Capítulo 2: Politopos

Un politopo es la generalización del concepto Polígono (2D), o poliedro (3D) a cualquier otra dimensión. Geométricamente un politopo es una región finita de un espacio *n*-dimensional encerrado por un número finito de hiperplanos. La parte de un politopo que se sitúa en uno de los hiperplanos, es llamada celda, y todas las celdas del politopo forman la "frontera" del mismo. Si los puntos medios de todas las aristas que salen de un vértice dado *v* de un politopo, están en un hiperplano, estos son los vértices de un politopo de dimensión (*n*-1), llamado figura vértice del politopo original en *v* [Hausmann 94].

2.1 Politopos Regulares

Los politopos regulares han sido materia estudios matemáticos desde hace mucho tiempo, en dos dimensiones estos son los polígonos regulares (con todos sus lados y ángulos iguales), en tres dimensiones estos son los poliedros regulares, también conocidos como sólidos platónicos (el tetraedro, el cubo, el octaedro, el dodecaedro y el icosaedro) y en cuatro dimensiones existen seis politopos regulares (simplex 4D, hipercubo 4D, 16-celdas, 24-celdas, 120-celdas y 600-celdas).

En dimensiones superiores existen solo tres politopos regulares "canónicos", el simplex, el hipercubo y el politopo cruz [Coxeter 63].

Se pueden definir los politopos regulares inductivamente de la siguiente forma: Un politopo se dice que es regular si todas sus celdas son regulares y hay una figura vértice regular en cada vértice [Coxeter 63]. En un politopo regular todas las celdas son iguales entre sí, similarmente, las figuras vértice de todos los vértices del politopo son iguales.

2.1.1 Ángulos

Antes de continuar con la explicación de los politopos, se definen algunos de los ángulos entre elementos geométricos (rectas, planos e hiperplanos) en el espacio *n*D.

2.1.1.1 Ángulo Entre Dos Rectas

En 2D, el ángulo (rectilíneo) es la porción del plano limitada por dos rectas con origen en un mismo punto llamado vértice del ángulo.

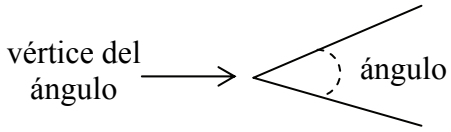

Figura 2.1 Ángulo entre dos rectas.

2.1.1.2 Ángulo Entre Dos Planos

En la geometría 3D, se dice que el ángulo entre dos planos es llamado ángulo diédrico. Si se toman dos planos intersectados a lo largo de una recta en común, llamada *arista*, se puede ver que los planos forman un ángulo, este ángulo se le conoce como *ángulo diédrico*, y este se mide por su ángulo rectilíneo correspondiente, que es el ángulo formado por las rectas sobre los planos y que son perpendiculares a la arista.

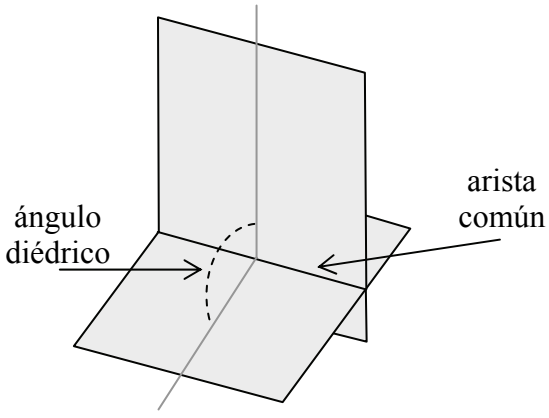

Figura 2.2 Ángulo entre dos planos.

2.1.1.3 Ángulo Entre Dos Hiperplanos

En [Olshevsky 06] se generaliza la definición de ángulo diédrico en cualquier dimensión, diciendo que: Un ángulo diédrico es el ángulo entre dos hiperplanos (n-1)D que se intersectan en el espacio nD. Dos hiperplanos (n-1)D se intersectan en un hiperplano (n-2)D. Para medir el ángulo diédrico, se escoge un punto en el hiperplano (n-2)D y se trazan dos rayos perpendiculares al hiperplano (n-2)D que caiga en cada uno de los dos hiperplanos (n-1)D. El ángulo diédrico entre los hiperplanos (n-1)D es el ángulo entre estos dos rayos.

2.1.2 Politopos Regulares 2D

Un politopo 2D (polígono) es una figura formada por una secuencia de puntos $A_1, A_2, \ldots, A_n$, y por los segmentos $\overline{A_1A_2}, \overline{A_2A_3}, \ldots, \overline{A_nA_1}$ que los unen (Figura 2.3), los puntos se llaman *vértices* del polígono, y los segmentos *lados* o *aristas* del mismo. Un polígono se dice que es convexo si dados cualesquiera dos puntos p y q interiores al polígono, el segmento rectilíneo $\overline{pq}$ está completamente contenido dentro del polígono [Do Carmo 76].

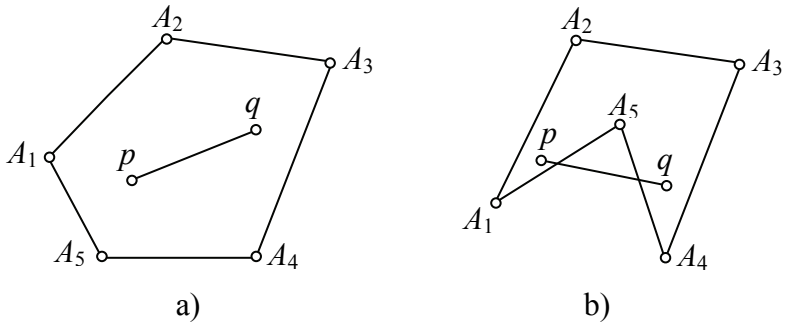

Figura 2.3 Ejemplo de polígonos: a) convexo, b) no convexo.

Los polígonos tienen un ángulo rectilíneo en cada arista, que es el ángulo interior que se forma donde dos lados adyacentes coinciden.

Un polígono regular es un polígono convexo con todos sus lados y ángulos interiores iguales. Algunos ejemplos de polígonos regulares son los siguientes (Figura 2.4):

- Triángulo equilátero, un triángulo con todos sus lados iguales y todos sus ángulos interiores iguales a 60°.
- Cuadrado, un cuadrilátero con todos sus lados iguales y ángulos interiores iguales a 90°.
- Pentágono regular, un polígono de cinco lados iguales y cinco ángulos interiores iguales a 108°.
- Hexágono regular, un polígono de seis lados iguales y seis ángulos interiores iguales a 120°.
- Octágono regular, un polígono de ocho lados iguales y ocho ángulos interiores iguales a 135°.

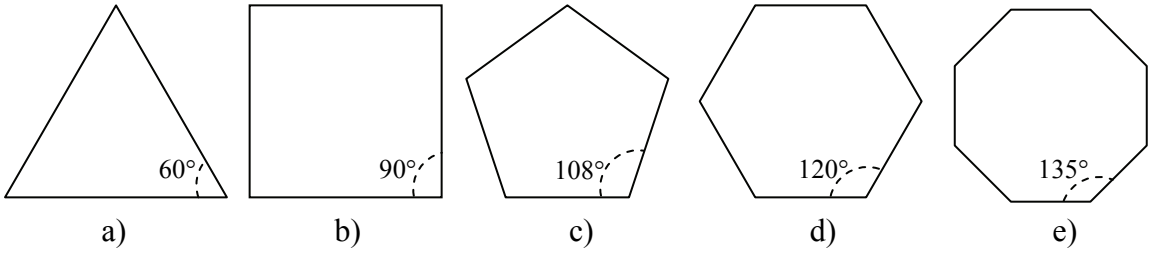

Figura 2.4 Algunos polígonos regulares. a) Triángulo rectángulo, b) Cuadrado, c) Pentágono regular, d) Hexágono regular, e) Octágono regular.

2.1.3 Politopos Regulares 3D

Los politopos regulares 3D (poliedros) están limitados en su frontera por caras, las cuales son polígonos regulares (ver Sección 2.1.2). Se pueden denotar mediante la pareja $\{p, q\}$, donde p es el número de lados que tienen los polígonos de las caras del poliedro, y q es el número de caras incidentes a cada vértice del poliedro.

Los poliedros, tienen un ángulo diédrico (también llamado ángulo de las caras) en cada arista, que es el ángulo interno que se forma donde dos caras adyacentes coinciden. Cada ángulo diédrico en un poliedro regular tiene el mismo valor.

Se sabe que sólo los siguientes cinco poliedros regulares (también conocidos cómo sólidos platónicos) son posibles [Coxeter 63]:

- Tetraedro {3, 3}, cuya frontera está formada por cuatro triángulos equiláteros, con tres incidentes localizados en cada vértice. Todos sus ángulos diédricos miden 70.53°.

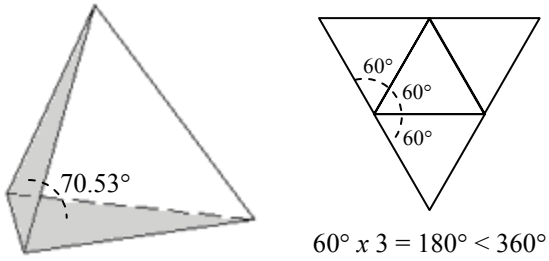

Figura 2.5 El tetraedro y el desenvolvimiento de su frontera.

- Cubo {4, 3} cuya frontera está formada por seis cuadrados, con tres localizados en cada vértice. Todos sus ángulos diédricos miden 90°.

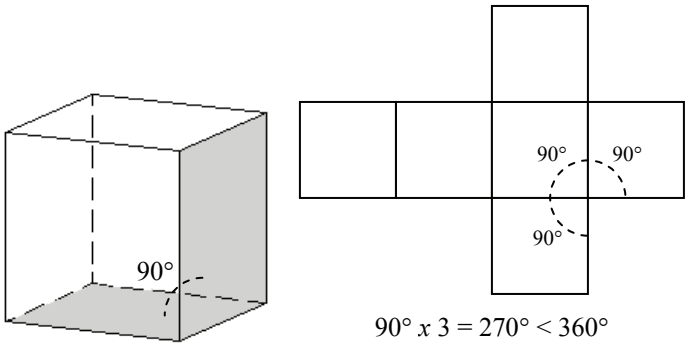

Figura 2.6: El cubo y el desenvolvimiento de su frontera.

- Octaedro {3, 4} cuya frontera está formada por ocho triángulos equiláteros, con cuatro localizados en cada vértice. Todos sus ángulos diédricos miden 109.45°.

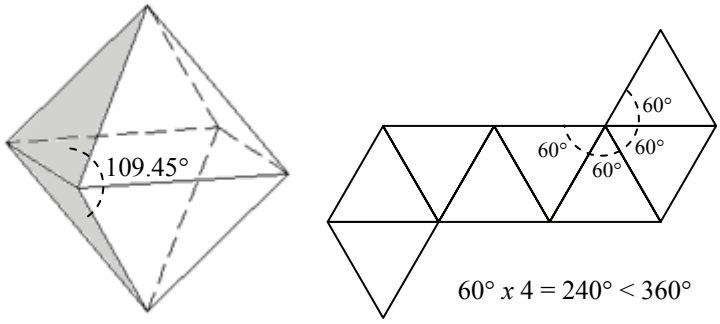

Figura 2.7: El octaedro y el desenvolvimiento de su frontera.

- Dodecaedro {5, 3} cuya frontera está formada por doce pentágonos equiláteros, con tres localizados en cada vértice. Todos sus ángulos diédricos miden 116.57°.

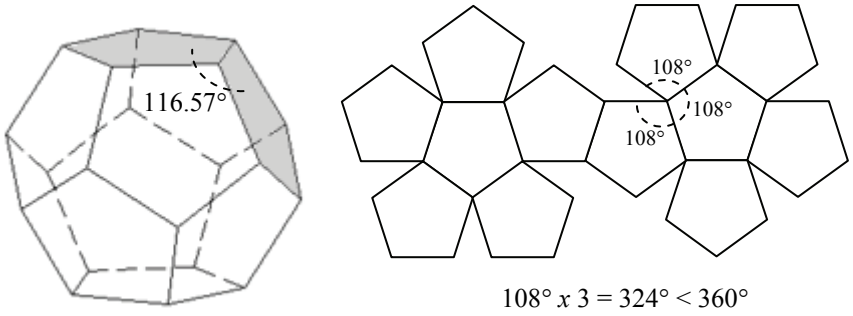

Figura 2.8: El Dodecaedro y el desenvolvimiento de su frontera.

- Icosaedro {3, 5} cuya frontera está formada por veinte triángulos equiláteros, con cinco localizados en cada vértice. Todos sus ángulos diédricos miden 138.2°.

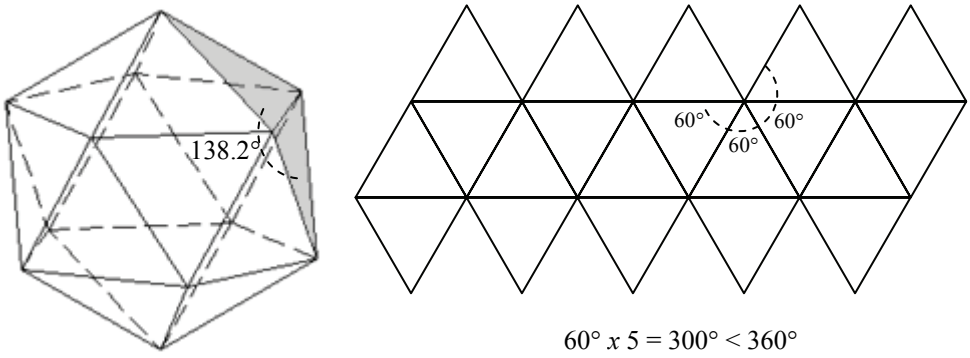

Figura 2.9: El Icosaedro y el desenvolvimiento de su frontera.

No se puede tener otros poliedros debido a las siguientes restricciones:

1. Al menos tres caras deben localizarse en un vértice, y,
2. Cuando se traza sobre un plano el desenvolvimiento de un poliedro, debe haber una abertura en alguna parte entre las caras adyacentes que están alrededor de un vértice, de otra forma el poliedro no puede ser envuelto en la tercera dimensión, en otras palabras, el ángulo cubierto por las caras que se localizan en un vértice debe ser menor a 360°.

Estas restricciones excluyen todas excepto las siguientes posibilidades: Se pueden poner tres, cuatro o cinco triángulos equiláteros, o tres cuadrados, o tres pentágonos equiláteros alrededor de un vértice, a partir de los hexágonos, ni siquiera se pueden colocar tres debido al tamaño de sus respectivos ángulos internos (Ver sección 2.1.2). Esto lleva exactamente a los cinco sólidos platónicos:

1. El tetraedro (Figura 2.5) tiene alrededor de cada vértice 3 triángulos equiláteros, entonces se cumple que $60° \times 3 = 180° < 360°$.
2. El cubo (Figura 2.6) tiene alrededor de cada vértice 3 cuadrados, entonces se cumple que $90° \times 3 = 270° < 360°$.
3. El octaedro (Figura 2.7) tiene alrededor de cada vértice 4 triángulos equiláteros, entonces se cumple que $60° \times 4 = 240° < 360°$.
4. El dodecaedro (Figura 2.8) tiene alrededor de cada vértice 3 pentágonos equiláteros, entonces se cumple que $108° \times 3 = 324° < 360°$.
5. El icosaedro (Figura 2.9) tiene alrededor de cada vértice 3 triángulos equiláteros, entonces se cumple que $60° \times 5 = 300° < 360°$.

2.1.4 Politopos Regulares 4D

Sean $\{p, q, r\}$ que denotan a un politopo 4D regular, donde cada celda $\{p, q\}$ de su frontera es un sólido platónico, y hay r celdas adyacentes a cada arista. Al igual que en 3D, se pueden "llevar" las celdas de un politopo regular a tres dimensiones y se puede apreciar que se mantienen las siguientes restricciones:

1. Al menos tres celdas deben localizarse en una arista.
2. Si se colocan todas las celdas que se localizan en una arista alrededor de esa arista, entonces debe haber una abertura en alguna parte entre las celdas, de otra forma el politopo no puede ser envuelto en la cuarta dimensión, en otras palabras, el ángulo diédrico cubierto por las celdas que se localizan en una arista debe ser menor a 360°).

Estas restricciones, aunado al tamaño de los ángulo diédricos en los poliedros regulares (ver Sección 2.1.3) excluyen todos, excepto los siguientes seis casos: Se puede poner tres, cuatro o seis tetraedros regulares alrededor de una arista, tres cubos, tres octaedros o tres dodecaedros [Coxeter 63], estas seis posibilidades llevan a los politopos regulares 4D (Figura 2.10 – Figura 2.15).

Los politopos 4D tienen un ángulo diédrico en cada arista, que es el ángulo interno que se forma donde dos sólidos adyacentes coinciden. Cada ángulo diédrico en un politopo 4D regular tiene el mismo valor.

- Simplex 4D {3, 3, 3} formado por cinco tetraedros, conteniendo tres por arista, entonces se cumple que $70.53° \times 3 = 211.59° < 360°$.

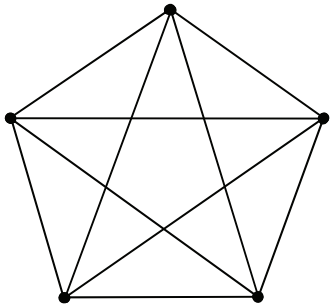

Figura 2.10: Simplex 4D.

- Hipercubo {4, 3, 3} formado de seis cubos, conteniendo tres por arista, entonces se cumple que 90° *x* 3 = 270° < 360°.

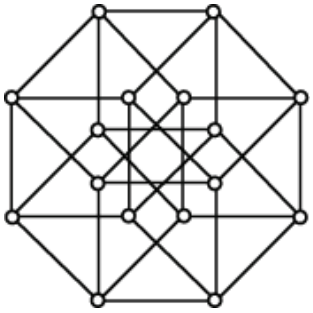

Figura 2.11: Hipercubo 4D.

- 16-celdas {3, 3, 4}, también conocido como politopo cruz, está formado por dieciséis tetraedros, conteniendo cuatro por arista, entonces se cumple que 70.53° *x* 4 = 282.12° < 360°.

Figura 2.12: 16 celdas (Politopo cruz 4D).

- 24-celdas {3, 4, 3} formado por 24 octaedros, conteniendo tres por arista, entonces se cumple que 109.45° *x* 3 = 328.35° < 360°.

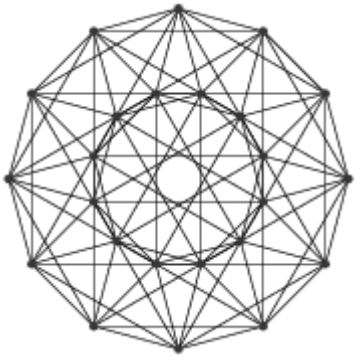

Figura 2.13: 24-celdas.

- 120-celdas {5, 3, 3} formado por 120 dodecaedros, conteniendo tres por arista, entonces se cumple que 116.57° *x* 3 = 349.71° < 360°.

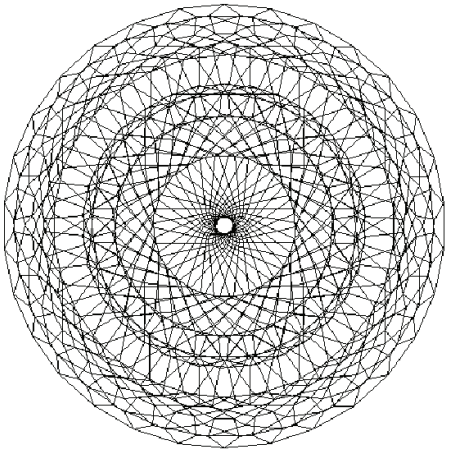

Figura 2.14: 120-celdas.

- 600-celdas {3, 3, 5} formado por 600 tetraedros, conteniendo cinco por arista, entonces se cumple que 70.53° *x* 5 = 352.65° < 360°.

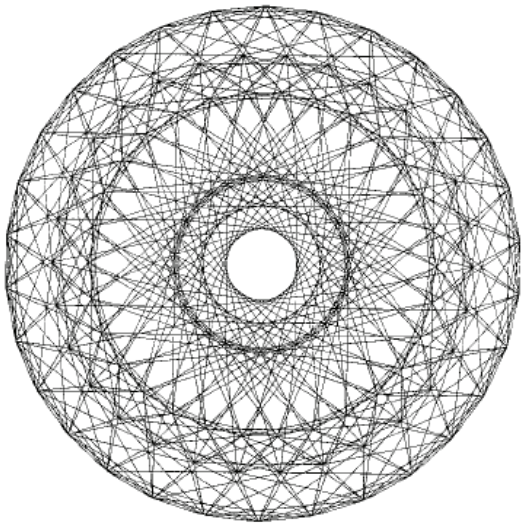

Figura 2.15: 600 celdas.

2.1.5 Politopos Regulares *n*D

Por argumentos similares al caso 3D y 4D, en dimensiones superiores existen solo 3 politopos regulares "canónicos", el simplex, el hipercubo y politopo cruz [Coxeter 63].

Los politopos *n*D tienen un ángulo diédrico en cada arista, que es el ángulo interno que se forma donde dos celdas adyacentes coinciden, esté ángulo es el ángulo diédrico entre los hiperplanos (*n*-1)D que contienen dichas celdas. Cada ángulo diédrico en un politopo *n*D regular tiene el mismo valor.

Ya se sabe lo que son dimensiones, y cómo pueden ser utilizadas, pero lo que puede representar un politopo *n*D podría ser algo no tan sencillo de entender en un principio. Por ejemplo, es fácil para muchos entender lo que es un cubo, pero tal vez no lo sea la analogía de un cubo en 4D. A continuación se analizará la forma en que se puede crear un cubo (3D) a partir del cuadrado (2D), el cual se deriva análogamente de un segmento de recta (1D), que a su vez se deriva del punto (0D).

2.2 Analogías Dimensionales para la Creación de Politopos

Una forma útil para explorar y entender 4D, y en general dimensiones superiores, es la analogía dimensional que existe de los politopos. Con la analogía dimensional se puede examinar cómo un fenómeno geométrico en particular en una dimensión inferior (n-1) se relaciona con un fenómeno geométrico equivalente la dimensión n, y entonces aplicar el mismo principio para relacionar está dimensión a una más grande (n+1).

2.2.1 Hipercubo

Se puede hacer uso de una extrusión (barrido) ortogonal para la producción de politopos de la familia del hipercubo para dimensiones superiores.

Se comienza entonces por el espacio 0D, donde el único objeto cero-dimensional que existe en este espacio es el punto (Figura 2.16), el cual no tiene largo, ancho o altura; ninguna de estas propiedades puedes describirse en este espacio.

Figura 2.16: Un punto.

En 1D para poder obtener un segmento de recta a partir de un punto, se crea un eje, y se arrastra este punto a una distancia finita del anterior, el camino formado por la extrusión es un segmento de recta (Figura 2.17). Así se crea la primera dimensión, representada por X_1.

Figura 2.17: Extrusión ortogonal del punto para formar una recta.

Si la extrusión es partiendo del origen hacia una distancia de una unidad, se pueden definir los vértices de una línea recta unidimensional (1D) como muestra la Tabla 2.1. Y las coordenadas de los vértices se pueden representar con (1) bit.

X_1	*Representación binaria*	*Representación decimal*
0	0	0
1	1	1

Tabla 2.1: Vértices del segmento de recta.

El cuadrado se puede obtener de manera análoga, creando un nuevo eje perpendicular a donde se encuentra el segmento de recta, y desplazándola sobre este eje a una distancia igual a su longitud, el camino descrito por la extrusión genera un cuadrado 2D, el cual tiene 4 vértices y 4 aristas (Figura 2.18). El nuevo eje representado por X_2 es la segunda dimensión.

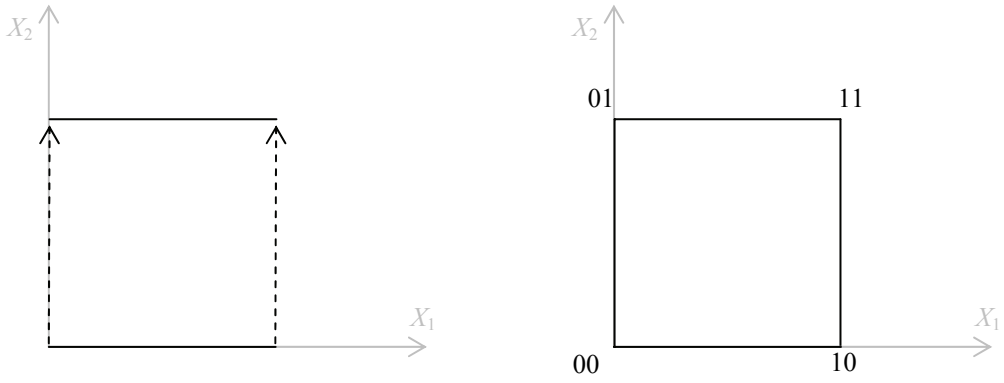

Figura 2.18: Extrusión ortogonal de un segmento de recta para formar un cubo.

Si uno de los vértices del cuadrado se posiciona en el origen y los vértices se extienden una unidad hacia el sentido positivo de los ejes, se pueden definir los vértices del cuadrado como muestra la Tabla 2.2. Y las coordenadas de los vértices se pueden representar con (2) bits.

X_1	X_2	*Representación binaria*	*Representación decimal*
0	0	00	0
0	1	01	1
1	0	10	2
1	1	11	3

Tabla 2.2: Vértices del cuadrado.

Para obtener el cubo a partir del cuadrado, se desplaza el cuadrado hacia un eje perpendicular a su plano de soporte, a una distancia igual a la medida de uno de sus lados. El camino formado por la extrusión produce ahora un cubo 3D, el cual tiene 8 vértices, 12 aristas, y 6 caras (Figura 2.19). El nuevo eje representado por X_3 es la tercera dimensión.

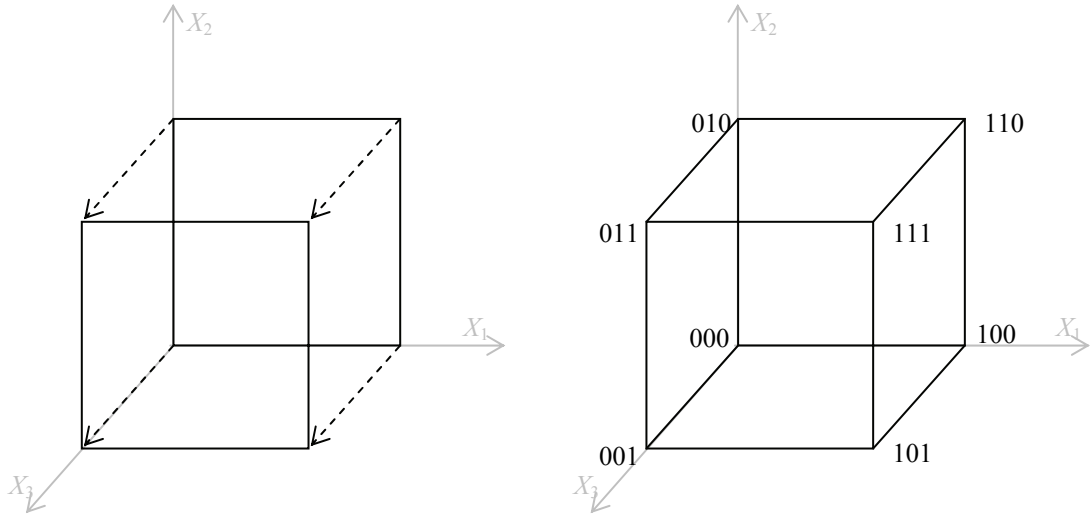

Figura 2.19: Extrusión ortogonal de un cuadrado para formar un cubo.

Observación 2.1:

Dado que este documento está siendo visualizado en un plano 2D (ya sea en papel o en una pantalla), la representación ortogonal del tercer eje en la Figura 2.19 es una proyección, sin embargo, deberá ser interpretada como una línea perpendicular a los otros dos. Además, es obvio el echo de que éste nuevo eje no se puede poner perpendicularmente a los otros dos en un plano, y esto no implica que el cubo o la tercera dimensión no exista.

De la misma forma que el cuadrado, si uno de los vértices del cubo se posiciona en el origen y los vértices se extienden una unidad hacia el sentido positivo de los ejes, se puede definir los vértices del cubo como muestra la Tabla 2.3. Y las coordenadas de los vértices se pueden representar con (3) bits.

X_1	X_2	X_3	*Representación binaria*	*Representación decimal*
0	0	0	000	0
0	0	1	001	1
0	1	0	010	2
0	1	1	011	3
1	0	0	100	4
1	0	1	101	5
1	1	0	110	6
1	1	1	111	7

Tabla 2.3: Vértices del cubo.

Siguiendo esta analogía dimensional, se puede obtener el hipercubo 4D a partir del cubo, desplazándolo hacia un eje perpendicular a su espacio de soporte, a una distancia igual a la medida de uno de sus lados. El camino formado por la extrusión produce un hipercubo 4D, el cuál tiene 16 vértices, 32 aristas, 24 caras, y 8 volúmenes [Pérez-Águila 03] (Figura 2.20). El nuevo eje representado por X_4 es la cuarta dimensión.

Observación 2.2:

Similar al caso anterior, dado que este documento está siendo visualizado en un plano 2D, la representación ortogonal del cuarto eje en la Figura 2.20 es una proyección de 4D a 2D, sin embargo, deberá ser interpretada como una línea perpendicular a las otras tres que forman el espacio 3D. Entonces el hecho de que un cuarto eje no se pueda poner perpendicularmente a los otros tres en 3D (o en su proyección en 2D), no implica que el hipercubo o la cuarta dimensión no exista.

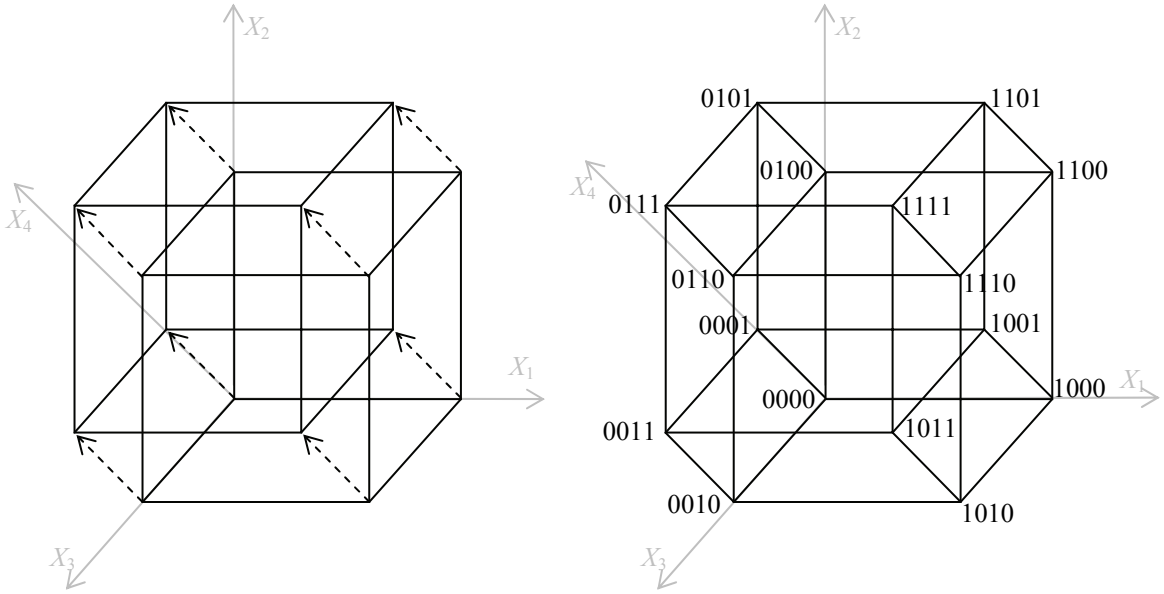

Figura 2.20: Extrusión ortogonal de un cubo para formar un hipercubo 4D.

Nuevamente, si uno de los vértices del hipercubo se posiciona en el origen y los vértices se extienden una unidad hacia el sentido positivo de los ejes, se pueden definir los vértices del hipercubo 4D como muestra la Tabla 2.4. Y las coordenadas de los vértices se pueden representar con (4) bits.

X_1	X_2	X_3	X_4	*Representación binaria*	*Representación hexadecimal*
0	0	0	0	0000	0
0	0	0	1	0001	1
0	0	1	0	0010	2
0	0	1	1	0011	3
0	1	0	0	0100	4
0	1	0	1	0101	5
0	1	1	0	0110	6
0	1	1	1	0111	7
1	0	0	0	1000	8
1	0	0	1	1001	9
1	0	1	0	1010	A
1	0	1	1	1011	B
1	1	0	0	1100	C
1	1	0	1	1101	D
1	1	1	0	1110	E
1	1	1	1	1111	F

Tabla 2.4: Vértices del hipercubo 4D.

Se observa que la Tabla 2.1 con los vértices del hipercubo 1D, tiene la representación de todos los números enteros positivos de 1 cifra binaria. La Tabla 2.2 con los vértices del hipercubo 2D, tiene la representación de todos los números de 2 cifras binarias. La Tabla 2.3 con los vértices del hipercubo 3D, tiene la representación de todos los números de 3 cifras binarias. Y la Tabla 2.4 con los vértices del hipercubo 4D, tiene la representación de todos los números de 4 cifras binarias. De esta forma, se pueden definir los vértices de un hipercubo *n*D con la creación de la tabla de la representación binaria de los números enteros positivos formados por *n* cifras binarias, es decir los números del 0 al 2^n-1 (ver Tabla 2.5). Y las coordenadas de los vértices se pueden representar con (*n*) bits.

X_1	X_2	…	X_{n-1}	X_n	***Representación binaria***	***Representación Decimal***
0	0	…*0*…	0	0	00…00	0
0	0	…*0*…	0	1	00…01	1
⋮	⋮	⋱	⋮	⋮	⋮	⋮
1	1	…*1*…	0	1	11…01	2^n-3
1	1	…*1*…	1	0	11…10	2^n-2
1	1	…*1*…	1	1	11…11	2^n-1

Tabla 2.5: Vértices del hipercubo *n*D.

2.2.2 Simplex

En [Coxeter 63] se muestra un método para obtener la familia de politopos llamados simplex. Se comienza por el espacio 0D, donde al igual que en el caso del hipercubo 0D, el único objeto cero-dimensional que existe es el punto (Figura 2.21).

•

Figura 2.21: Simplex 0D (un punto).

Después se crea un eje X_1 y se coloca un nuevo punto a una distancia finita del anterior, la unión de este punto con el origen forma un segmento de recta 1D. (Figura 2.22).

Figura 2.22: Simplex 1D (segmento de recta).

Si un punto se encuentra sobre el origen, y el otro a una distancia de una unidad, se pueden definir los vértices del simplex 1D como muestra la Tabla 2.6. Y las coordenadas de los vértices se pueden representar con (1) bit.

X_1	*Representación binaria*	*Representación decimal*
0	0	0
1	1	1

Tabla 2.6: Vértices del simplex 1D (segmento de recta).

Esta definición de vértices coincide con el caso del hipercubo 1D, pero aquí en la primera dimensión terminan las similitudes.

En 2D se debe seleccionar un punto fuera del eje donde se encuentra el segmento de recta, este nuevo punto se une a los vértices del simplex 1D, esto genera un triángulo 2D, el cuál tiene 3 vértices y 3 aristas. (Figura 2.23). Aunque la siguiente figura para considerarse regular debería ser un triángulo equilátero, ha sido modificada para facilitar la generación de los vértices y la evolución del politopo simplex a otras dimensiones. Por tanto el politopo simplex que se generará para las simulaciones, no será regular.

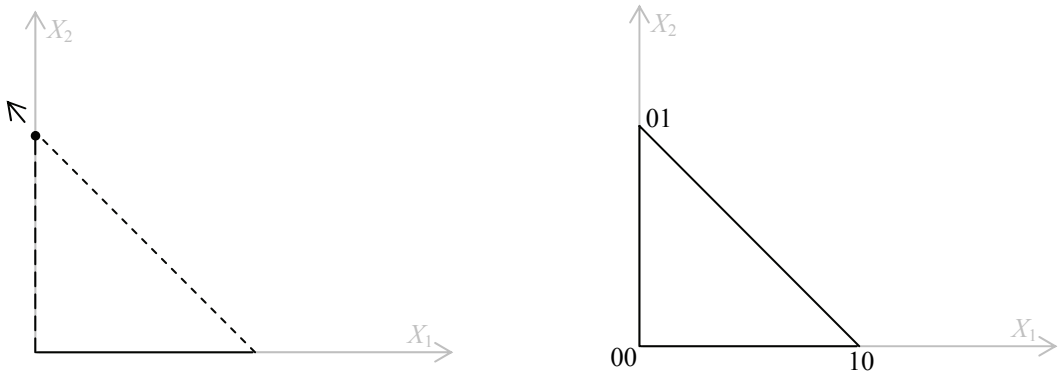

Figura 2.23: Simplex 2D (triángulo).

Para este caso particular de triángulo 2D está representado por los vértices que muestra la Tabla 2.7. Y las coordenadas de los vértices se pueden representar con (2) bits.

X_1	X_2	*Representación binaria*	*Representación decimal*
0	0	00	0
0	1	01	1
1	0	10	2

Tabla 2.7: Vértices simplex 2D (triángulo).

Para 3D, nuevamente se selecciona un punto fuera del plano de soporte del triángulo, este nuevo punto se une a todos los vértices del simplex 2D, esto genera un tetraedro 3D, el cuál tiene 4 vértices, 6 aristas y 4 caras triangulares (Figura 2.24). De nueva cuenta, el tetraedro trazado no es regular.

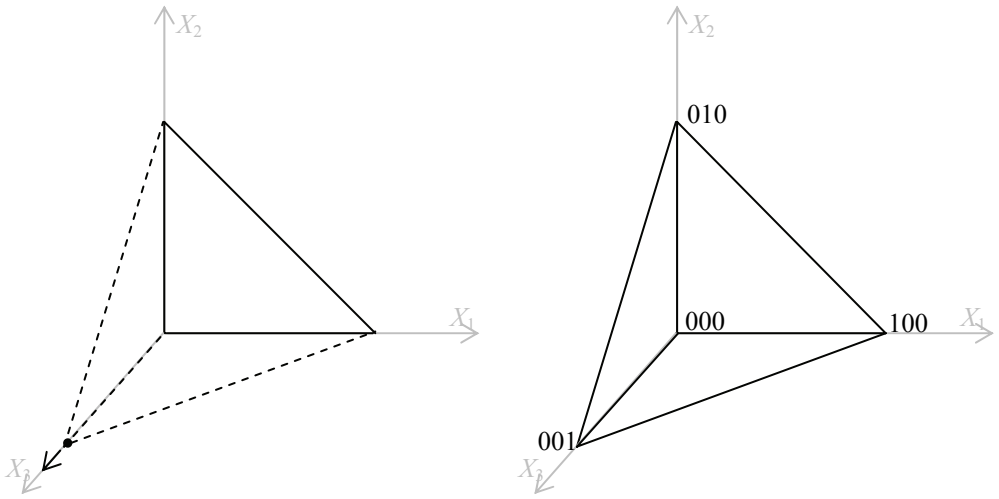

Figura 2.24: Simplex 3D (tetraedro).

Para este caso particular de tetraedro 3D, los vértices están representados como muestra la Tabla 2.8. Y las coordenadas de los vértices se pueden representar con (3) bits.

X_1	X_2	X_3	*Representación binaria*	*Representación Decimal*
0	0	0	000	0
0	0	1	001	1
0	1	0	010	2
1	0	0	100	4

Tabla 2.8: Vértices del simplex 3D (tetraedro).

Siguiendo esta analogía dimensional, se puede obtener el simplex 4D a partir del tetraedro, seleccionando nuevamente un punto fuera del volumen de soporte del tetraedro, este nuevo punto se une a todos los vértices del simplex 3D, esto genera el simplex 4D, el cuál tiene 5 vértices, 10 aristas, 10 caras, y 5 volúmenes [Pérez-Águila 03] (Figura 2.25). Nuevamente, el simplex 4D formado no es regular.

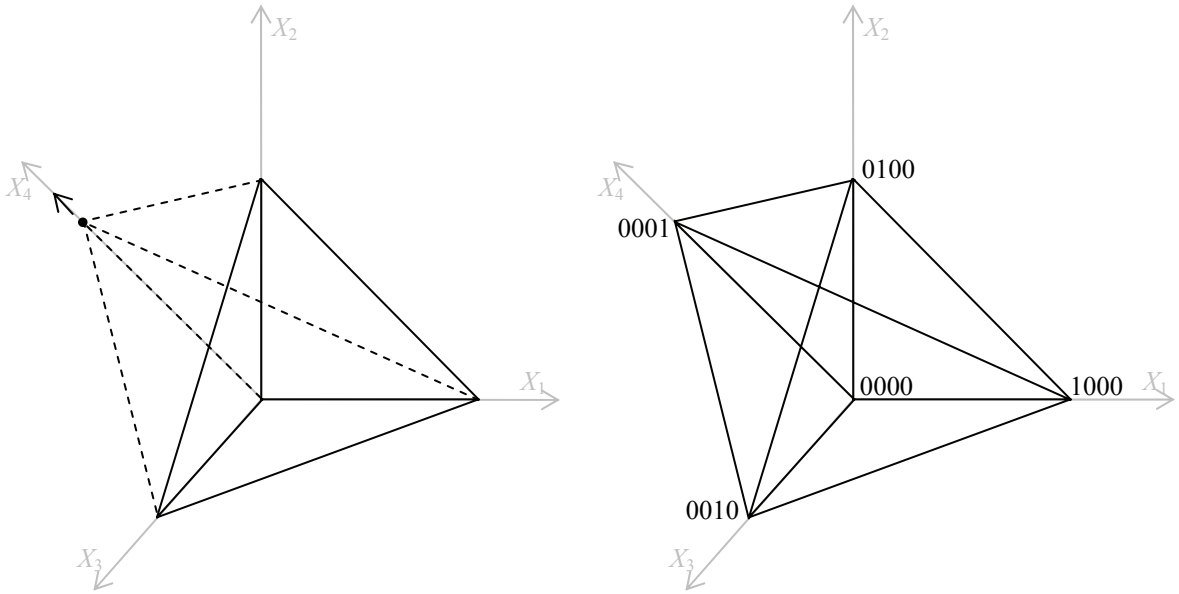

Figura 2.25: Simplex 4D.

Para este caso particular de tetraedro 3D, los vértices están representados como muestra la Tabla 2.9. Y las coordenadas de los vértices se pueden representar con (4) bits.

X_1	X_2	X_3	X_4	*Representación binaria*	*Representación decimal*
0	0	0	0	0000	0
0	0	0	1	0001	1
0	0	1	0	0010	2
0	1	0	0	0100	4
1	0	0	0	1000	8

Tabla 2.9: Vértices del simplex 4D.

Se observa que la Tabla 2.6 con los vértices del simplex 1D tiene la representación binaria de los números: (0, 1). La Tabla 2.7 con los vértices del simplex 2D, tiene la

representación de los números (0, 1, 2). La Tabla 2.8 con los vértices del simplex 3D, tiene la representación de los números de (0, 1, 2, 4). Y la Tabla 2.9 con los vértices del simplex 4D, tiene la representación de los números de (0, 1, 2, 4, 8). De esta forma, se pueden definir los vértices de un simplex *n*D (recordando que no es regular) con la creación de la tabla con la representación binaria del cero y todos los valores de 2^k, donde $(0 \le k \le n-1)$ (ver Tabla 2.10). Y las coordenadas de los vértices se pueden representar con (*n*) bits.

X_1	X_2	…	X_{n-1}	X_n	***Representación binaria***	***Representación Decimal***
0	0	…*0*…	0	0	00…00	0
0	0	…*0*…	0	1	00…01	2^0=1
0	0	…*0*…	1	0	00…10	2^1=2
⋮	⋮	⋱	⋮	⋮	⋮	⋮
0	1	…*0*…	0	0	01…00	2^{n-2}
1	0	…*0*…	0	0	10…00	2^{n-1}

Tabla 2.10: Vértices del simplex *n*D.

2.2.3 Politopo Cruz

Utilizando la metodología presentada en [Coxeter 63] se hará la construcción de la familia de politopos llamados politopos cruz. Al igual que en las metodologías anteriores, se comienza por el espacio 0D, con el punto (Figura 2.26).

Figura 2.26: Politopo cruz 0D (un punto).

Después de este punto, se crea el eje X_1, y se generan otros dos nuevos puntos en el origen, y se desplazan en direcciones opuestas a una distancia finita sobre el eje creado, estos dos puntos se unen entre si, formando un segmento de recta 1D (Figura 2.27).

Figura 2.27: Politopo cruz 1D (segmento de recta).

Si el desplazamiento de cada punto se hace de una unidad, se pueden definir los vértices de una línea recta unidimensional (1D) como muestra la Tabla 2.11.

No. de vértice	X_1
0	-1
1	1

Tabla 2.11: Vértices del politopo cruz 1D (segmento de recta).

El politopo cruz 1D coincide en forma con el hipercubo 1D y el simplex 1D, pero no geométricamente en esta definición, pues difieren en la coordenada de uno de sus vértices (el -1).

Ahora, en 2D se crea el eje X_2, se generan otros dos nuevos puntos en el origen, y se desplazan en direcciones opuestas sobre el eje creado, estos puntos se unen a los vértices de la recta 1D para formar el politopo cruz 2D, un cuadrado, el cuál tiene 4 vértices, y 4 aristas (Figura 2.28).

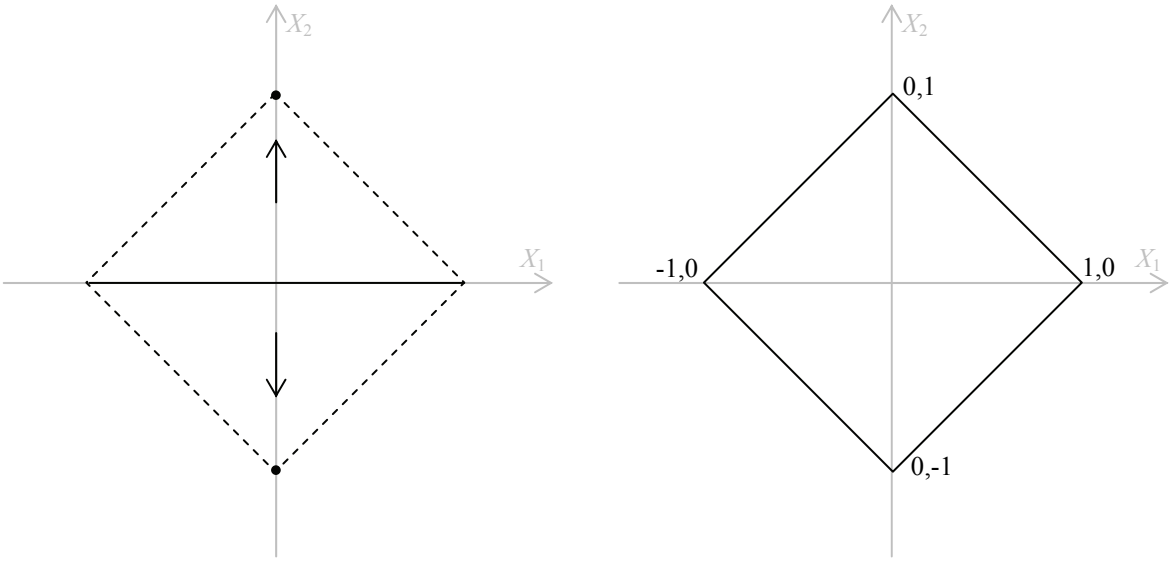

Figura 2.28: Politopo cruz 2D (cuadrado).

De esta forma, el politopo cruz 2D está representado por los vértices que se muestran en la Tabla 2.12.

No. de vértice	X_1	X_2
0	-1	0
1	1	0
2	0	-1
3	0	1

Tabla 2.12: Vértices del politopo cruz 2D (cuadrado).

Aunque en esta definición, el politopo cruz 2D difiere con el simplex 2D, coincide con el hipercubo 2D en forma, pero no geométricamente. Y es aquí, en la segunda dimensión, donde terminan las similitudes entre estos politopos.

Para 3D se crea el eje X_3, se generan otros dos puntos en el origen, y se desplazan en direcciones opuestas sobre el eje creado, estos puntos se unen a los vértices de politopo cruz 2D para formar el politopo cruz 3D, un octaedro, el cuál tiene 6 vértices, 12 aristas y 8 caras triangulares (Figura 2.29).

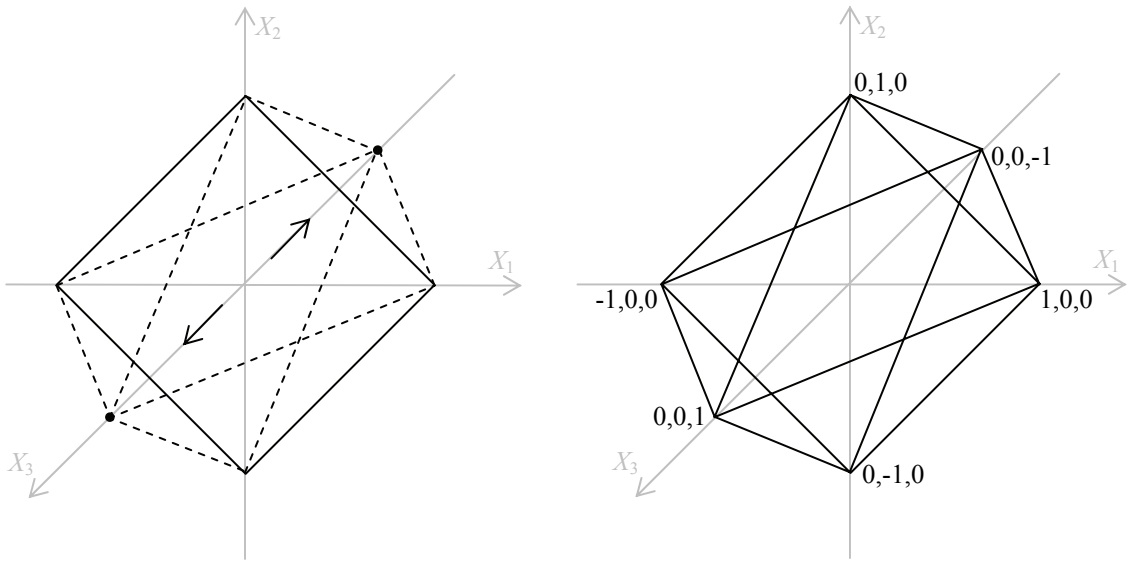

Figura 2.29: Politopo cruz 3D (octaedro).

De esta forma, el politopo cruz 3D está representado por los siguientes vértices que se muestran en la Tabla 2.13.

No. de vértice	X_1	X_2	X_3
0	-1	0	0
1	1	0	0
2	0	-1	0
3	0	1	0
4	0	0	-1
5	0	0	1

Tabla 2.13: Vértices del politopo cruz 3D (octaedro).

Siguiendo esta analogía dimensional, se puede obtener el politopo cruz 4D a partir del octaedro, se crea el eje X_4 y se generan dos nuevos puntos en el origen y se desplazan en direcciones opuestas sobre el eje creado, estos puntos se unen cada uno de los vértices de politopo cruz 3D para formar el politopo cruz 4D, el cuál tiene 8 vértices, 24 aristas y 32 caras y 16 volúmenes [Pérez-Águila 03] (Figura 2.30).

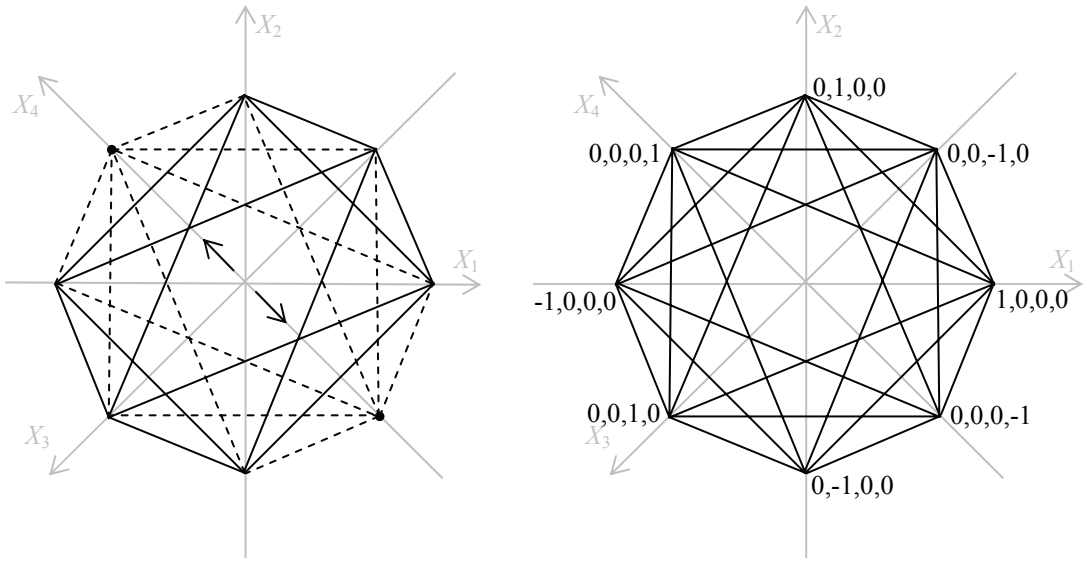

Figura 2.30: Politopo cruz 4D.

De esta forma, los vértices del politopo cruz 4D quedan como muestra la Tabla 2.14:

No. de vértice	X_1	X_2	X_3	X_4
0	-1	0	0	0
1	1	0	0	0
2	0	-1	0	0
3	0	1	0	0
4	0	0	-1	0
5	0	0	1	0
6	0	0	0	-1
7	0	0	0	1

Tabla 2.14: Vértices del politopo cruz 4D.

Se observa que la Tabla 2.11 con los vértices del politopo cruz 1D, tiene la representación de la permutación de (±1). La Tabla 2.12 con los vértices del politopo cruz 2D, tiene la representación de la permutación de (±1, 0). La Tabla 2.13 con los vértices del politopo cruz 3D, la representación de la permutación de (±1, 0, 0). Y la Tabla 2.14 con los vértices del politopo cruz 4D, tiene la representación de la permutación de (±1, 0, 0, 0). De esta forma, se pueden definir los vértices de un politopo cruz *n*D con la creación de la tabla que contenga la permutación de (+1, 0, …, 0), con *n* dígitos (ver Tabla 2.15).Y las coordenadas de los vértices se pueden representar con (*n*) bits.

No. de vértice	X_1	X_2	…	X_{n-1}	X_n
0	-1	0	…*0*…	0	0
1	1	0	…*0*…	0	0
2	0	-1	…*0*…	0	0
3	0	1	…*0*…	0	0
⋮	⋮	⋮	⋱	⋮	⋮
2*n*-4	0	0	…*0*…	-1	0
2*n*-3	0	0	…*0*…	1	0
2*n*-2	0	0	…*0*…	0	-1
2*n*-1	0	0	…*0*…	0	1

Tabla 2.15: Vértices del politopo cruz *n*D.

2.3 Trazo Automático de Politopos

Esta parte es la primera aportación de la presente investigación. En los temas anteriores se hizo un repaso de las analogías dimensionales que se utilizan para la creación de politopos de dimensiones mayores a 4D.

Esta información será de utilidad para proponer un método que permita generar de forma automática los vértices y el orden en que tienen que ser trazados para crear las aristas y tener finalmente el modelo de alambres de los politopos regulares *n*D.

Como se observó anteriormente, todos los politopos se derivan del punto 0D, pero para fines de representación y propuesta de la solución del trazo automático, se comienza desde el espacio 2D.

2.3.1 Trazo del Hipercubo

Un cuadrado 2D está representado por los siguientes vértices:

X_1	X_2	*Representación binaria*	*Representación decimal*
0	0	00	0
0	1	01	1
1	0	10	2
1	1	11	3

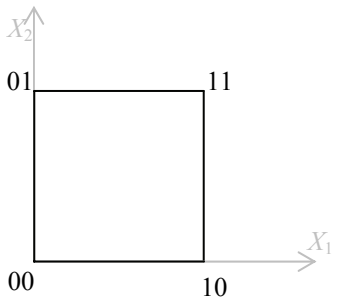

Figura 2.31: El hipercubo 2D (cuadrado) y sus vértices.

De aquí se puede visualizar que las aristas están dadas por los siguientes pares de vértices en su representación decimal: (0, 1), (0, 2), (1, 3), (2, 3).

La observación geométrica que ayuda a realizar el trazo de este cuadrado es la siguiente: Dado un vértice, este formará aristas hacía otros vértices que tendrán que ser adyacentes al primero, es decir, hacia donde hay solamente un cambio en alguno de sus

coordenadas, lo que formará un ángulo recto. Si se observan estos vértices en su representación binaria, significa que la arista se forma hacia aquellos vértices donde sólo un valor de 0 cambie a 1. Por ejemplo para el vértice 0= (0,0), las aristas que se forman son aquellas donde solo hay un cambio en alguno de sus valores en 0, entonces el punto 0= (0,0) forma aristas con los vértices: 1= (0,**1**) y 2= (**1**,0).

Se toma ahora el vértice 1= (0, 1) y si se hace la misma operación, se observa que éste forma aristas sólo con el vértice 3= (**1**, 1). El vértice 2= (1,0) forma una arista con el vértice 3= (**1**,1). El vértice 3 no contiene 0's.

No se consideran los cambios de 1 a 0, porque esto representaría una arista ya contemplada, solo que el trazo sería en sentido inverso.

Entonces siguiendo esta idea para el trazo del cuadrado 2D, las aristas se forman con los pares de vértices de la Tabla 2.16:

0= (0,0) → 1= (0,1)	1= (0,1) → 3= (1,1)
0= (0,0) → 2= (1,0)	2= (1,0) → 3= (1,1)

Tabla 2.16: Las cuatro aristas del cuadrado.

Un cubo 3D está representado por los siguientes vértices.

X_1	X_2	X_3	*Representación binaria*	*Representación decimal*
0	0	0	000	0
0	0	1	001	1
0	1	0	010	2
0	1	1	011	3
1	0	0	100	4
1	0	1	101	5
1	1	0	110	6
1	1	1	111	7

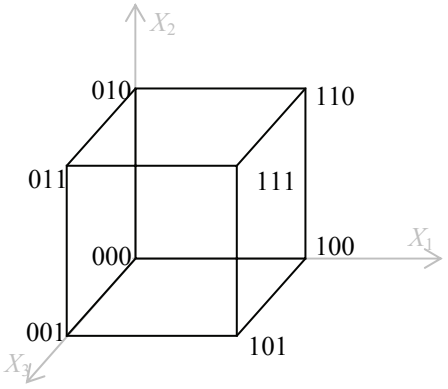

Figura 2.32: El hipercubo 3D (cubo) y sus vértices.

Siguiendo la idea anterior, se toma nuevamente el vértice 0 = (0,0,0), las aristas que se forman son aquellas donde solo hay un cambio en alguno de sus valores en 0, entonces el punto (0,0,0) forma aristas con los vértices: 1= (0,0,**1**), 2= (0,**1**,0) y 4= (**1**,0,0).

Ahora se toma el vértice 1 = (0,0,1), éste forma aristas con el vértice 3 = (0,**1**,1) y el 5= (**1**,0,1). El vértice 2= (0,1,0) forma aristas con 3= (0,1,**1**) y 6= (**1**,1,0), y si se continua con estas operaciones para el trazo del cubo 3D, se tiene que las aristas son las formadas por pares de vértices mostrados en la Tabla 2.17, que son las 12 aristas que forman al cubo.

0= (0,0,0) → 1= (0,0,1)	1= (0,0,1) → 5= (1,0,1)	4= (1,0,0) → 5= (1,0,1)
0= (0,0,0) → 2= (0,1,0)	2= (0,1,0) → 3= (0,1,1)	4= (1,0,0) → 6= (1,1,0)
0= (0,0,0) → 4= (1,0,0)	2= (0,1,0) → 6= (1,1,0)	5= (1,0,1) → 7= (1,1,1)
1= (0,0,1) → 3= (0,1,1)	3= (0,1,1) → 7= (1,1,1)	6= (1,1,0) → 7= (1,1,1)

Tabla 2.17: Las 12 aristas del cubo.

En su representación decimal, la forma en que se pueden obtener los vértices que forman aristas con un vértice dado es, tan solo sumar 2^k al valor del vértice, donde k es la posición donde se hace el cambio de 0 a 1.

Por ejemplo:

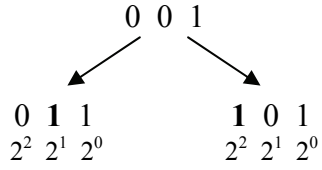

Figura 2.33: Ejemplo de cálculo de aristas para el cubo.

Se observa que el vértice 1 forma aristas con los vértices obtenidos de las operaciones: $(1+2^1) = \mathbf{3}$ y $(1+2^2) = \mathbf{5}$.

Algo importante notar aquí es que, los vértices del hipercubo están representados por todos aquellos números enteros desde el 0 a 2^n-1, donde n es la dimensión del politopo.

Con la idea anteriormente descrita, en general se puede definir un algoritmo para la generación de los vértices de un hipercubo *n*D conociendo su dimensión.

Las variables importantes del siguiente algoritmo son:

- *dimension* = Variable entera para indicar la dimensión del politopo a crear, sirve como parámetro para determinar la cantidad de dígitos del número binario.
- *numVertex* = Variable entera para el número de vértices de politopo a crear.
- *vertexChar* = Variable tipo cadena para almacenar el número binario del vértice.
- *vertexes[v][i]* = Arreglo de arreglos de enteros del vértice, donde *v* indica el número de vértice, e *i*, cada uno de los dígitos que forman el vértice.
- *edges[i]* = Arreglo de enteros para representar la secuencia en que deben ser trazados los vértices para el trazo de las aristas.

```
Procedure getEdges (dimension)
 numVertex = 2^dimension
 i=0;
 for v=0 to numVertex-2
       vertexChar=bin(v,dimension)
       vertexChar=reverse(vertex) //Invertir el número binario
       //Se buscan los lugares donde hay 0's
       for k=0 to dimension-1
             if vertexChar[k]=='0'
                   edges[i]=v
                   v2 = v+2^k
                   edges[i+1]=v2;
                   i=i+2;
                   vertexes[v][dimension-k]=0;
             endIf
             vertexes[v][dimension-k]=1;
       endFor
 endFor
endProcedure
```

La entrada del algoritmo es la *dimensión* del hipercubo, la salida es el arreglo *vertexes*, que contiene de forma secuencial, los números de los vértices que deben ser unidos para formas las aristas, es decir, el vértice en la posición *vertexes[0]* se une con el

vértice de la posición *vertexes[1]*, vertexes[2] con vertexes[3], y así sucesivamente, hasta unir los últimos 2 vértices del arreglo.

Para un hipercubo 4D, el algoritmo indica que el trazo de aristas deber ser de acuerdo a los pares de vértices de la Tabla 2.18:

0= (0,0,0,0)→1= (0,0,0,1)	3= (0,0,1,1)→B= (1,0,1,1)	8= (1,0,0,0)→C= (1,1,0,0)
0= (0,0,0,0)→2= (0,0,1,0)	4= (0,1,0,0)→5= (0,1,0,1)	9= (1,0,0,1)→B= (1,0,1,1)
0= (0,0,0,0)→4= (0,1,0,0)	4= (0,1,0,0)→6= (0,1,1,0)	9= (1,0,0,1)→D= (1,1,0,1)
0= (0,0,0,0)→8= (1,0,0,0)	4= (0,1,0,0)→C= (1,1,0,0)	A= (1,0,1,0)→B= (1,0,1,1)
1= (0,0,0,1)→3= (0,0,1,1)	5= (0,1,0,1)→7= (0,1,1,1)	A= (1,0,1,0)→E= (1,1,1,0)
1= (0,0,0,1)→5= (0,1,0,1)	5= (0,1,0,1)→D= (1,1,0,1)	B= (1,0,1,1)→F= (1,1,1,1)
1= (0,0,0,1)→9= (1,0,0,1)	6= (0,1,1,0)→7= (0,1,1,1)	C= (1,1,0,0)→D= (1,1,0,1)
2= (0,0,1,0)→3= (0,0,1,1)	6= (0,1,1,0)→E= (1,1,1,0)	C= (1,1,0,0)→E= (1,1,1,0)
2= (0,0,1,0)→6= (0,1,1,0)	7= (0,1,1,1)→F= (1,1,1,1)	D= (1,1,0,1)→F= (1,1,1,1)
2= (0,0,1,0)→A= (1,0,1,0)	8= (1,0,0,0)→9= (1,0,0,1)	E= (1,1,1,0)→F= (1,1,1,1)
3= (0,0,1,1)→7= (0,1,1,1)	8= (1,0,0,0)→A= (1,0,1,0)	

Tabla 2.18: Las 32 aristas del hipercubo 4D.

Estos pares de vértices representan las 32 aristas del hipercubo 4D, según la fórmula que calcula el número de aristas de un hipercubo *n*D: $n2^{n-1}$ [Pérez-Águila 03], donde para n= 4, se tiene que: $4\left(2^3\right)=32$.

2.3.2 Trazo del Simplex

Un simplex 2D (triángulo) está representado por los siguientes vértices:

X_1	X_2	*Representación binaria*	*Representación decimal*
0	0	00	0
0	1	01	1
1	0	10	2

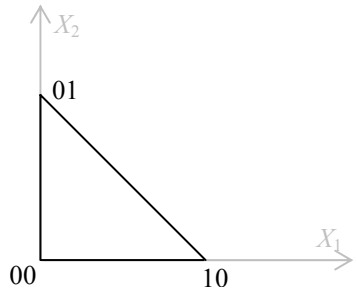

Figura 2.34: El simplex 2D (triángulo) y sus vértices.

La observación geométrica que ayuda en este caso para el trazo del triángulo es el siguiente: Cada vértice tendrá que ser unido con todos los demás, entonces se comienza

con el vértice 0= (0,0), el cuál se une con el resto, es decir con 1= (0,1) y 2= (1,0), después se toma el siguiente vértice 1= (0,1), el cual se tendría que unir con el 0= (0,0), pero se repetiría un trazo, así que solo se hace la unión hacia los vértices posteriores, por tanto 1= (0,1) se une con 2= (1,0), y con esto se tienen todos los trazos de las aristas, representados por los pares de vértices que se muestran en la Tabla 2.19.

0= (0,0) → 1= (0,1)	1= (0,1) → 2= (1,0)
0= (0,0) → 2= (1,0)	

Tabla 2.19: Las tres aristas del triángulo.

Un simplex 3D (tetraedro) está representado por los siguientes vértices

X_1	X_2	X_3	*Representación binaria*	*Representación Decimal*
0	0	0	000	0
0	0	1	001	1
0	1	0	010	2
1	0	0	100	4

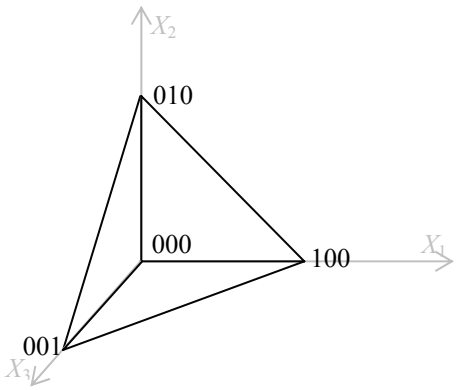

Figura 2.35: El simplex 3D (tetraedro) y sus vértices.

Se comienza nuevamente con el vértice 0= (0,0,0), y se une con el resto de vértices, 1= (0,0,1), 2= (0,1,0) y 4= (0,1,0), enseguida se toma el vértice 1= (0,0,1) y se une con 2= (0,1,0) y 4= (0,1,0), y finalmente el vértice 2= (0,1,0) se une con 4= (0,1,0). Así se tiene que las aristas del tetraedro son las formadas por los pares de vértices de la Tabla 2.20:

0= (0,0,0) → 1= (0,0,1)	1= (0,0,1) → 2= (0,1,0)
0= (0,0,0) → 2= (0,1,0)	1= (0,0,1) → 4= (1,0,0)
0= (0,0,0) → 4= (1,0,0)	2= (0,1,0) → 4= (1,0,0)

Tabla 2.20: Las seis aristas del tetraedro.

Algo importante notar aquí es que los vértices del simplex están representados por el cero y todos aquellos números enteros en su representación binaria iguales a 2^k, con $0 \leq k < n-1$, donde n es la dimensión del politopo.

Con esta idea, se puede definir en general, un algoritmo para la generación de los vértices del simplex nD.

Las variables importantes del siguiente algoritmo son:

- *dimension* = Variable entera para indicar la dimensión del politopo a crear, sirve como parámetro para determinar la cantidad de dígitos del número binario.
- *numVertex* = Variable entera para el número de vértices de politopo a crear.
- *vertexChar* = Variable tipo cadena para representar el número binario del vértice.
- *vertexes[v][i]* = Arreglo de arreglos de enteros del vértice, donde *v* indica el número de vértice, e *i*, cada uno de los dígitos que forman el vértice.
- *edges[i]* = Arreglo de enteros para representar la secuencia en que deben ser trazados los vértices para el trazo de las aristas.

```
Procedure getEdges (dimension)
 numVertex = dimension+1
 vertexes[0]= bin(0,dimension)
 for v=-1 to numVertex-2       //Creación de vértices
       if v==-1    //Caso especial del 0.
             vertexChar[v]=bin(0,dimension);
       else
             vertexChar[v]=bin(2^v,dimension)
       for k=0 to dimension-1
             if vertexChar[k]=='0'
             vertexes[v+1][k]=0
             else
                   vertexes[v+1][k]=1
 endfor
 //Creación de aristas
 i=0;
 for v=0 to numVertex-2
       for k=v+1 to numVertex-1
             edges[i]=v;
             edges[i+1]=k;
             i=i+2;
       endfor
endProcedure
```

La entrada del algoritmo es la *dimensión* del simplex, la salida es el arreglo *vertexes*, que contiene de forma secuencial, los números de los vértices que deben ser unidos para formas las aristas, que se unen de la misma forma que la salida del algoritmo del hipercubo.

Para un simplex 4D, el algoritmo indica que el trazo de aristas deber ser de acuerdo a los pares de vértices de la Tabla 2.21:

0= (0,0,0,0) → 1= (0,0,0,1)	1= (0,0,0,1) → 4= (0,1,0,0)
0= (0,0,0,0) → 2= (0,0,1,0)	1= (0,0,0,1) → 8= (1,0,0,0)
0= (0,0,0,0) → 4= (0,1,0,0)	2= (0,0,1,0) → 4= (0,1,0,0)
0= (0,0,0,0) → 8= (1,0,0,0)	2= (0,0,1,0) → 8= (1,0,0,0)
1= (0,0,0,1) → 2= (0,0,1,0)	4= (0,1,0,0) → 8= (1,0,0,0)

Tabla 2.21: Las 10 aristas del simplex 4D.

Estos pares de vértices representan las 10 aristas del simplex 4D, según la fórmula que calcula el número de aristas de un simplex *n*D: $C(n+1,2)$ [Pérez-Águila 03], donde para $n = 4$, se tiene que:

$$C\binom{4+1}{2} = \frac{5!}{2!(5-2)!} = \frac{120}{12} = 10$$

2.3.3 Trazo del Politopo Cruz.

Un politopo cruz 2D (cuadrado) está representado por los siguientes vértices:

No. de vértice	X_1	X_2
0	-1	0
1	1	0
2	0	-1
3	0	1

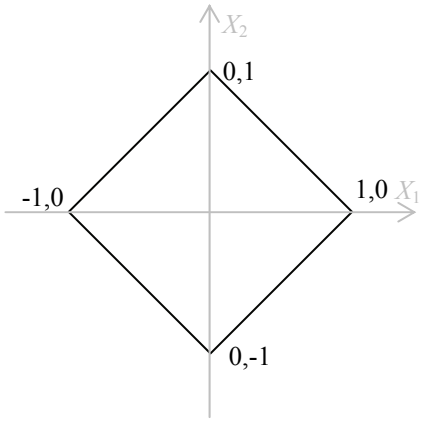

Figura 2.36: El politopo cruz 2D (cuadrado) y sus vértices.

La observación geométrica que ayuda en este caso para el trazo del politopo cruz 2D es el siguiente: De forma similar al caso de simplex, cada vértice tendrá que ser unido con el resto de los vértices posteriores (para no repetir aristas), excepto con el otro vértice sobre el mismo eje. Entonces comenzando en el vértice 0= (-1,0), se une con todos, excepto al vértice 1= (1,0) por ser el otro punto sobre el eje X_1, por tanto solo se crean aristas con los vértices 2= (0,-1) y 3= (0,1), después se toma el siguiente vértice 1= (1,0), que también se une a 2= (0,-1) y 3= (0,1), después se toma al vértice 2= (0,-1), pero el único vértice posterior que queda es el otro punto sobre el eje X_2, por tanto terminan los trazos de aristas, representados por los pares de vértices que se muestran en la Tabla 2.22.

0= (-1,0) → 2= (0,-1)	1= (1,0) → 2= (0,-1)
0= (-1,0) → 3= (0,1)	1= (1,0) → 3= (0,1)

Tabla 2.22: Las cuatro aristas del politopo cruz 2D (cuadrado).

Un politopo cruz 3D (octaedro) está representado por los siguientes vértices:

No. de vértice	X_1	X_2	X_3
0	-1	0	0
1	1	0	0
2	0	-1	0
3	0	1	0
4	0	0	-1
5	0	0	1

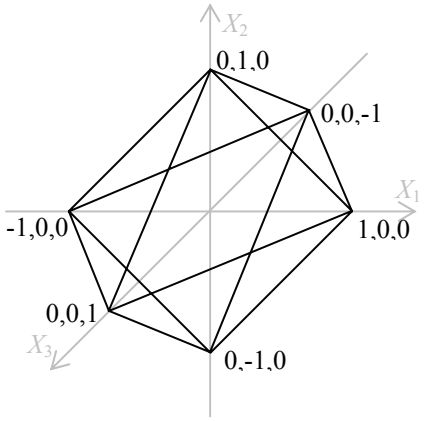

Figura 2.37: El politopo cruz 3D (octaedro) y sus vértices.

Utilizando la misma observación, se comienza con el vértice 0= (-1,0,0), se une con 2= (0,-1,0), 3= (0,1,0), 4= (0,0,-1) y 5=(0,0,1), después se toma el vértice 1= (1,0,0), que también se une a 2= (0,-1,0), 3= (0,1,0), 4= (0,0,-1) y 5= (0,0,1), enseguida se toma el vértice 2= (0,-1,0), que se une a 4= (0,0,-1) y 5=(0,0,1), y finalmente el vértice 3= (0,1,0) se

une a 4= (0,0,-1) y 5=(0,0,1), de esta forma los trazos de aristas quedan determinadas por la unión de los pares de vértices de la Tabla 2.23.

0= (-1,0,0) → 2= (0,-1,0)	1= (1,0,0) → 2= (0,-1,0)	2= (0,-1,0) → 4= (0,0,-1)
0= (-1,0,0) → 3= (0,1,0)	1= (1,0,0) → 3= (0,1,0)	2= (0,-1,0) → 5= (0,0,1)
0= (-1,0,0) → 4= (0,0,-1)	1= (1,0,0) → 4= (0,0,-1)	3= (0,1,0) → 4= (0,0,-1)
0= (-1,0,0) → 5= (0,0,1)	1= (1,0,0) → 5= (0,0,1)	3= (0,1,0) → 5= (0,0,1)

Tabla 2.23: Las 12 aristas del politopo cruz 3D (octaedro).

Algo importante notar aquí es que los vértices del politopo cruz están representados por la permutación de las coordenadas (±1, 0, . . . , 0), con *n* dígitos, donde *n* es la dimensión del politopo.

Con esta idea, se puede definir en general, un algoritmo para la generación de los vértices del politopo cruz *n*D.

Las variables importantes del siguiente algoritmo son:

- *dimension* = Variable entera para indicar la dimensión del politopo a crear.
- *numVertex* = Variable entera para el número de vértices de politopo a crear.
- *vertexes[v][i]* = Arreglo de arreglos de enteros del vértice, donde *v* indica el número de vértice, e *i*, cada uno de los dígitos que forman el vértice.
- *edges[i]* = Arreglo de enteros para representar la secuencia en que deben ser trazados los vértices para el trazo de las aristas.

La entrada del algoritmo es la *dimensión* del politopo cruz, la salida es el arreglo *vertexes*, que contiene de forma secuencial, los números de los vértices que deben ser unidos para formas las aristas, que se unen de la misma forma que la salida de los algoritmos para el hipercubo y el simplex.

```
//dimension, parámetro para el # de dig.
Procedure getEdges (dimension)
 numVertex = dimension*2
 //Creación de vértices
 v=0
 while v < numVertex-1
 fill(vertexes[v],0)//Llenar de 0's
       vertexes[v][int(v/2)]=-1 //vértice negativo
 fill(vertexes[v+1],0)//Llenar de 0's
 vertexes[v+1][int(v/2)]=1 //vértice positivo
 v=v+2
 endwhile
  //Creación de aristas
 i=0
 for v=0 to numVertex-3
 if odd(v+1)
       k=v+2
 else
       k=v+1
 for k=k to numVertex
       edges[i]=v
       edges[i+1]=k
       i=i+2;
       endfor
 endfor
endProcedure
```

Para un politopo cruz 4D, el algoritmo indica que el trazo de aristas deber ser de acuerdo a los pares de vértices de la Tabla 2.24:

0= (-1,0,0,0) → 2= (0,-1,0,0)	1= (1,0,0,0) → 4= (0,0,-1,0)	3= (0,1,0,0) → 4= (0,0,-1,0)
0= (-1,0,0,0) → 3= (0,1,0,0)	1= (1,0,0,0) → 5= (0,0,1,0)	3= (0,1,0,0) → 5= (0,0,1,0)
0= (-1,0,0,0) → 4= (0,0,-1,0)	1= (1,0,0,0) → 6= (0,0,0,-1)	3= (0,1,0,0) → 6= (0,0,0,1)
0= (-1,0,0,0) → 5= (0,0,1,0)	1= (1,0,0,0) → 7= (0,0,0,1)	3= (0,1,0,0) → 7= (0,0,0,-1)
0= (-1,0,0,0) → 6= (0,0,0,-1)	2= (0,-1,0,0) → 4= (0,0,-1,0)	4= (0,0,-1,0) → 6= (0,0,0,1)
0= (-1,0,0,0) → 7= (0,0,0,1)	2= (0,-1,0,0) → 5= (0,0,1,0)	4= (0,0,-1,0) → 7= (0,0,0,-1)
1= (1,0,0,0) → 2= (0,-1,0,0)	2= (0,-1,0,0) → 6= (0,0,0,1)	5= (0,0,1,0) → 6= (0,0,0,1)
1= (1,0,0,0) → 3= (0,1,0,0)	2= (0,-1,0,0) → 7= (0,0,0,-1)	5= (0,0,1,0) → 7= (0,0,0,-1)

Tabla 2.24: Las 24 aristas del politopo cruz 4D.

Estos pares de vértices representan las 24 aristas del politopo cruz 4D, según la fórmula que calcula el número de aristas de un politopo cruz *n*D: $4C(n,2)$ [Pérez-Águila 03], donde para *n*= 4, se tiene que:

$$4C\binom{4}{2} = 4\left(\frac{4!}{2!(4-2)!}\right) = 4\left(\frac{24}{4}\right) = 24$$

2.4 Resumen

En este capítulo, se analizaron las propiedades que deben cumplir los politopos para considerarse regulares, comenzando desde los polígonos en 2D, siguiendo con los poliedros en 3D y en general de los politopos regulares *n*D, donde se vio que a partir de 5D solo son posibles 3 de ellos: el hipercubo, el simplex y el politopo cruz.

Se analizó la representación topológica de estos tres politopos regulares y se propusieron los algoritmos para su trazo automático. La finalidad de esto, es poder generar automáticamente objetos *n*-dimensionales, a los cuales se les pueda aplicar las transformaciones geométricas *n*-dimensionales, analizadas en el siguiente capítulo, y poder obtener los efectos de tales transformaciones.