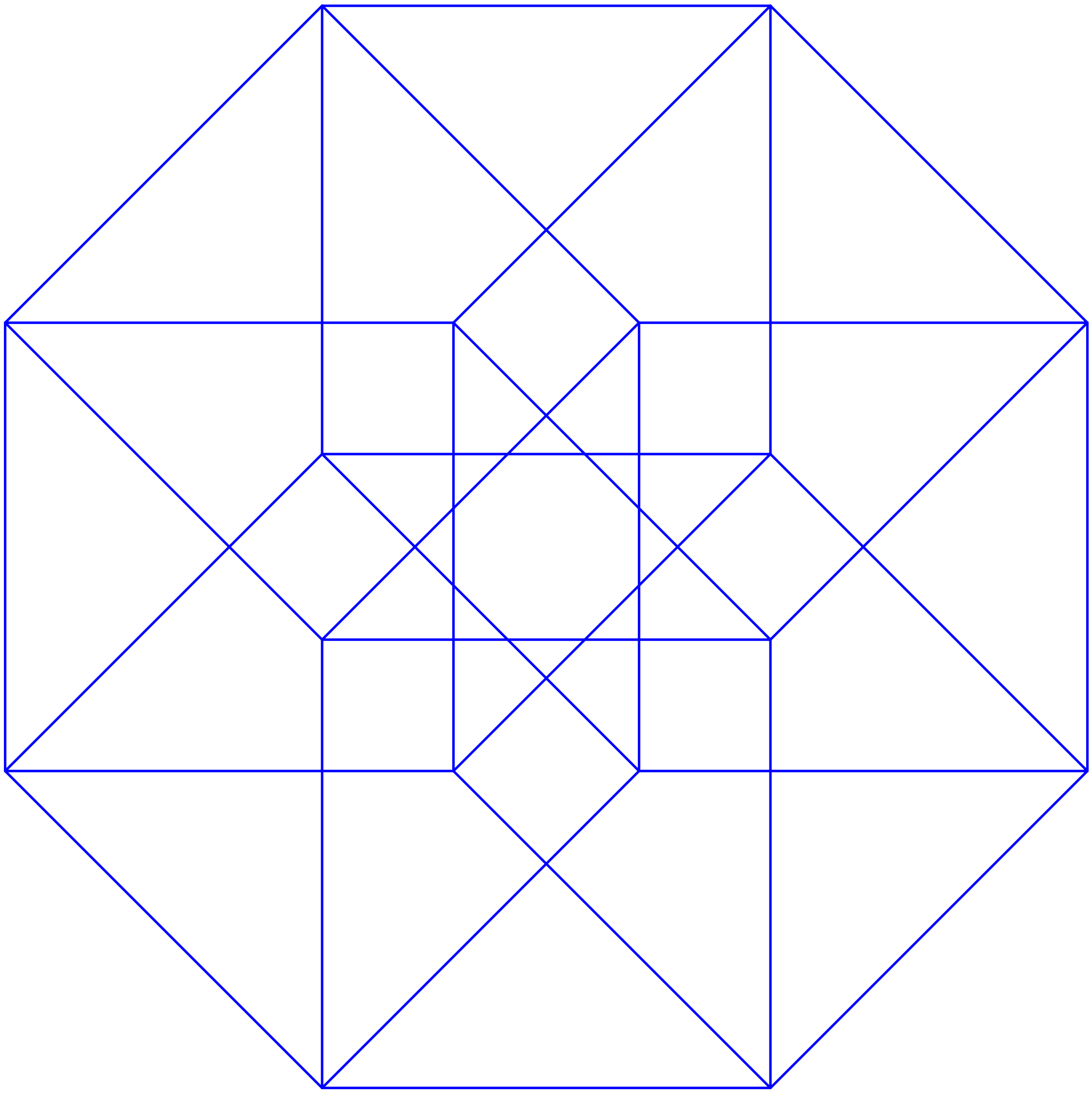


Raumdiagonalen des Würfels

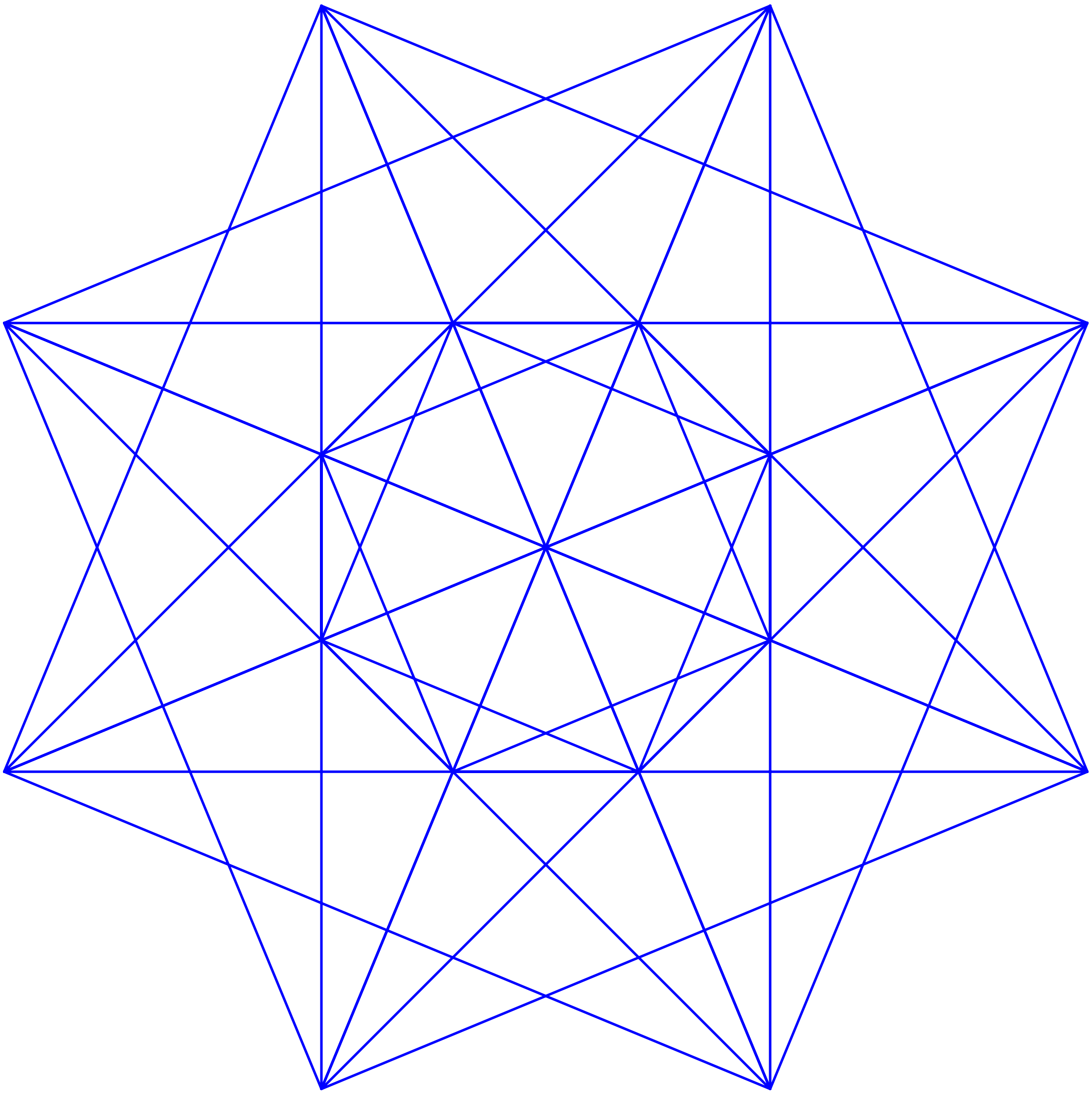
3.4 4-dimensionaler Hyperwürfel



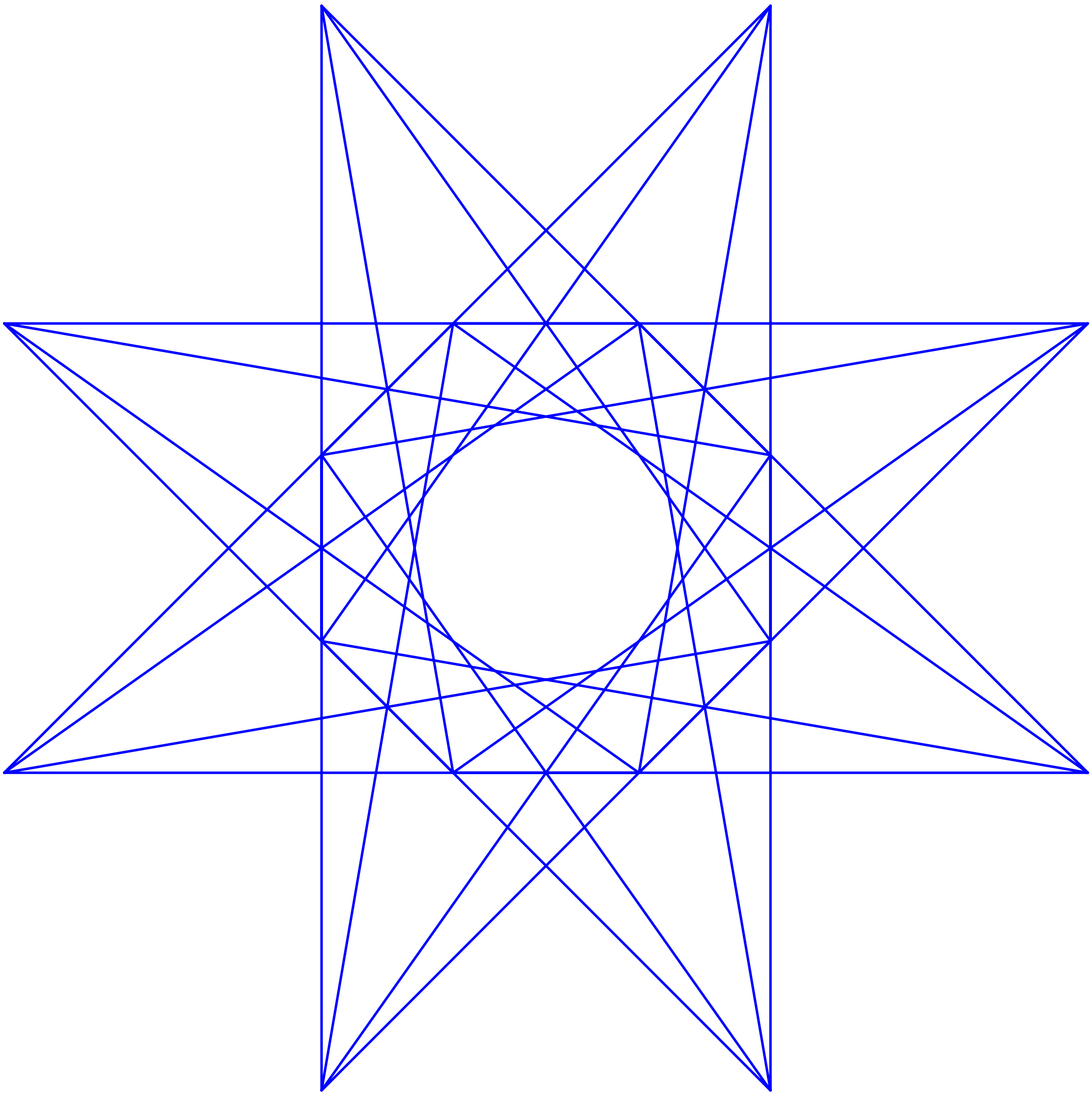
Dimension = 4, Hamming-Distanz = 0



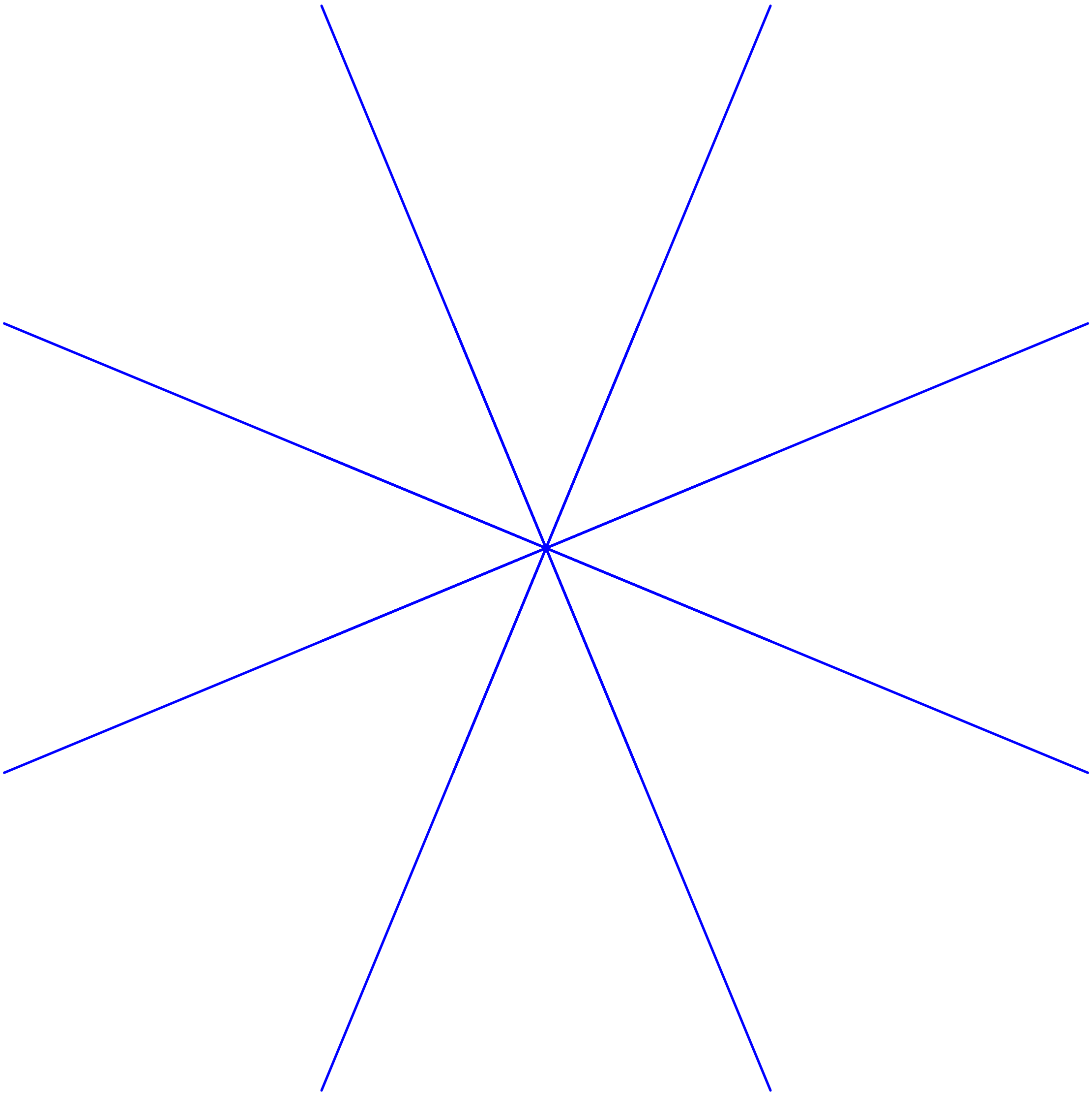
Dimension = 4, Hamming-Distanz = 1



Dimension = 4, Hamming-Distanz = 2

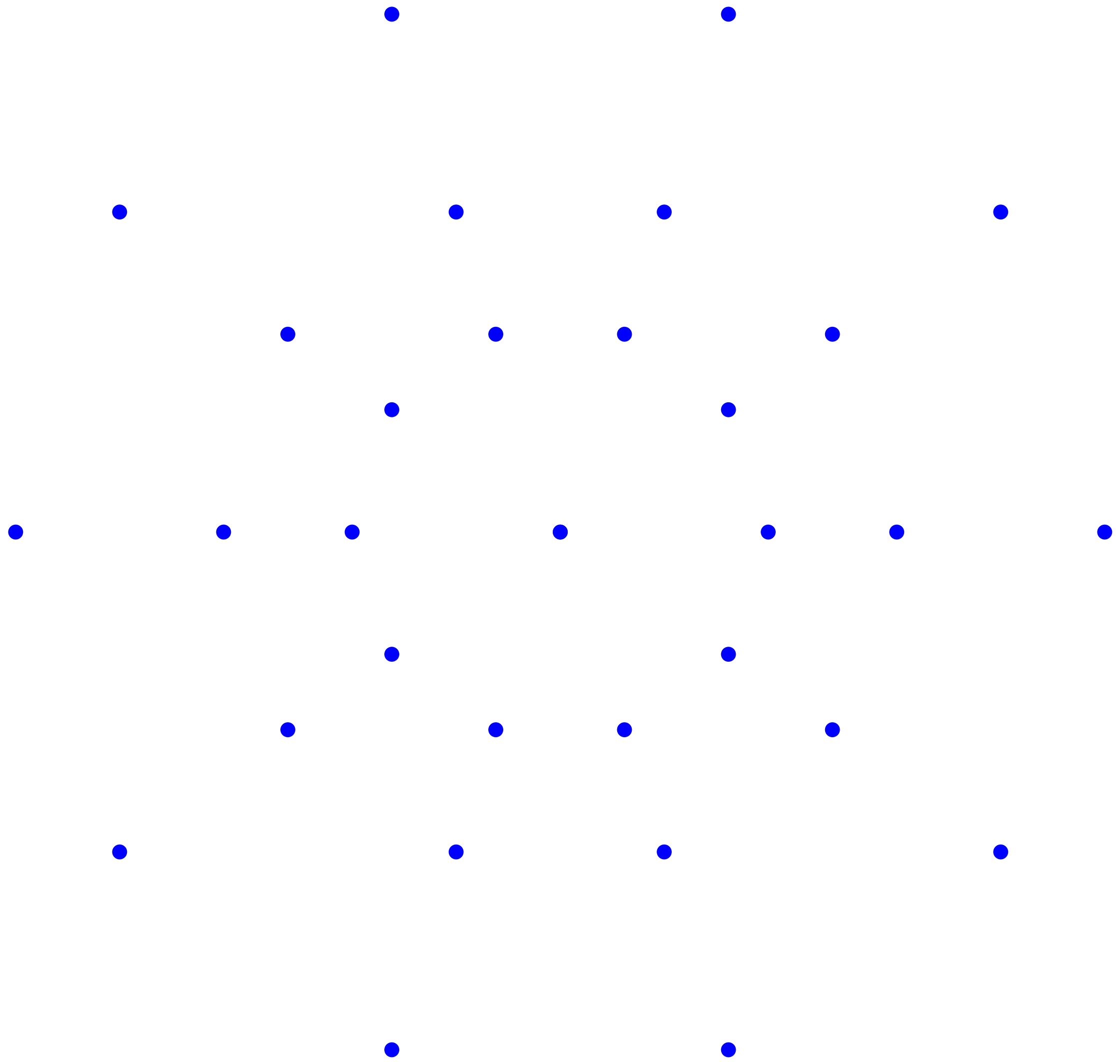


Dimension = 4, Hamming-Distanz = 3

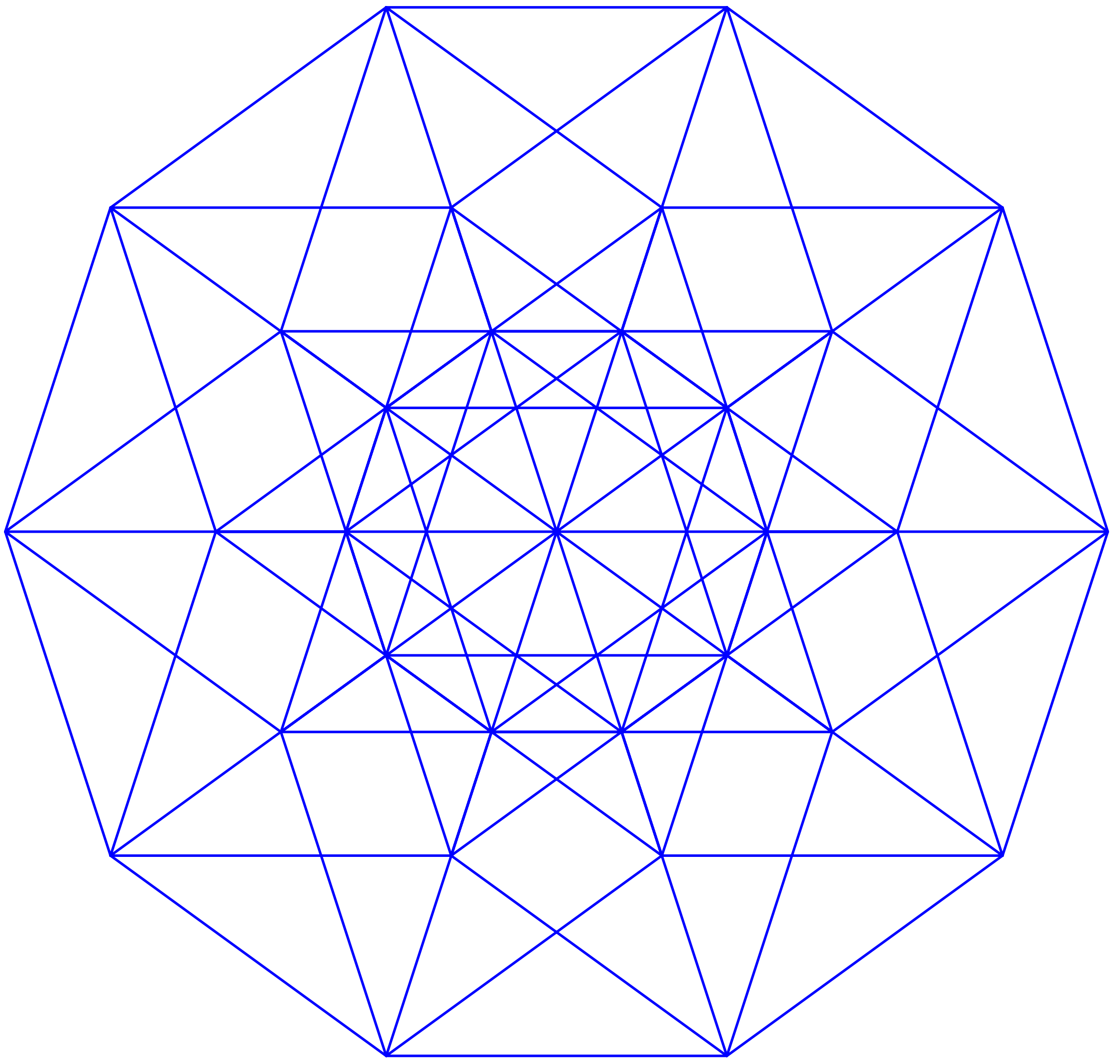


Dimension = 4, Hamming-Distanz = 4

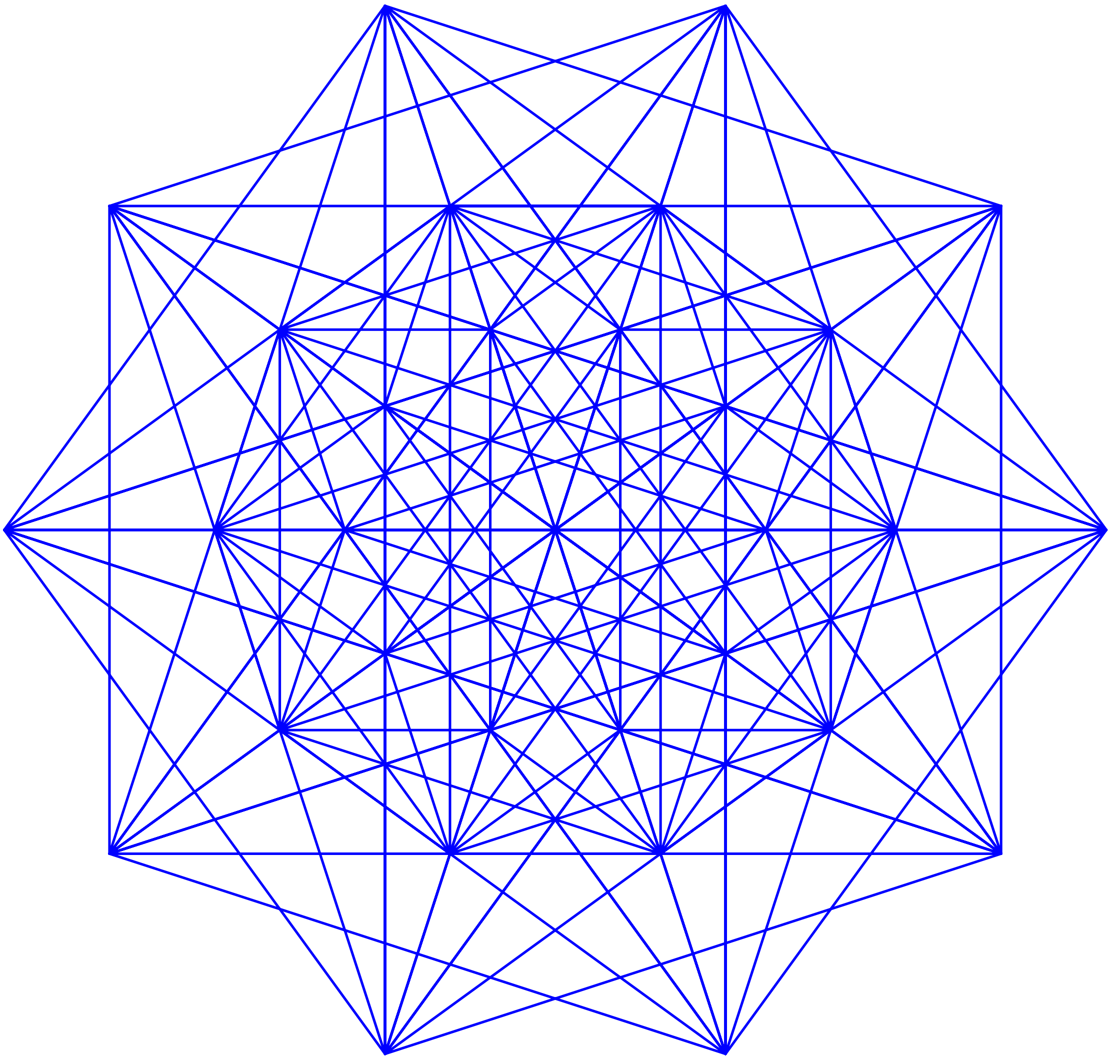
3.5 5-dimensionaler Hyperwürfel



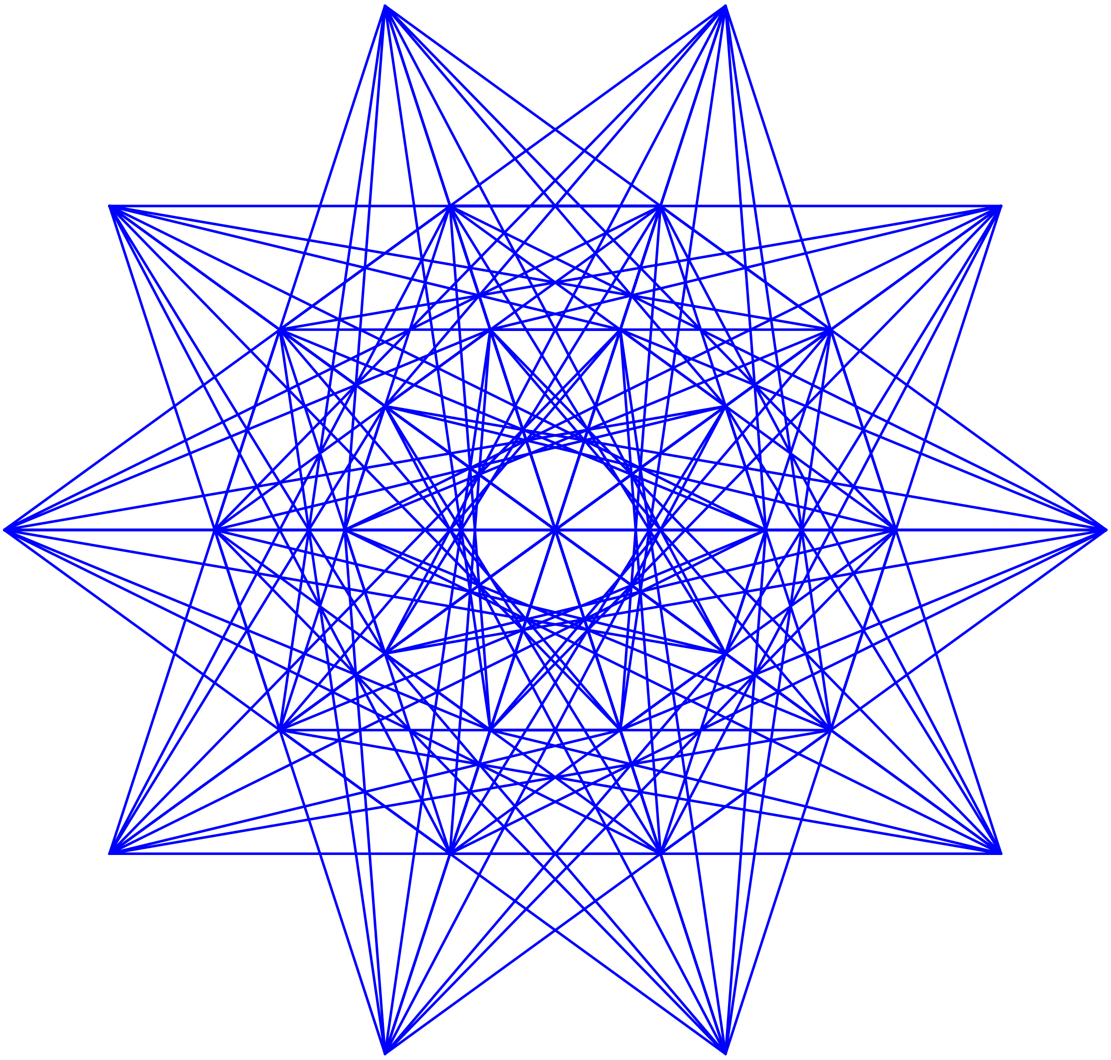
Dimension = 5, Hamming-Distanz = 0



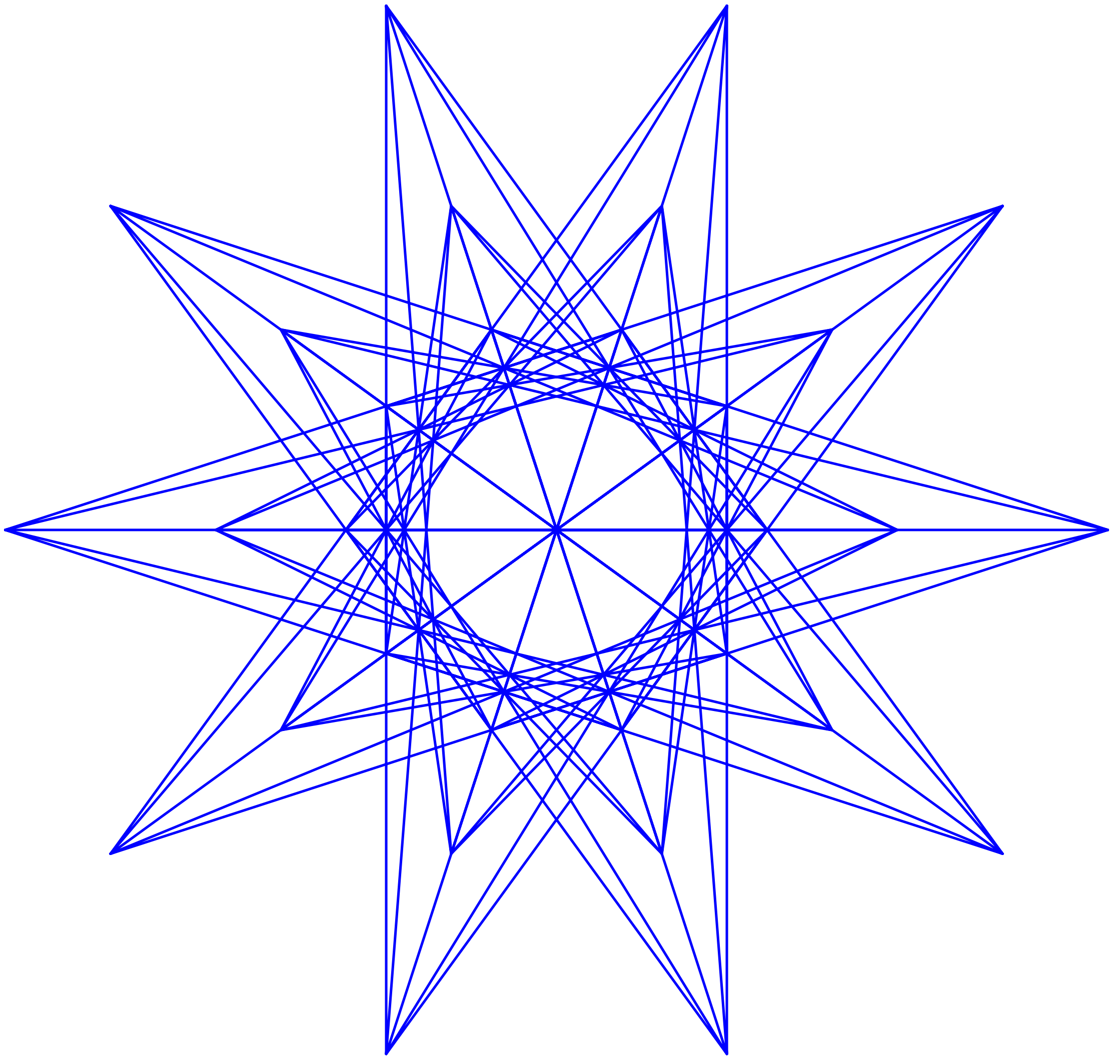
Dimension = 5, Hamming-Distanz = 1



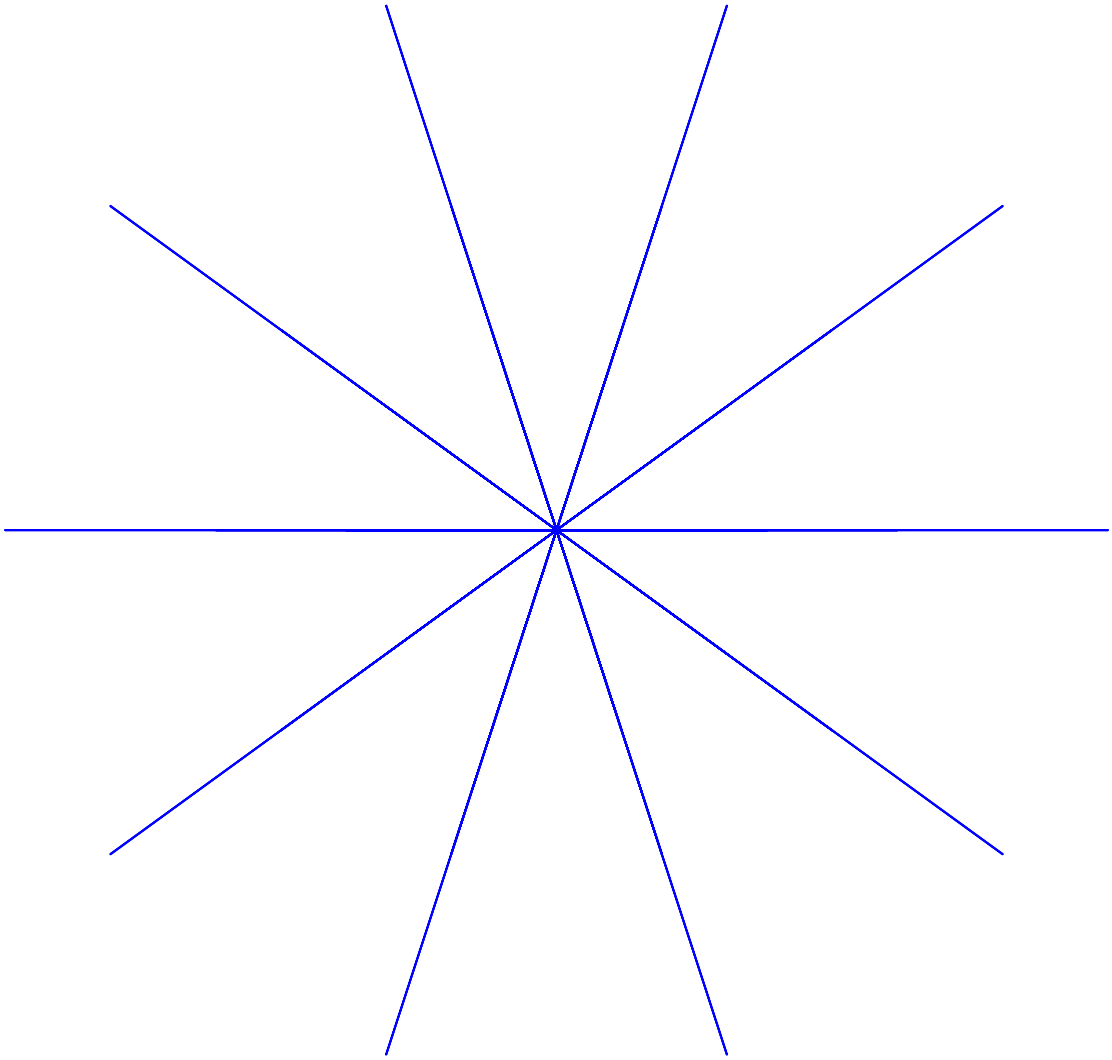
Dimension = 5, Hamming-Distanz = 2



Dimension = 5, Hamming-Distanz = 3

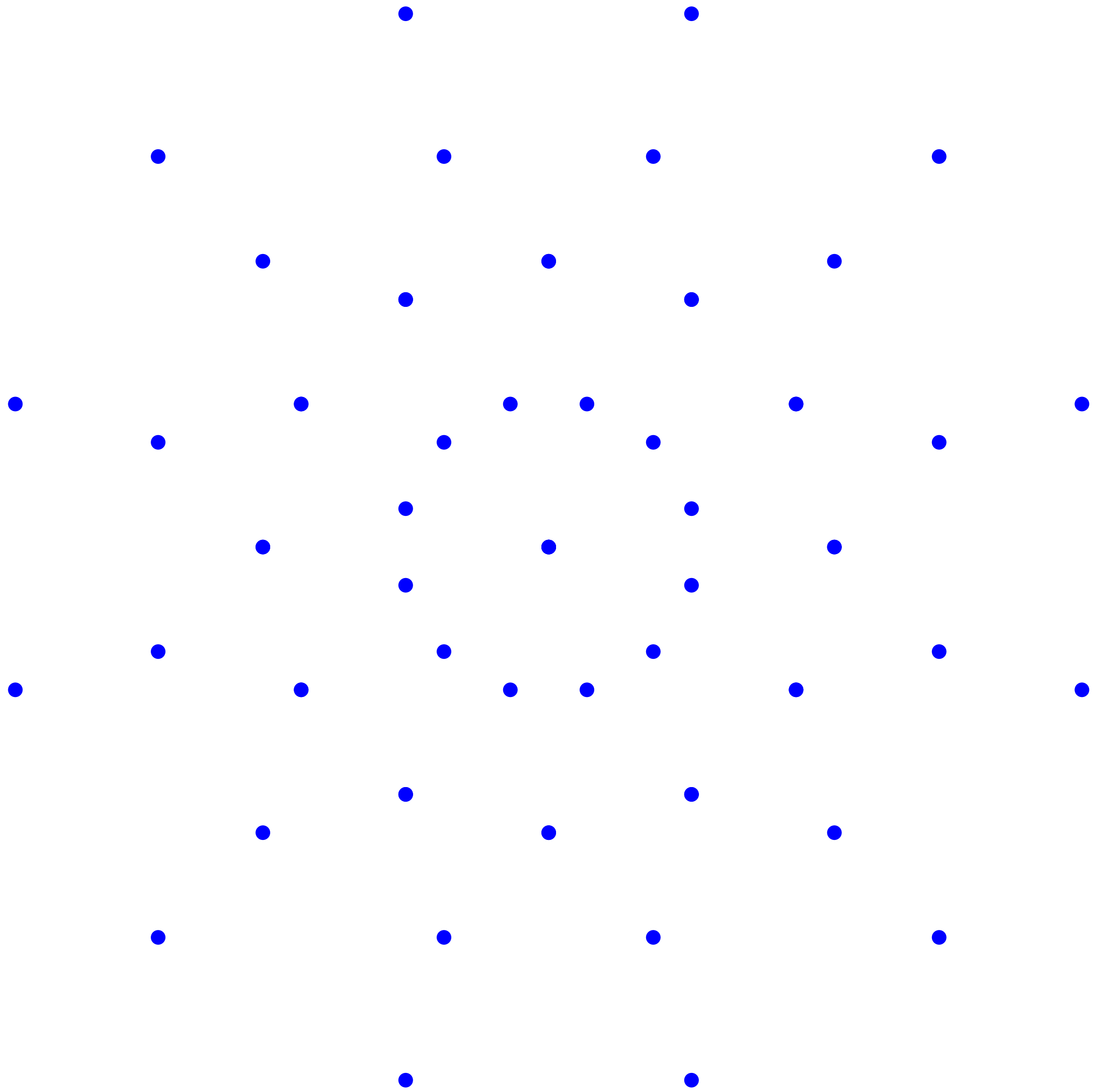


Dimension = 5, Hamming-Distanz = 4

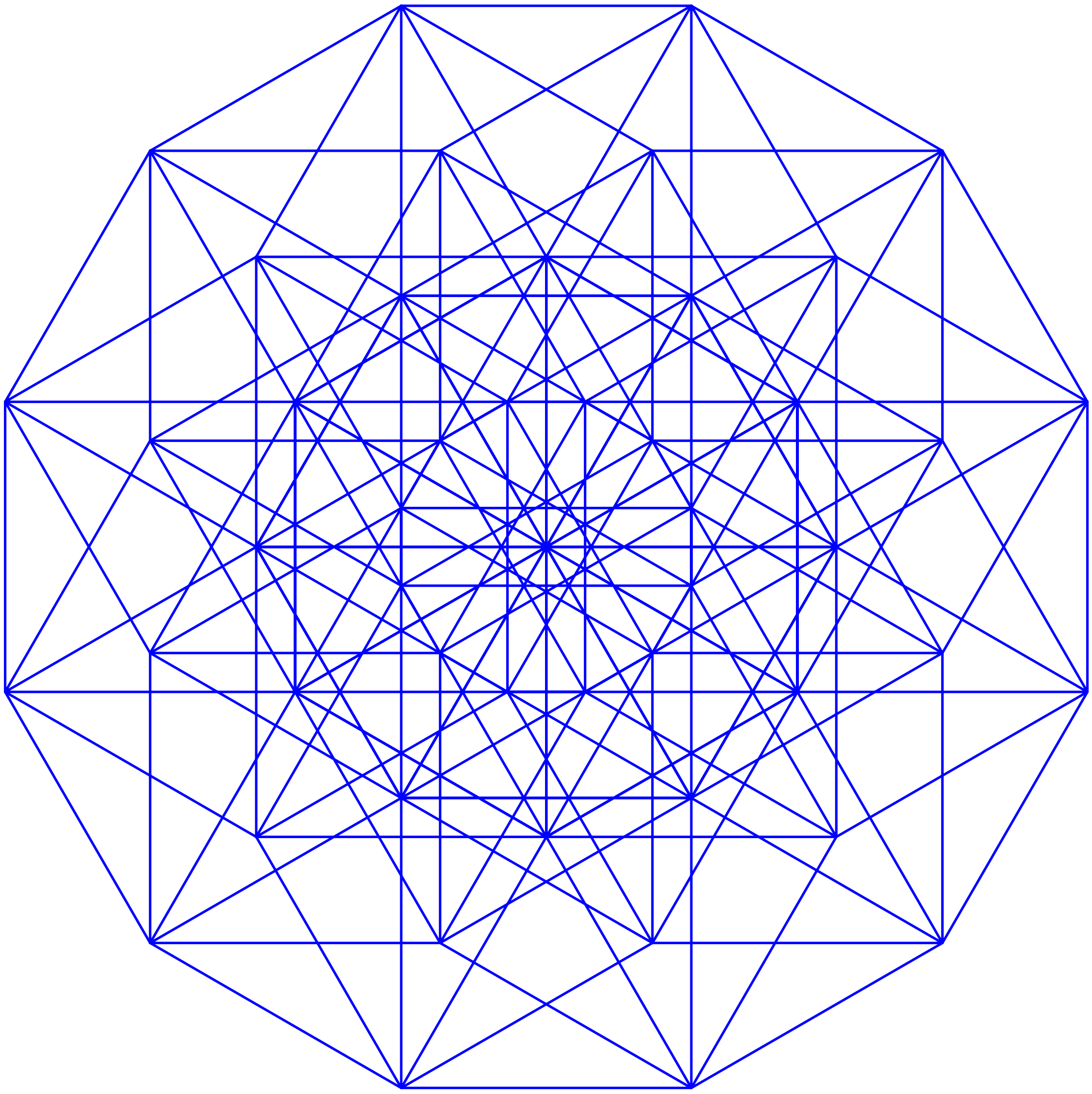


Dimension = 5, Hamming-Distanz = 5

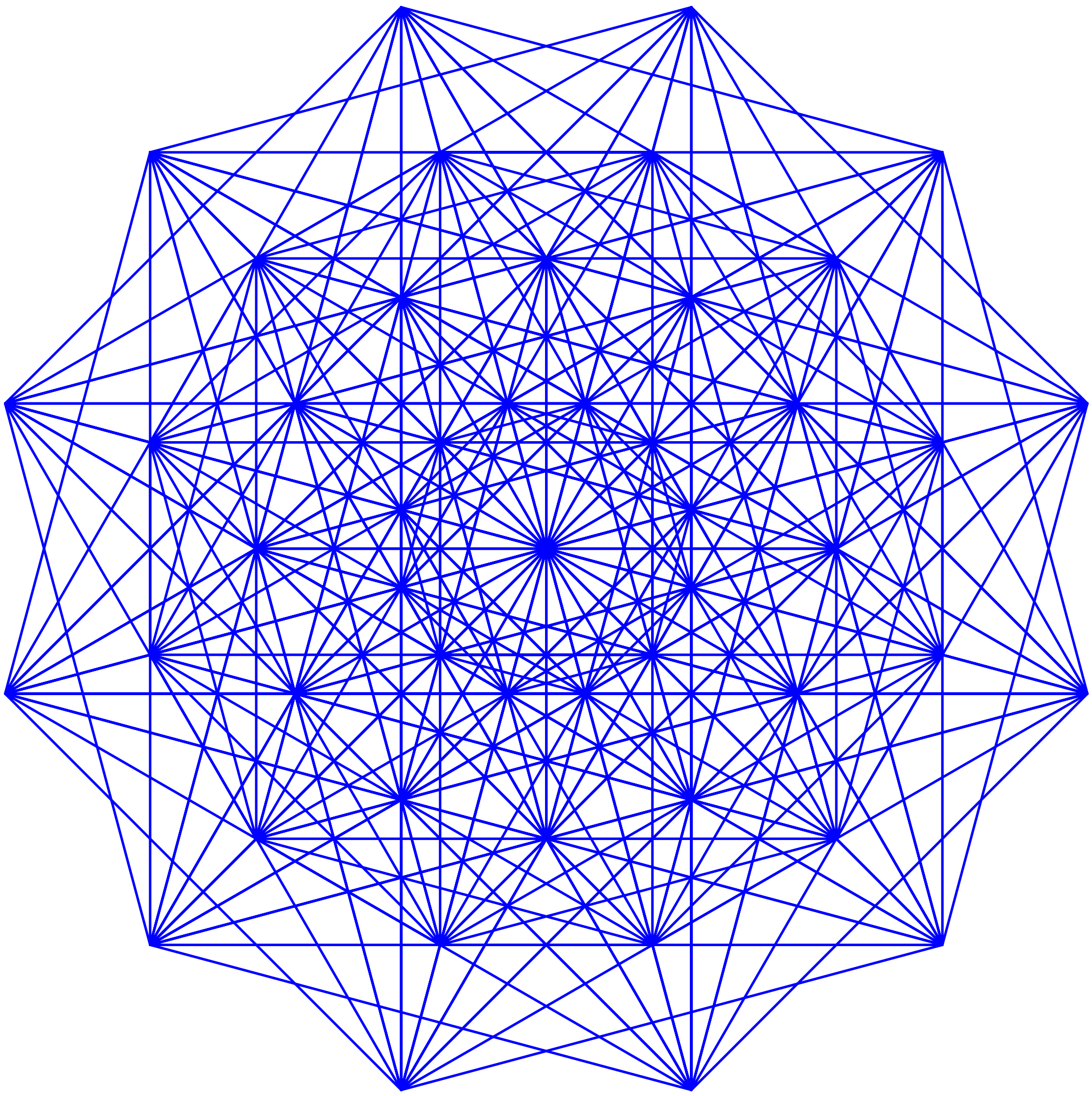
3.6 6-dimensionaler Hyperwürfel



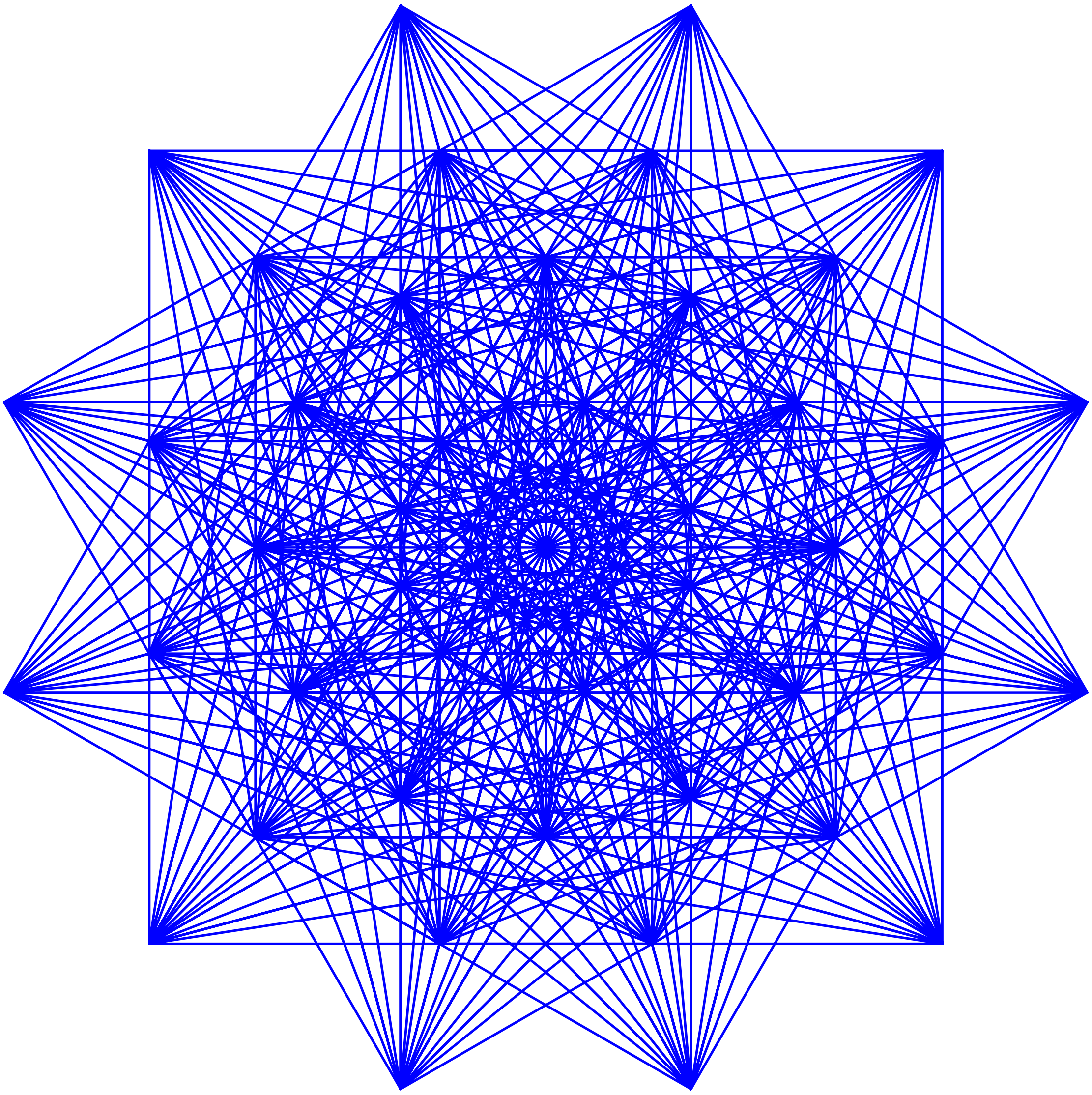
Dimension = 6, Hamming-Distanz = 0



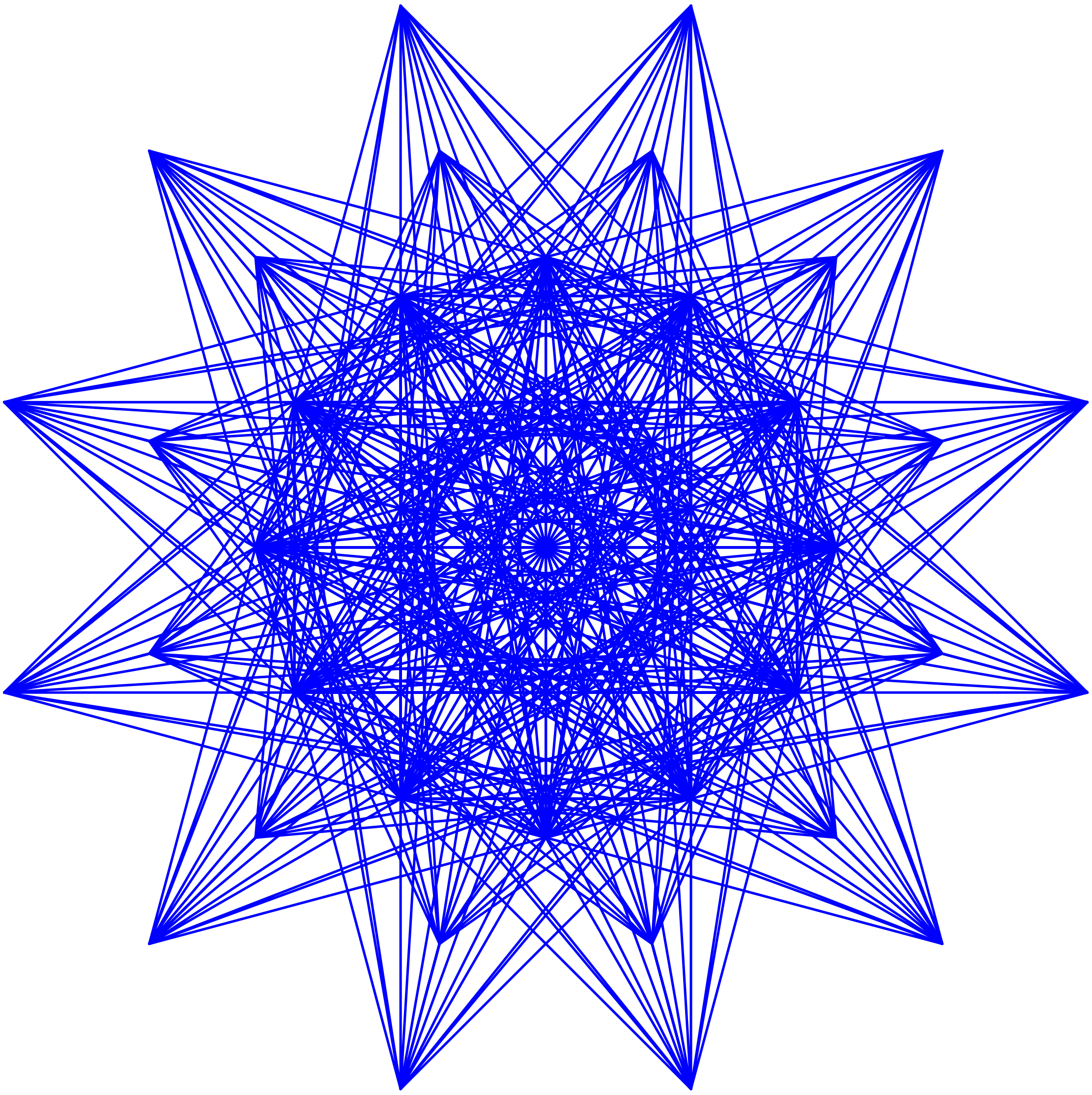
Dimension = 6, Hamming-Distanz = 1



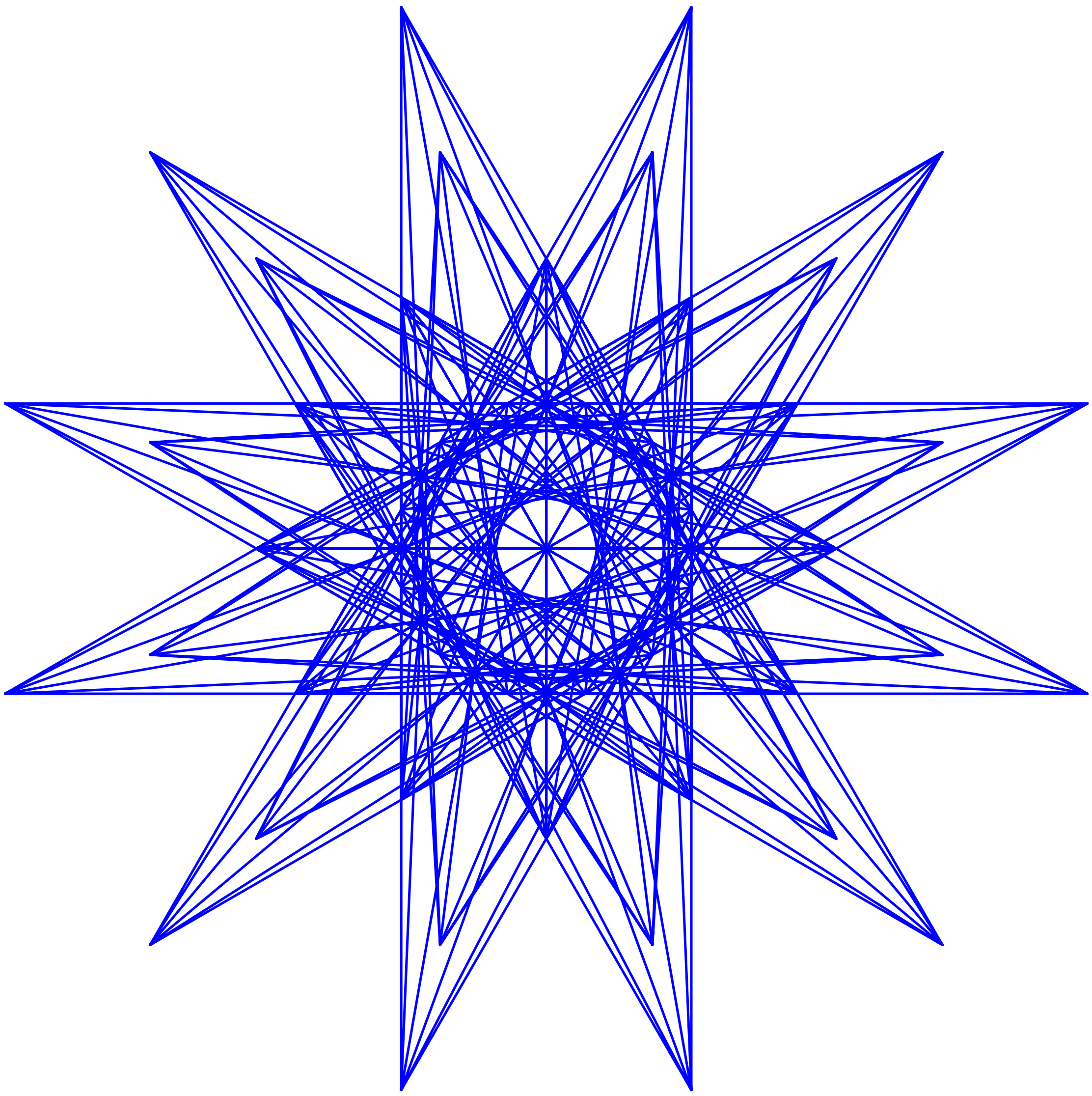
Dimension = 6, Hamming-Distanz = 2



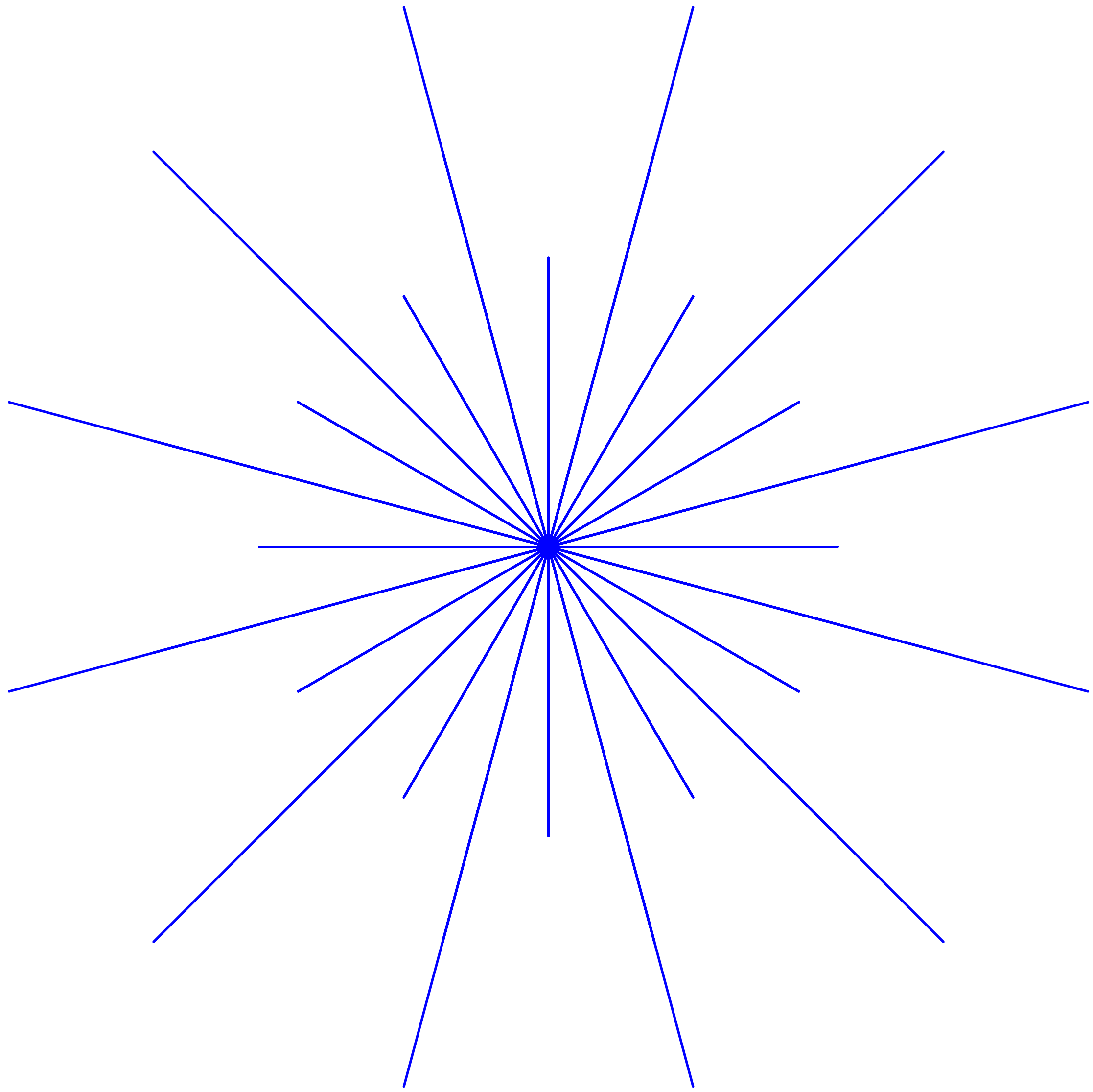
Dimension = 6, Hamming-Distanz = 3



Dimension = 6, Hamming-Distanz = 4

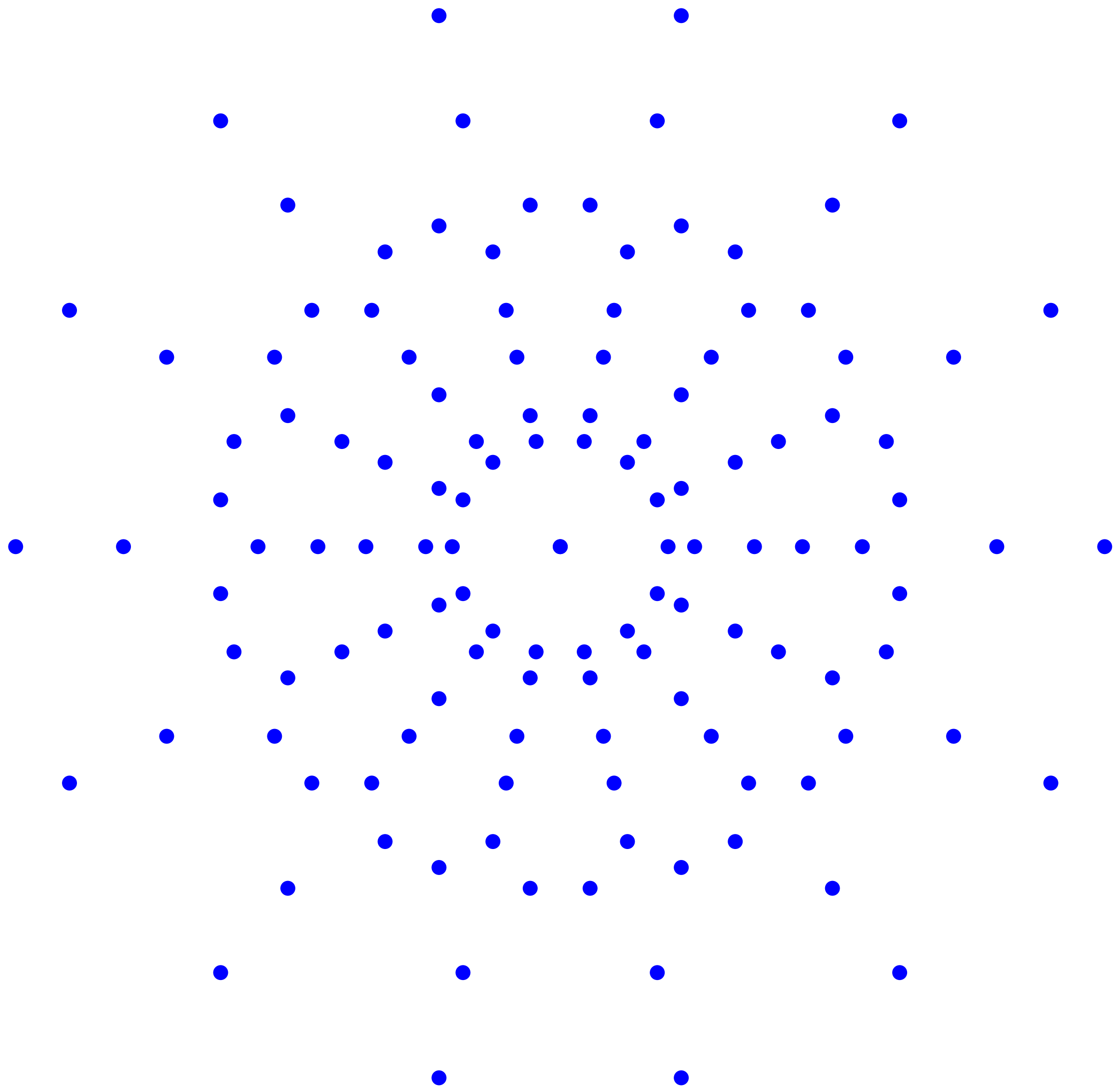


Dimension = 6, Hamming-Distanz = 5

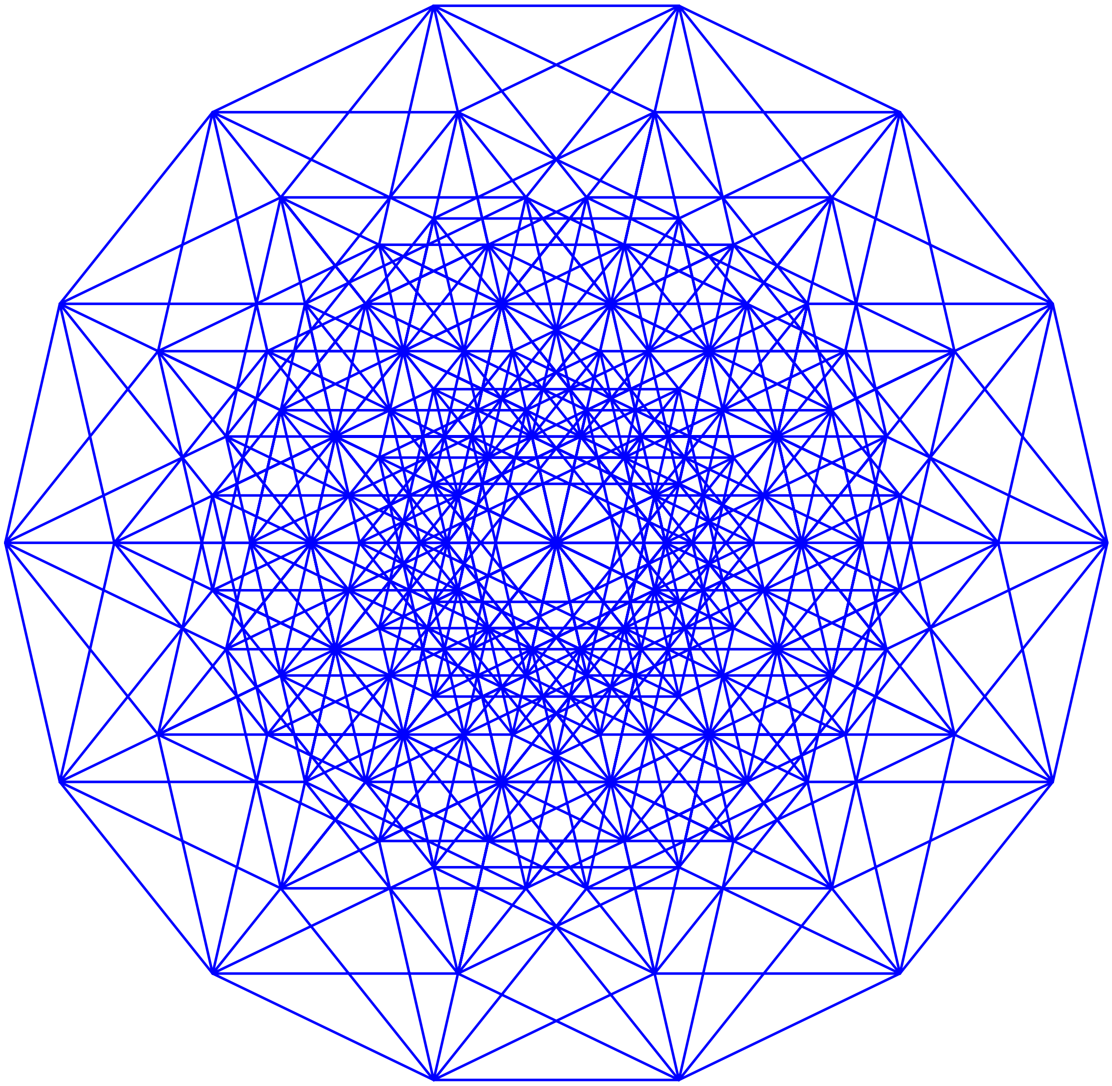


Dimension = 6, Hamming-Distanz = 6

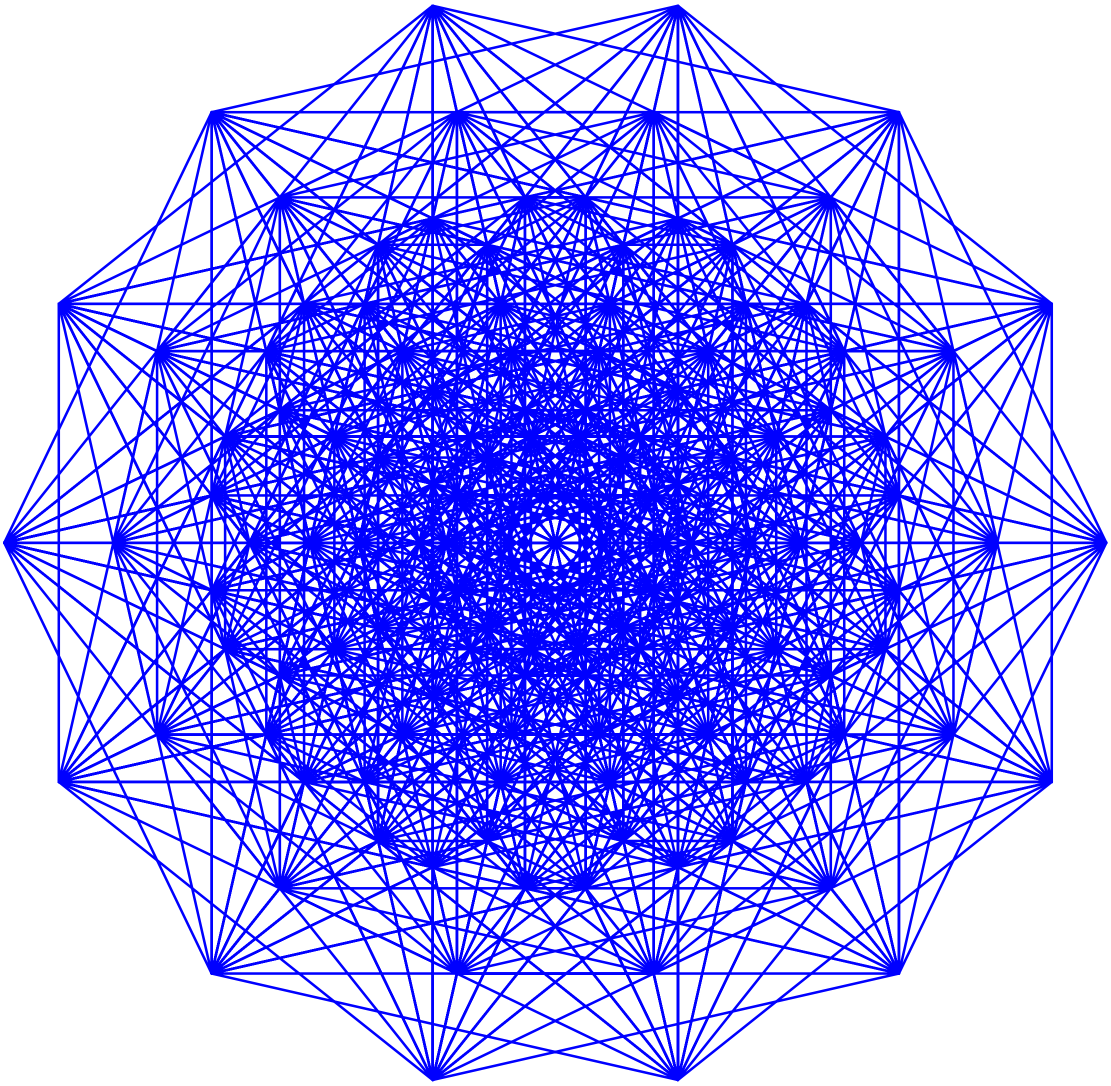
3.7 7-dimensionaler Hyperwürfel



Dimension = 7, Hamming-Distanz = 0



Dimension = 7, Hamming-Distanz = 1



Dimension = 7, Hamming-Distanz = 2

Und so weiter und so fort.

4 MuPAD-Programm

Im Folgenden exemplarisch das Programm für den vierdimensionalen Würfel mit Kanten ($n = 4$, Hamming-Distanz = 1):

```
n:=4:
N:=2^n-1:
Hamming:=1:

for k from 0 to N do
 for j from 1 to n do
  K[k,j]:=round(frac(k*(1/2)^(n-j+1))):
 end_for:
end_for:

for k1 from 0 to N do
 for k2 from k1 to N do
  H[k1,k2]:=0:
  for j from 1 to n do
   H[k1,k2]:=H[k1,k2]+( (K[k1,j]+K[k2,j]) mod 2):
  end_for:
 end_for:
end_for:

for k from 0 to N do
 x[k]:=0:
 y[k]:=0:
 for j from 1 to n do
  x[k]:=x[k]+round(frac(k*(1/2)^(n-j+1)))*cos(j*PI/n):
  y[k]:=y[k]+round(frac(k*(1/2)^(n-j+1)))*sin(j*PI/n):
 end_for:
end_for:

for k from 0 to N do
 P[k]:=[x[k],y[k]]:
end_for:

Punkt:=k->plot::Point2d(P[k], PointSize=2, PointColor=[1,0,0],
Visible=FALSE):

Kante:=(k1,k2)->plot::Polygon2d([P[k1],P[k2]]):

Kanten:=proc(k1,k2)
 begin
  if H[k1,k2]=Hamming then Kante(k1,k2):
  else Punkt(k1):
  end_if:
 end_proc:

plot((Kanten(k1,k2)$k2=k1..N)$k1=0..N,  Scaling=Constrained,
Axes=None, Width=150, Height=150):
```