UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

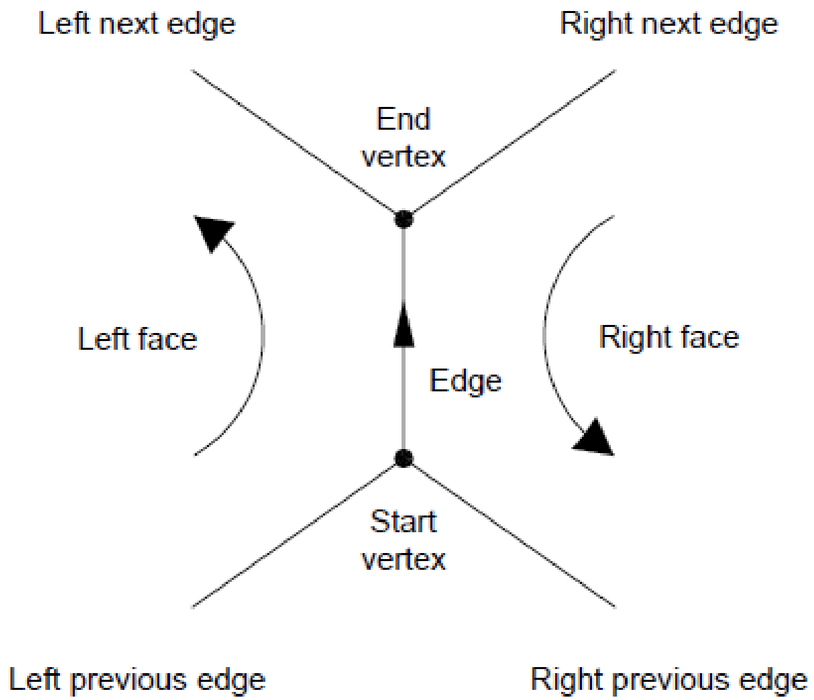
Sesión 9.0: Alas y celdas

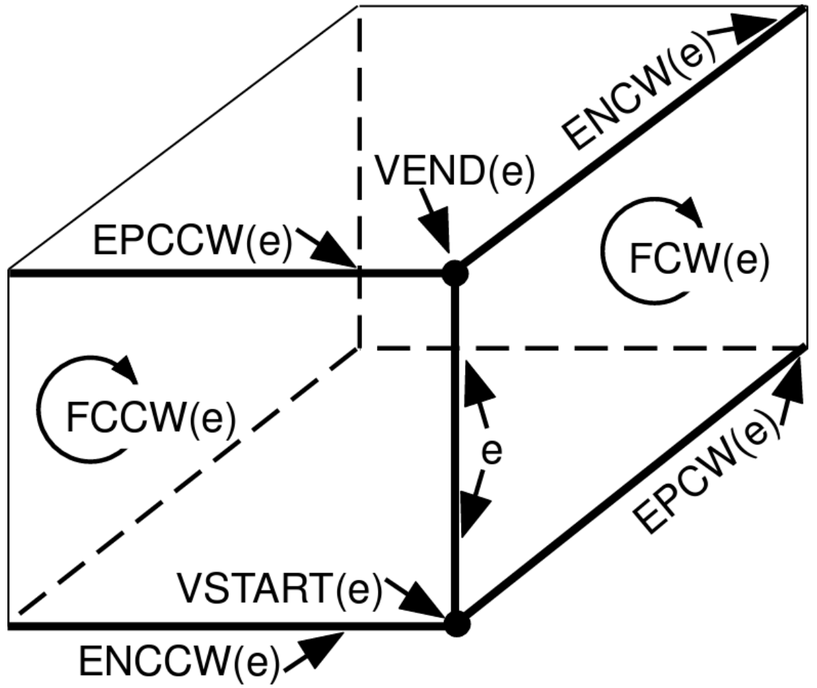
CS3102 EDA

Índice

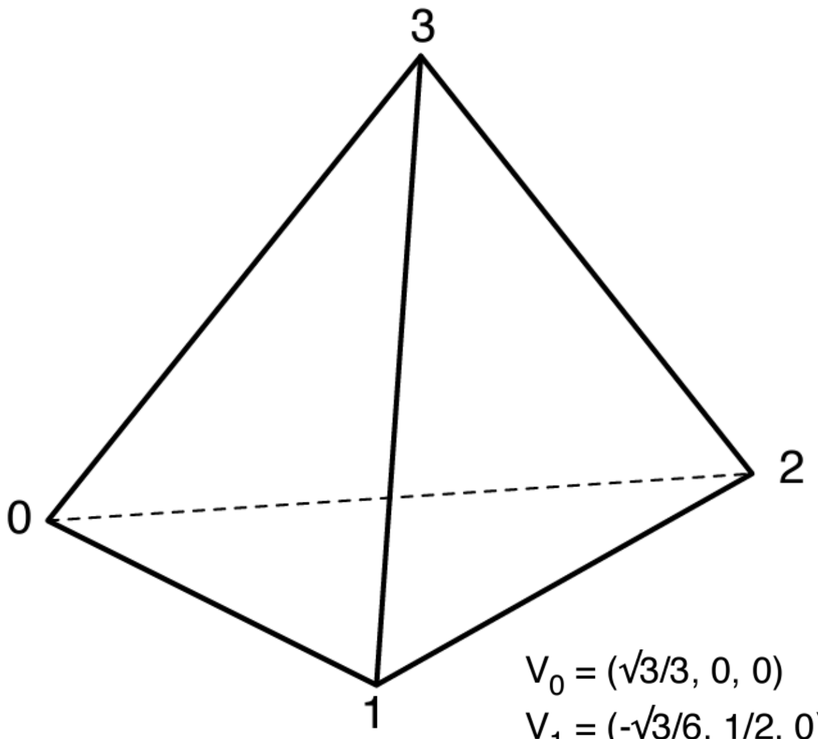
1. **Winged** Edges

Winged *Edges*



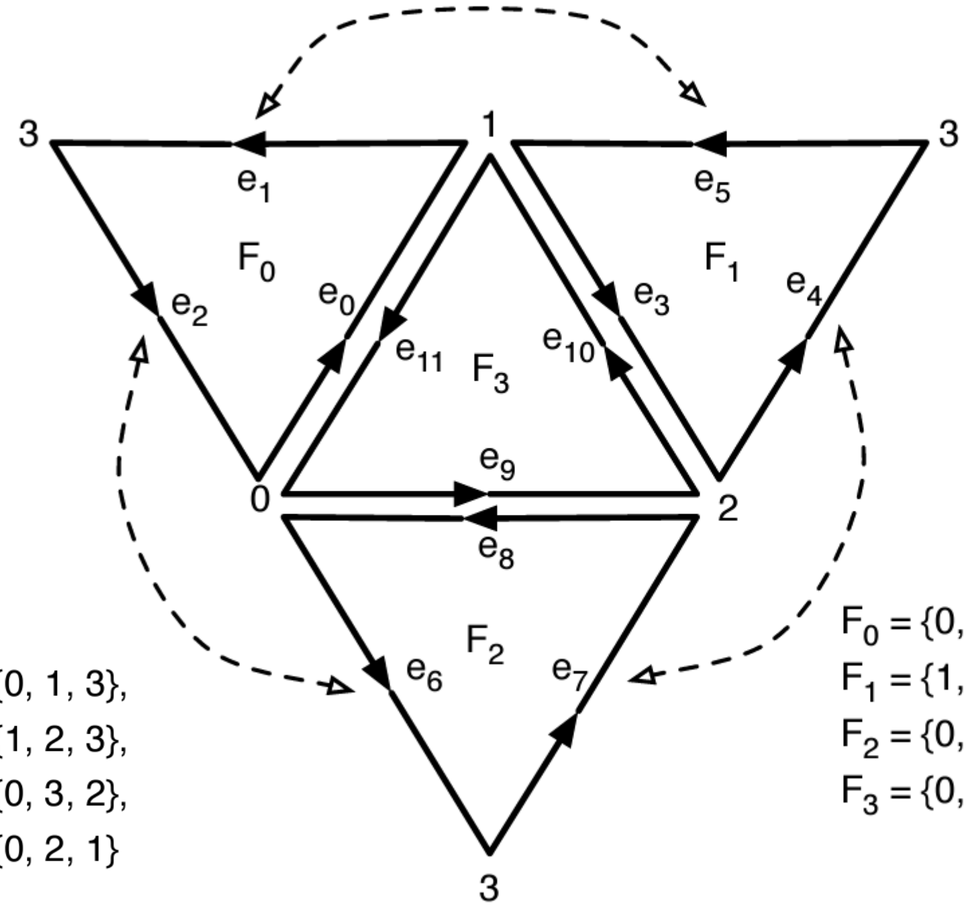


Winged *Edges*



$V_0 = (\sqrt{3}/3, 0, 0)$
$V_1 = (-\sqrt{3}/6, 1/2, 0)$
$V_2 = (-\sqrt{3}/6, -1/2, 0)$
$V_3 = (0, 0, \sqrt{6}/3)$

$F_0 = \{0, 1, 3\}$,
$F_1 = \{1, 2, 3\}$,
$F_2 = \{0, 3, 2\}$,
$F_3 = \{0, 2, 1\}$

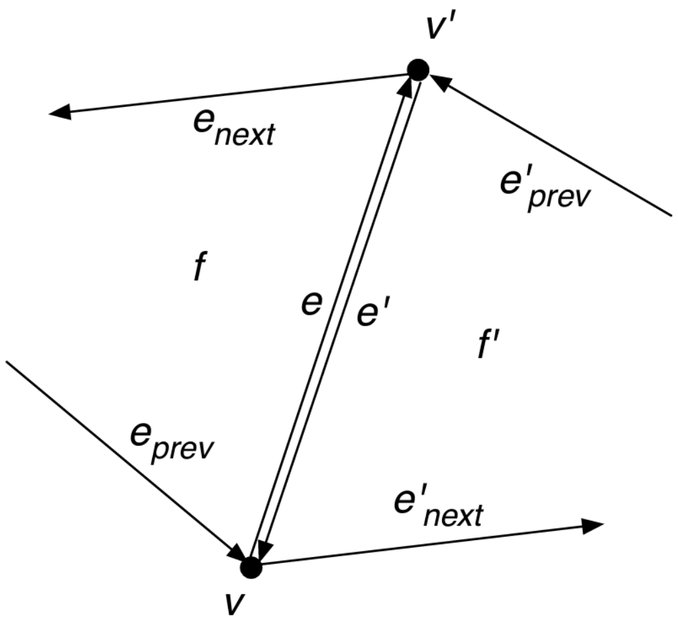


CCW orientation
from visible side

"winged edges"

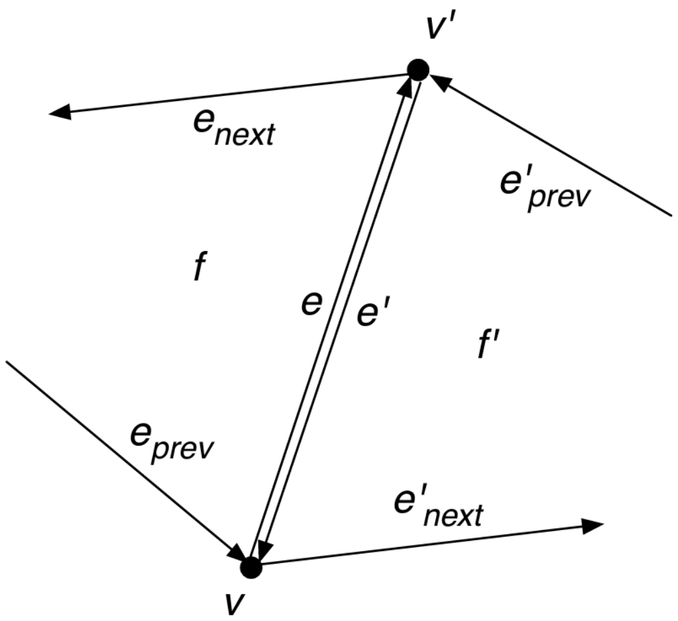
$F_0 = \{0, 1, 3\}$ 0→1, 1→3, 3→0
$F_1 = \{1, 2, 3\}$ 1→2, 2→3, 3→1
$F_2 = \{0, 3, 2\}$ 0→3, 3→2, 2→0
$F_3 = \{0, 2, 1\}$ 0→2, 2→1, 1→0

Winged *Edges*



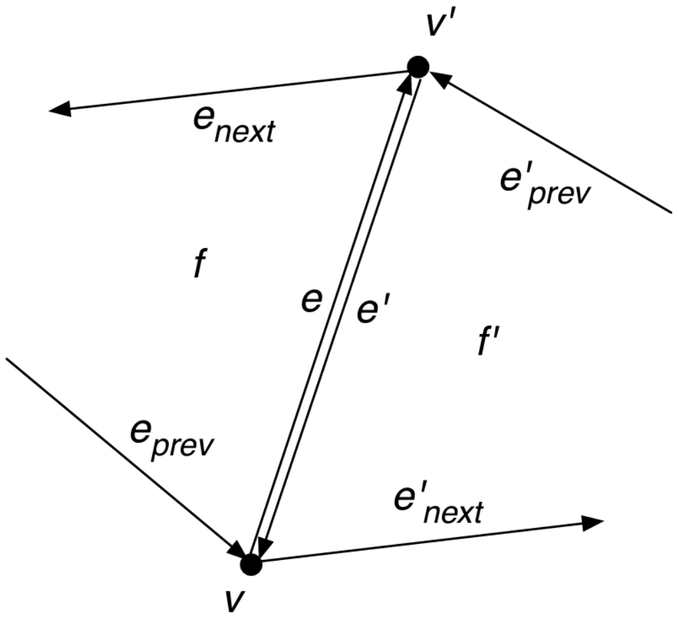

```
struct Edge {
  Vertex *v;
  Face *f;
  Edge *prev, *next;
  Edge *sym;
  ...
};
```

Winged *Edges*



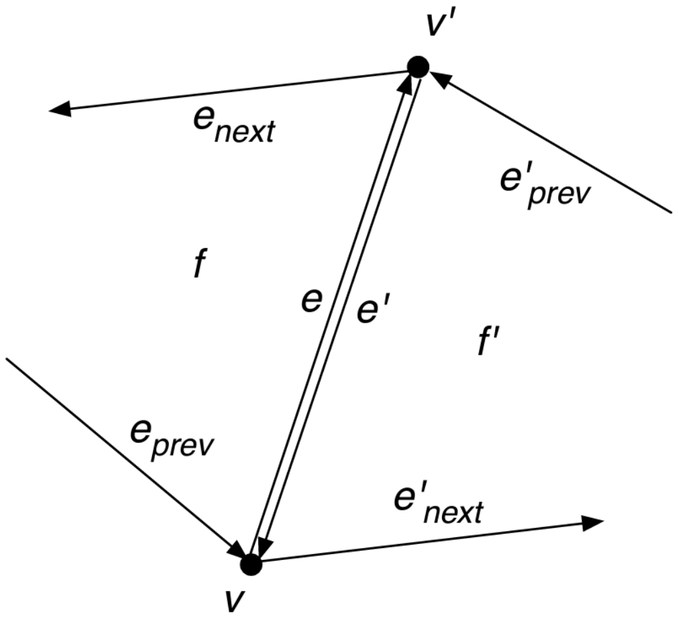

```
struct Edge {
  Vertex *v;
  Face *f;
  Edge *prev, *next;
  Edge *sym;
  ...
};

struct Vertex { Edge *e;...};
struct Face {Edge *e; ...};
```

Winged *Edges*



Iterar sobre las aristas de la cara f

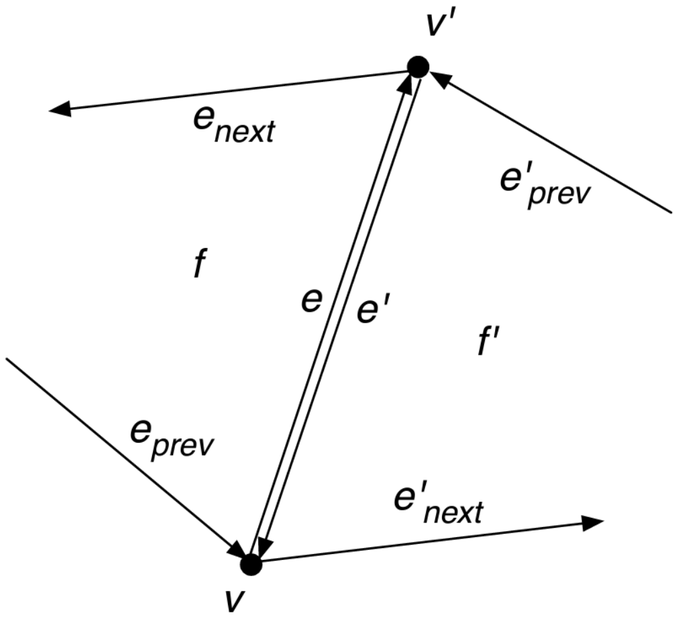
Winged *Edges*



Iterar sobre las aristas de la cara f

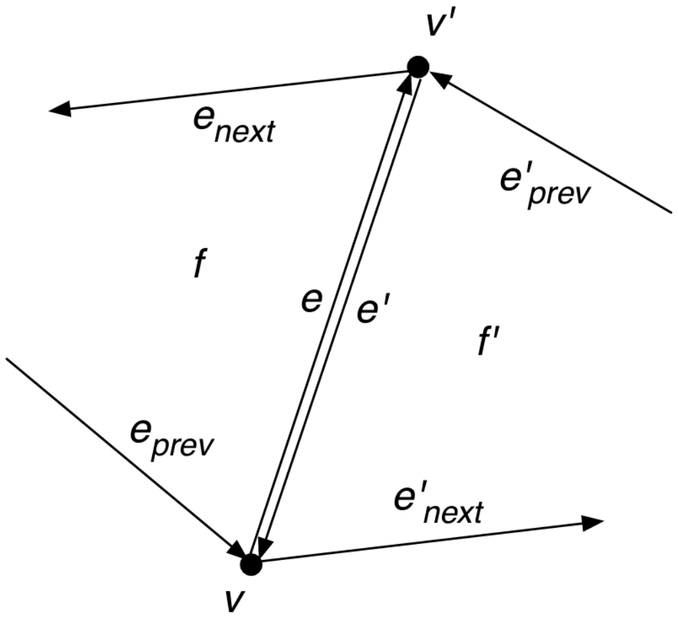
```
Edge *start = f->e;
Edge *e = start;
do {
  visit(e);
  e = e->next;  // CCW order
} while (e != start);
```

Winged *Edges*



Calcule la normal de la cara f

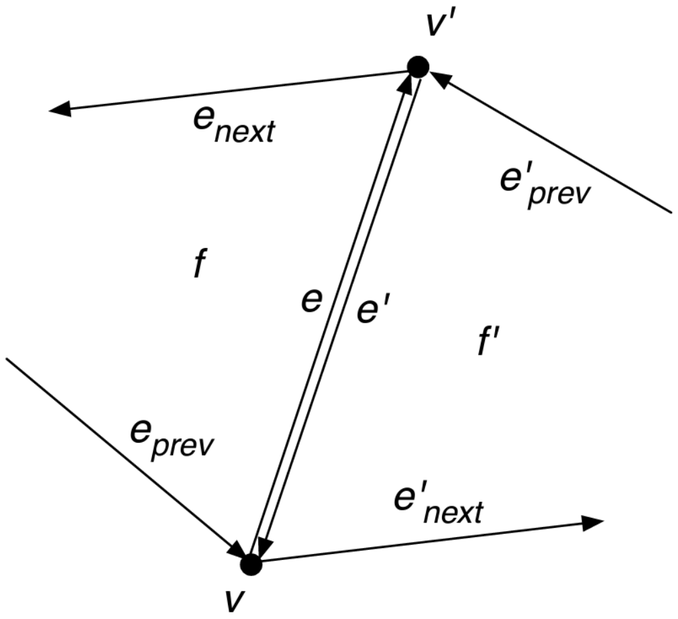
Winged *Edges*



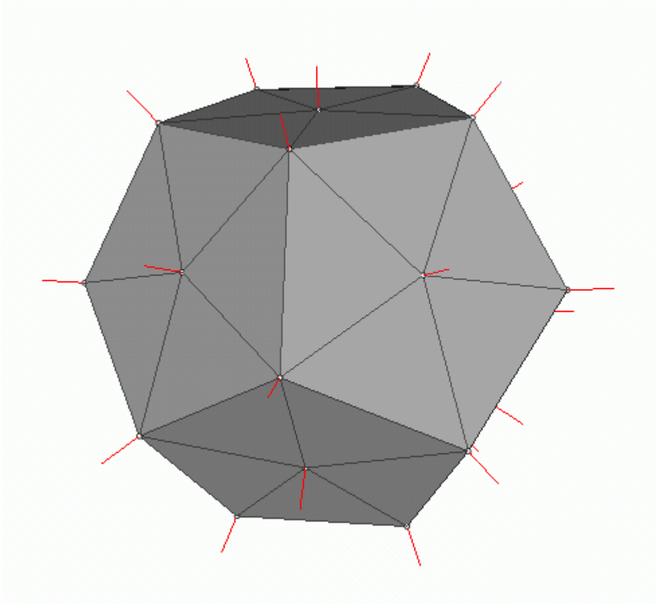
Calcule la normal de la cara f

```
f->norm = Vec3(0,0,0);
Edge *start = f->e;
Edge *e = start;
do {
   const Vec3& v = e->v->pos; // vertex position
   const Vec3& vnext = e->next->v->pos;
   f->norm.x += (v.y - vnext.y)*(v.z + vnext.z);
   f->norm.y += (v.z - vnext.z)*(v.x + vnext.x);
   f->norm.z += (v.x - vnext.x)*(v.y + vnext.y);
   e = e->next;
} while (e != start);
f->norm.normalize();
```

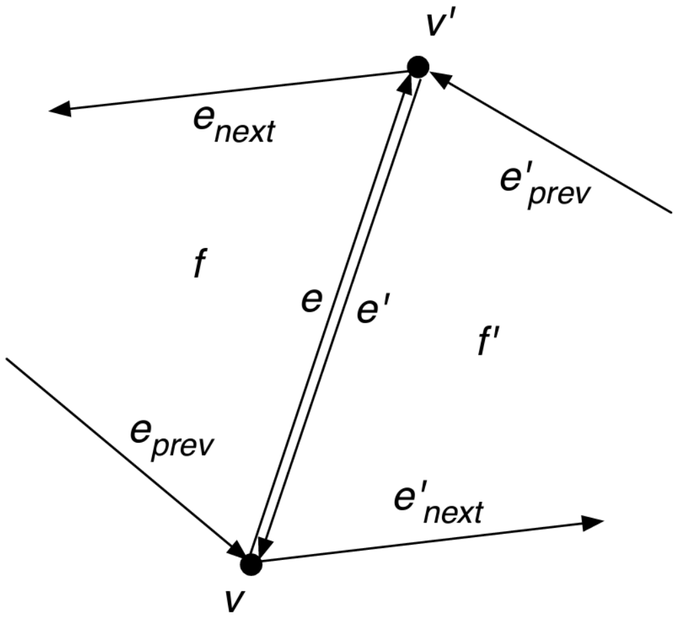
Winged *Edges*



Calcule la normal del vértice v



Winged *Edges*



Calcule la normal del vértice v

```
v->norm = Vec3(0,0,0);
Edge *e, *start;
e = start = v->e;
do {
   v->norm += e->f->norm;
   e = e->prev->sym;
} while (e != start);
v->norm.normalize();
```

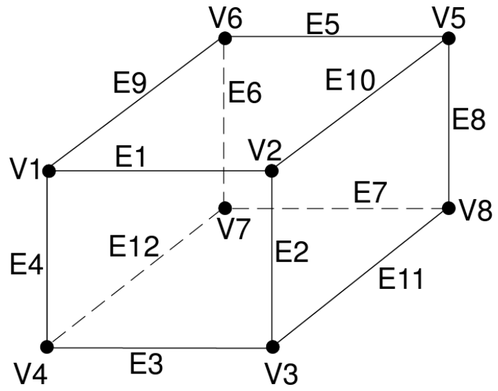
Winged *Edges*

Vertex-Edge table

VERTEX	X	Y	Z	ESTART
V_1	X_1	Y_1	Z_1	E_1
V_2	X_2	Y_2	Z_2	E_2
V_3	X_3	Y_3	Z_3	E_3
V_4	X_4	Y_4	Z_4	E_4
V_5	X_5	Y_5	Z_5	E_5
V_6	X_6	Y_6	Z_6	E_6
V_7	X_7	Y_7	Z_7	E_7
V_8	X_8	Y_8	Z_8	E_8

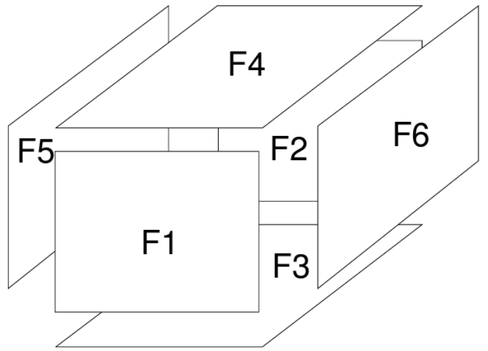
Face-Edge table

FACE	ESTART
F_1	E_1
F_2	E_5
F_3	E_{11}
F_4	E_9
F_5	E_4
F_6	E_8



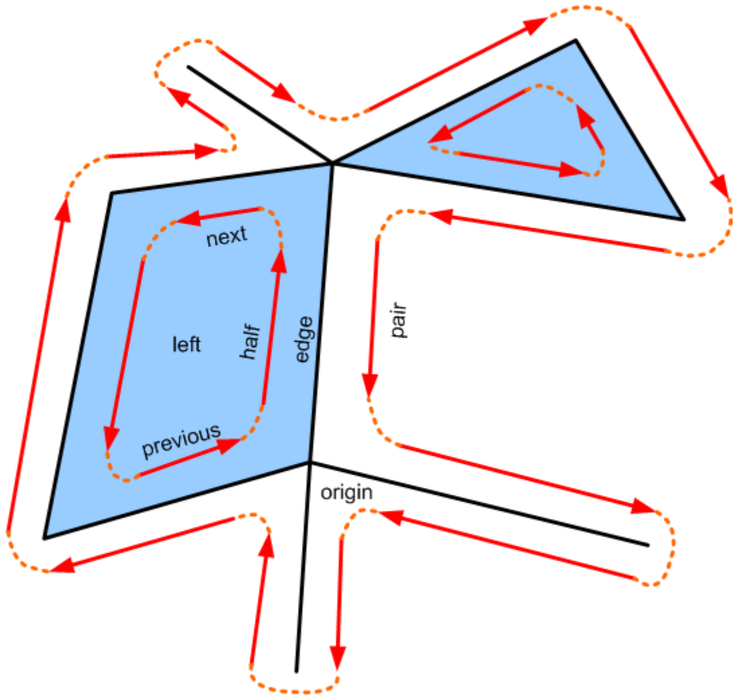
Edge-Edge relation

EDGE	VSTART	VEND	EPCW	ENCW	EPCCW	ENCCW	FCW	FCCW
E_1	V_1	V_2	E_4	E_2	E_{10}	E_9	F_1	F_4
E_2	V_2	V_3	E_1	E_3	E_{11}	E_{10}	F_1	F_6
E_3	V_3	V_4	E_2	E_4	E_{12}	E_{11}	F_1	F_3
E_4	V_4	V_1	E_3	E_1	E_9	E_{12}	F_1	F_5
E_5	V_5	V_6	E_8	E_6	E_9	E_{10}	F_2	F_4
E_6	V_6	V_7	E_5	E_7	E_{12}	E_9	F_2	F_5
E_7	V_7	V_8	E_6	E_8	E_{11}	E_{12}	F_2	F_3
E_8	V_8	V_5	E_7	E_5	E_{10}	E_{11}	F_2	F_6
E_9	V_1	V_6	E_1	E_5	E_6	E_4	F_4	F_5
E_{10}	V_5	V_2	E_5	E_1	E_2	E_8	F_4	F_6
E_{11}	V_3	V_8	E_3	E_7	E_8	E_2	F_3	F_6
E_{12}	V_7	V_4	E_7	E_3	E_4	E_6	F_3	F_5

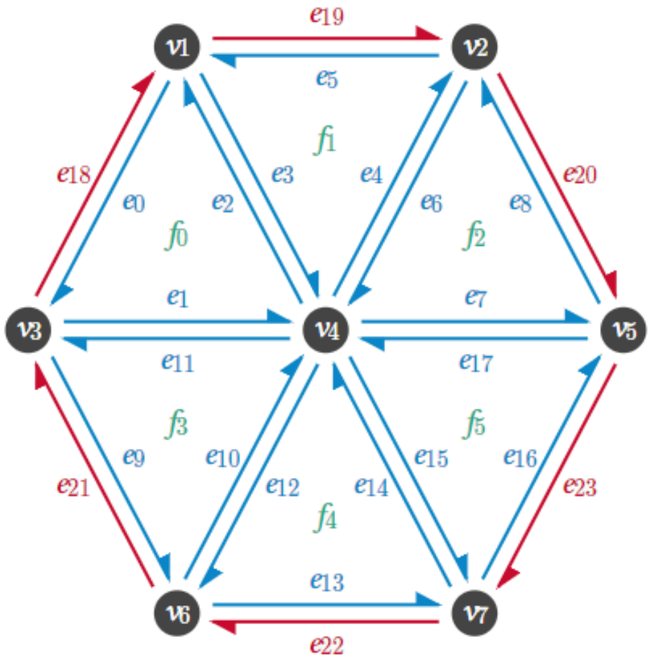


2. Half Edges

Half *Edges*



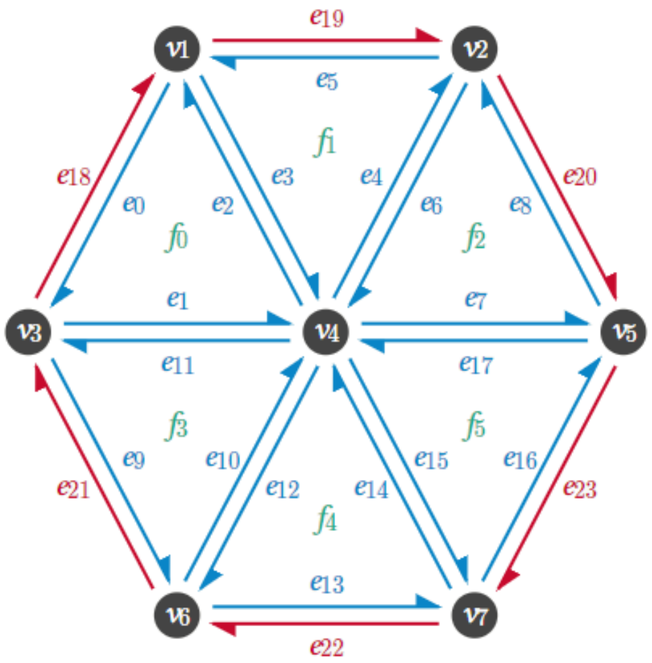
Half *Edges*



Vertex	Coordinate	Incident edge
v_1	(1, 4, 0)	e_0
v_2	(3, 4, 0)	e_5
v_3	(0, 2, 0)	e_1
v_4	(2, 2, 0)	e_2
v_5	(4, 2, 0)	e_8
v_6	(1, 0, 0)	e_{10}
v_7	(3, 0, 0)	e_{14}

Face	Half-edge
f_0	e_0
f_1	e_3
f_2	e_6
f_3	e_9
f_4	e_{12}
f_5	e_{15}

Half *Edges*

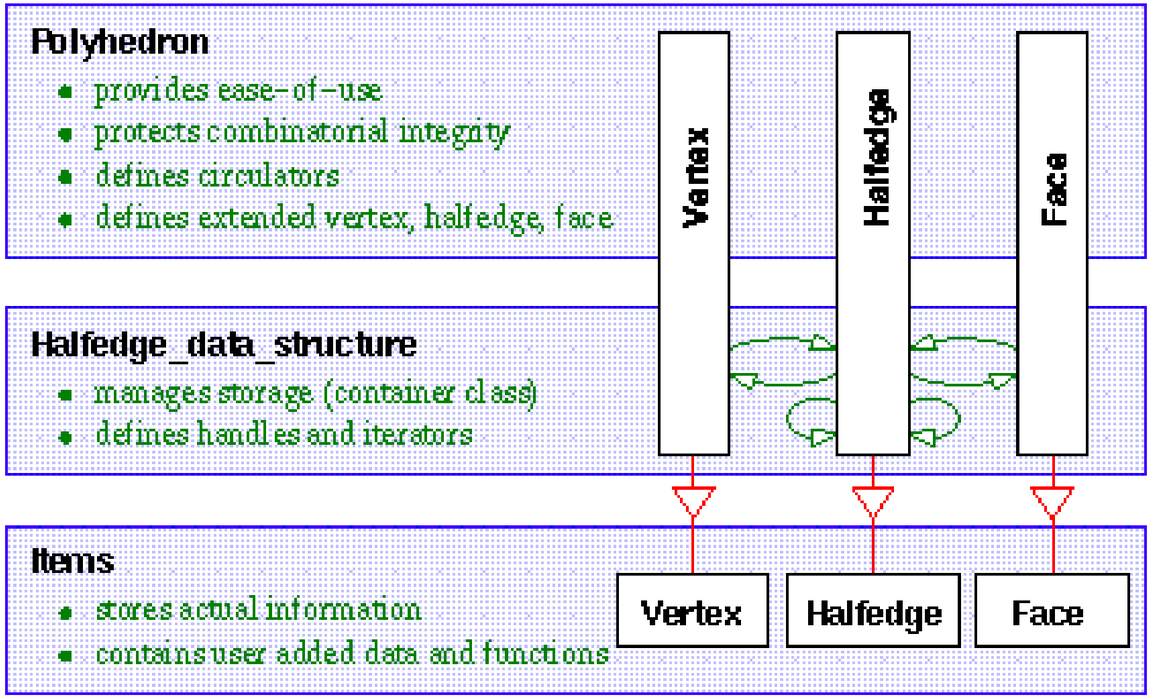


Half-edge	Origin	Twin	Incident face	Next	Prev
e_0	v_1	e_{18}	f_0	e_1	e_2
e_1	v_3	e_{11}	f_0	e_2	e_0
e_2	v_4	e_3	f_0	e_0	e_1
e_3	v_1	e_2	f_1	e_4	e_5
e_4	v_4	e_6	f_1	e_5	e_3
e_5	v_2	e_{19}	f_1	e_3	e_4
e_6	v_2	e_4	f_2	e_7	e_8
e_7	v_4	e_{17}	f_2	e_8	e_6
e_8	v_5	e_{20}	f_2	e_6	e_7
e_9	v_3	e_{21}	f_3	e_{10}	e_{11}
e_{10}	v_6	e_{12}	f_3	e_{11}	e_9
e_{11}	v_4	e_1	f_3	e_9	e_{10}
e_{12}	v_4	e_{10}	f_4	e_{13}	e_{14}
e_{13}	v_6	e_{22}	f_4	e_{14}	e_{12}
e_{14}	v_7	e_{15}	f_4	e_{12}	e_{13}
e_{15}	v_4	e_{14}	f_5	e_{16}	e_{17}
e_{16}	v_7	e_{23}	f_5	e_{17}	e_{15}
e_{17}	v_5	e_7	f_5	e_{15}	e_{16}
e_{18}	v_3	e_0	$\emptyset$	e_{19}	e_{21}
e_{19}	v_1	e_5	$\emptyset$	e_{20}	e_{18}
e_{20}	v_2	e_8	$\emptyset$	e_{23}	e_{19}
e_{21}	v_6	e_9	$\emptyset$	e_{18}	e_{22}
e_{22}	v_7	e_{13}	$\emptyset$	e_{21}	e_{23}
e_{23}	v_5	e_{16}	$\emptyset$	e_{22}	e_{20}

Half *Edges*

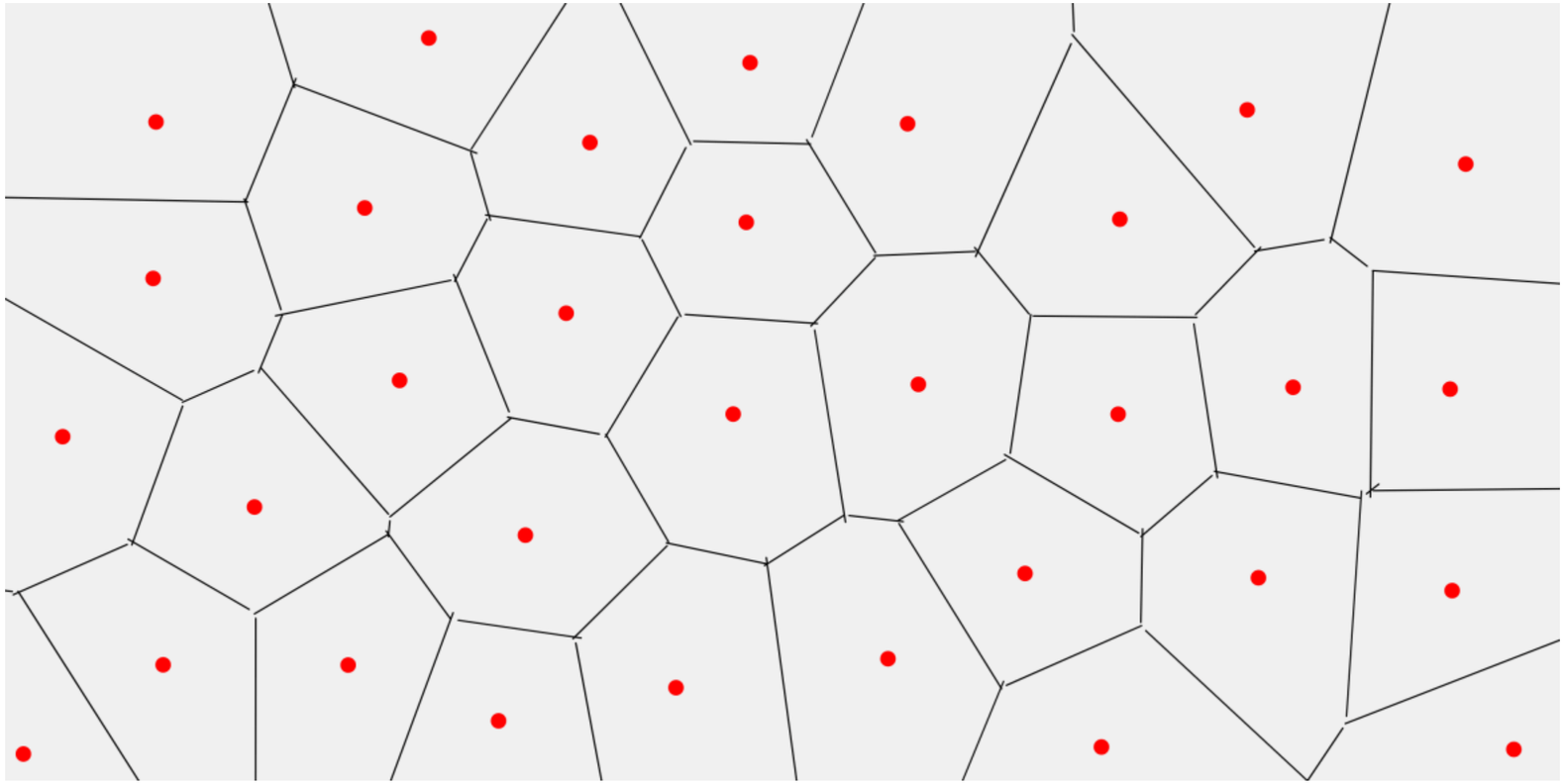
https://jerryyin.info/geometry-processing-algorithms/half-edge/

Half *Edges*

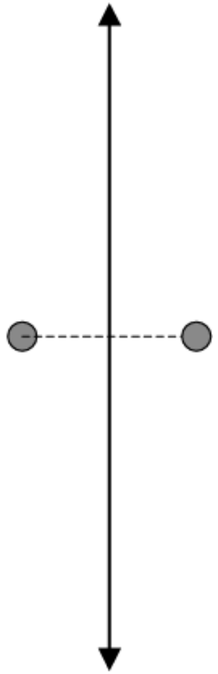

Polyhedron
provides ease-of-use
protects combinatorial integrity
defines circulators
defines extended vertex, halfedge, face
Halfedge_data_structure
manages storage (container class)
defines handles and iterators
Items
stores actual information
contains user added data and functions
Vertex
Halfedge
Face
Vertex
Halfedge
Face

3. **Voronoi** Diagram

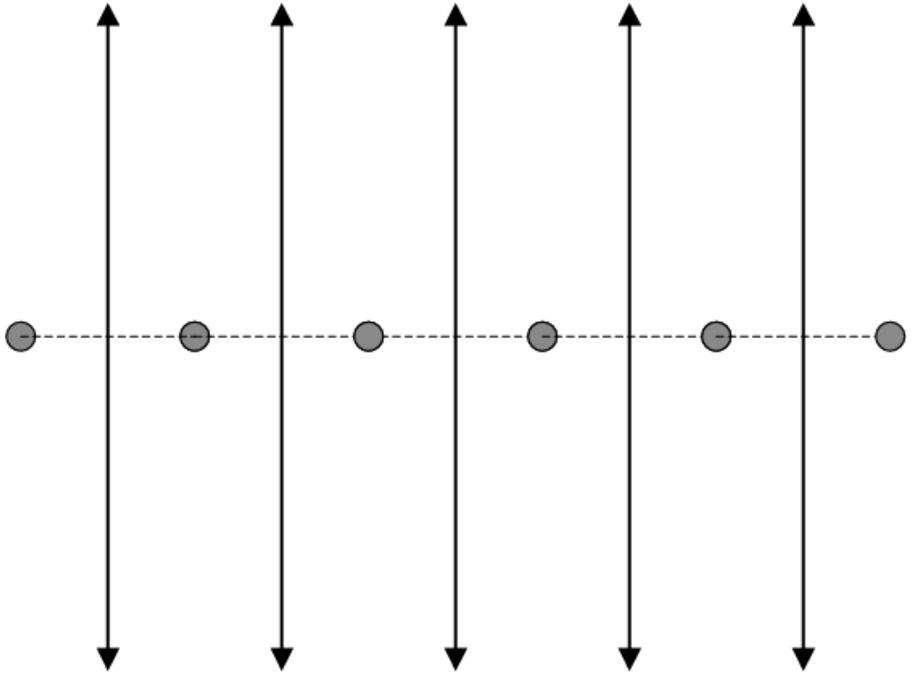
Voronoi *Diagram*



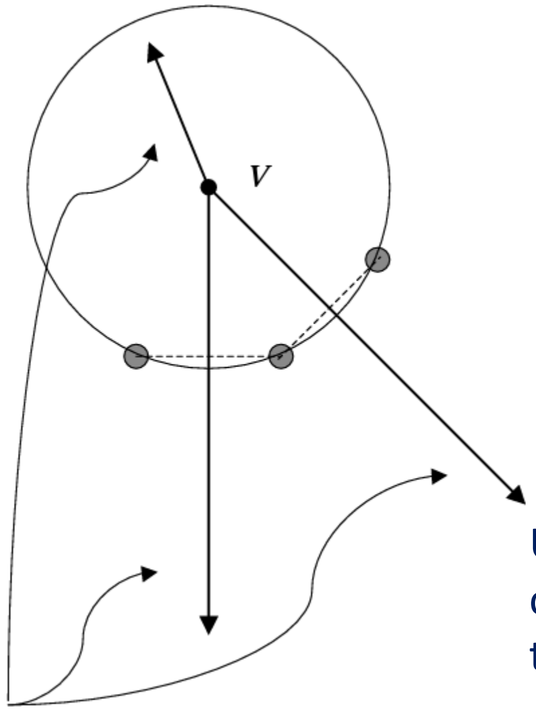
Voronoi *Diagram*



Voronoi *Diagram*



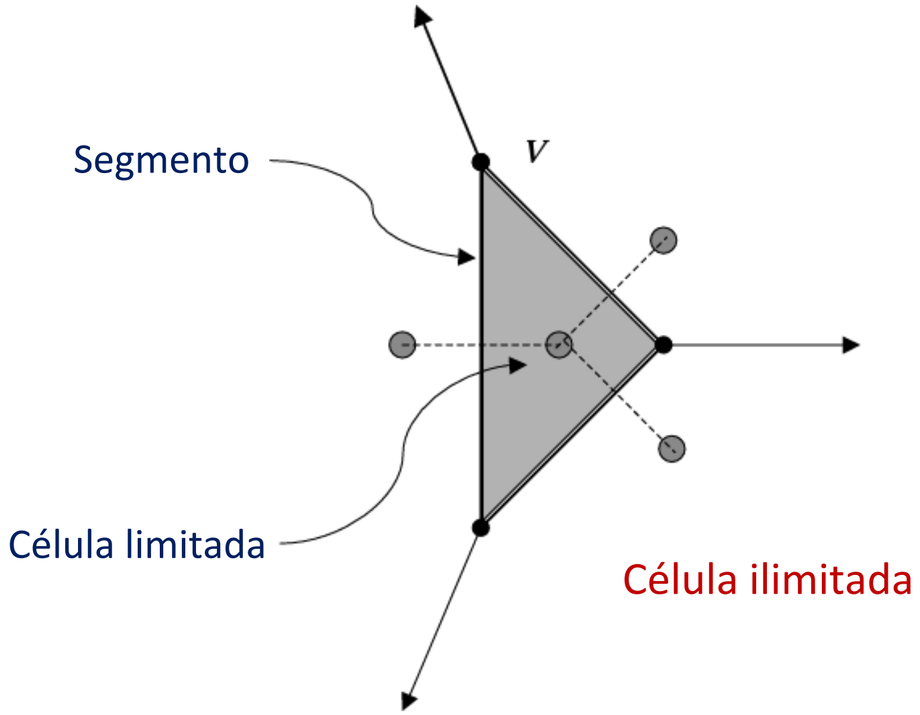
Voronoi *Diagram*



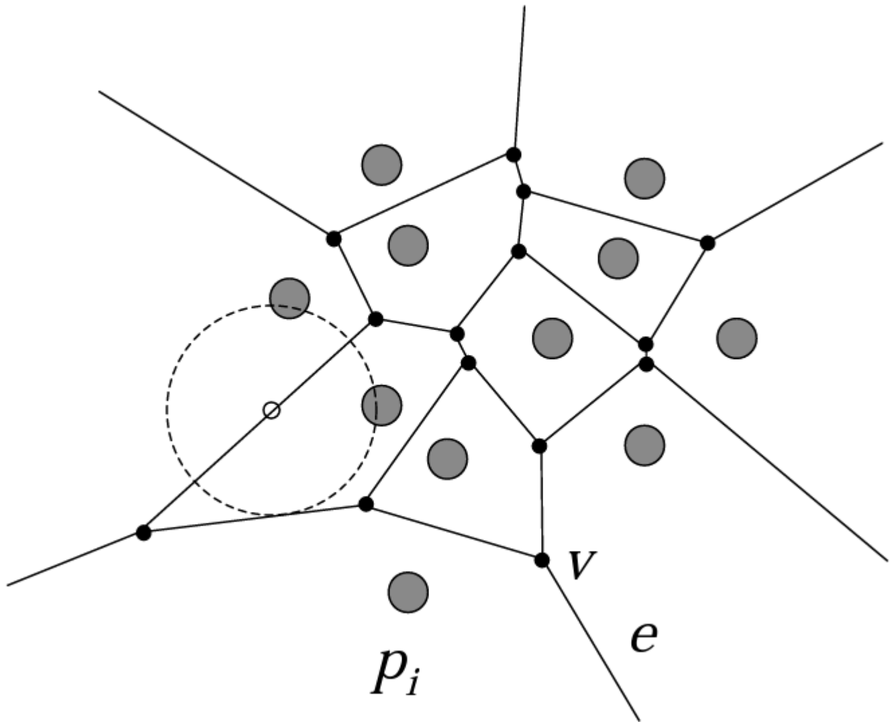
Un vértice de Voronoi es el centro de un círculo vacío que toca 3 o más sitios.

Líneas medias

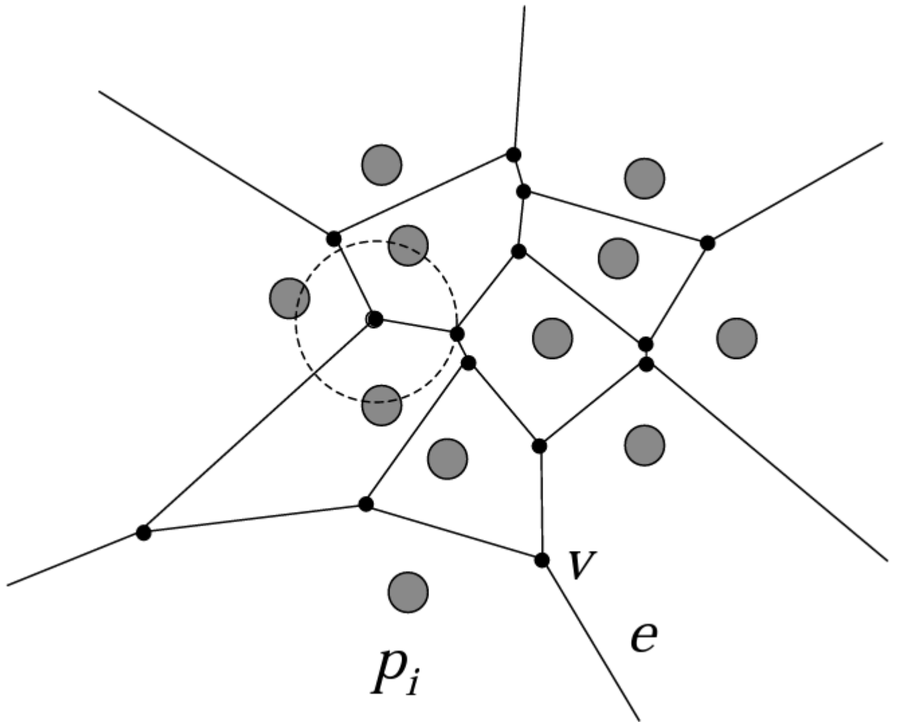
Voronoi *Diagram*



Voronoi *Diagram*



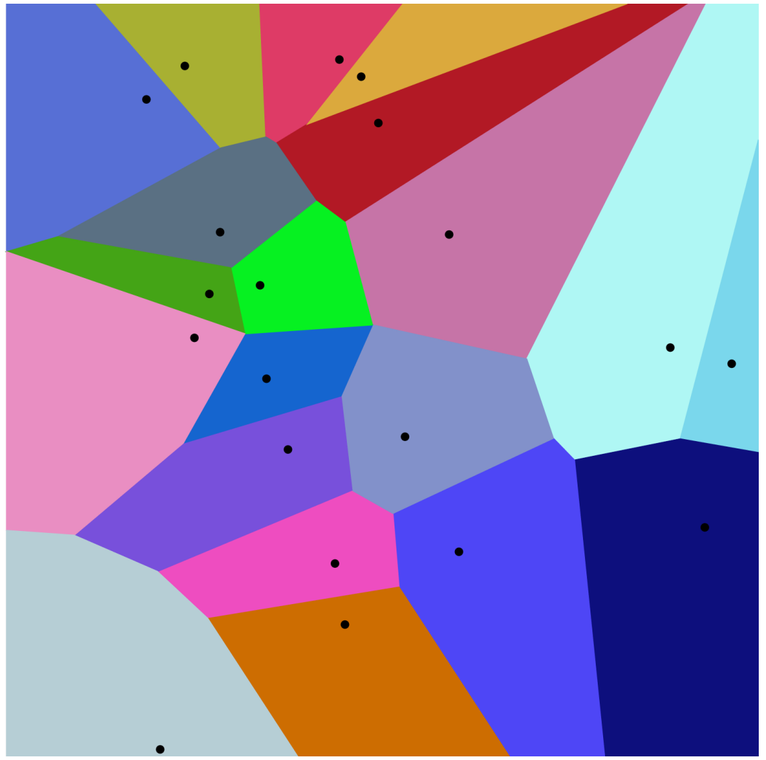
Línea equidistante a los dos puntos



Punto equidistante a los tres puntos

Voronoi *Diagram*

Distancia Manhattan



Distancia Euclideana

UTEC

UNIVERSIDAD DE INGENIERÍA
Y TECNOLOGÍA