




PROGRAMMAZIONE E CALCOLO SCIENTIFICO

Progetto finale a cura di Furnò Angelo Daniele,
Lupini Michele e Susca Erica



Indice

- Introduzione:

- *Scopo del progetto e requisiti*

- Parte 1:

- *Ipotesi*
 - *Costruzione del poligono*
 - *Taglio del poligono concavo*
 - *Taglio del poligono convesso*
 - *Stampa del poligono*

- Parte 2:

- *Costruzione del dominio*
 - *Costruzione della cella*
 - *Costruzione della mesh*
 - *Stampa della mesh*

Introduzione

Scopo del progetto e requisiti

- Nella Scienza Computazionale, un importante compito è il taglio di un poligono in diverse parti.
 - *L'obiettivo è generare una collezione di sottopoligoni, dato un poligono iniziale e una retta tagliante*
 - *Il risultato del taglio è visibile grazie a un file Matlab, che permette la stampa a schermo*
- La creazione di una mesh poligonale è un rilevante problema della Computer Grafica
 - *L'obiettivo è rivestire un dato dominio a partire da un elemento di riferimento*
 - *La mesh è visibile grazie a un file Matlab, che permette la stampa a schermo dei poligoni che la costituiscono*

TAGLIO DEL POLIGONO

PARTE 1



Ipotesi

- I dati del problema sono memorizzati in un file di input, contenente:
 - *un insieme di punti : punti del poligono iniziale*
 - *un insieme di vertici : vertici del poligono iniziale*
 - *una coppia di punti : estremi del segmento tagliante*
- I vertici del poligono sono disposti in senso antiorario
- Non ci sono ulteriori assunzioni riguardo a concavità, convessità, parallelismo della retta con i segmenti o parallelismo di segmenti tra loro

Costruzione del poligono

- La costruzione del poligono avviene a partire da un file di input avente una determinata formattazione.
- Tutti i dati del problema vengono raccolti in una classe su misura in diverse strutture dati:
 - *un vettore di punti che contiene tutti i punti del problema, non ordinati e caratterizzati da un indice*
 - *un poligono che rappresenta il poligono iniziale attraverso un vettore di indici associati ai punti del problema*
 - *un segmento che rappresenta la retta tagliante attraverso una coppia di indici*
 - *un vettore di segmenti utilizzato per contenere i lati del poligono iniziale*
 - *un vettore di poligoni che rappresenta i sottopoligoni generati a seguito del taglio*

Taglio del poligono concavo

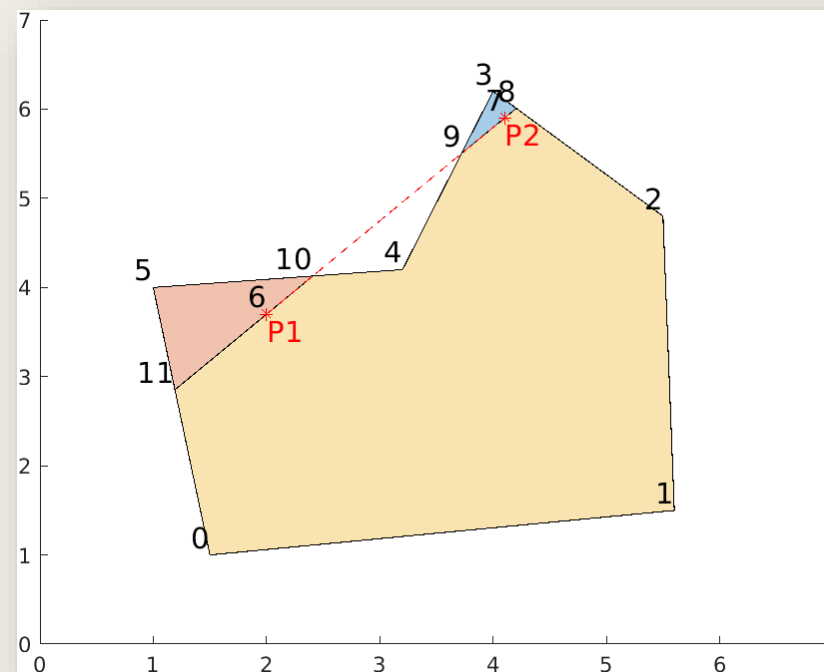
- Dato un poligono concavo del tipo dato in figura, utilizziamo delle strutture dati ausiliarie:

- *In BorderPoints vengono memorizzati tutti i punti sul bordo del poligono*

BorderPoints	0	1	2	8	3	9	4	10	5	11
--------------	---	---	---	---	---	---	---	----	---	----

- *In LinePoints vengono memorizzati tutti i punti appartenenti alla retta tagliante ordinati secondo l'ascissa curvilinea*

LinePoints	11	6	10	9	7	8
------------	----	---	----	---	---	---



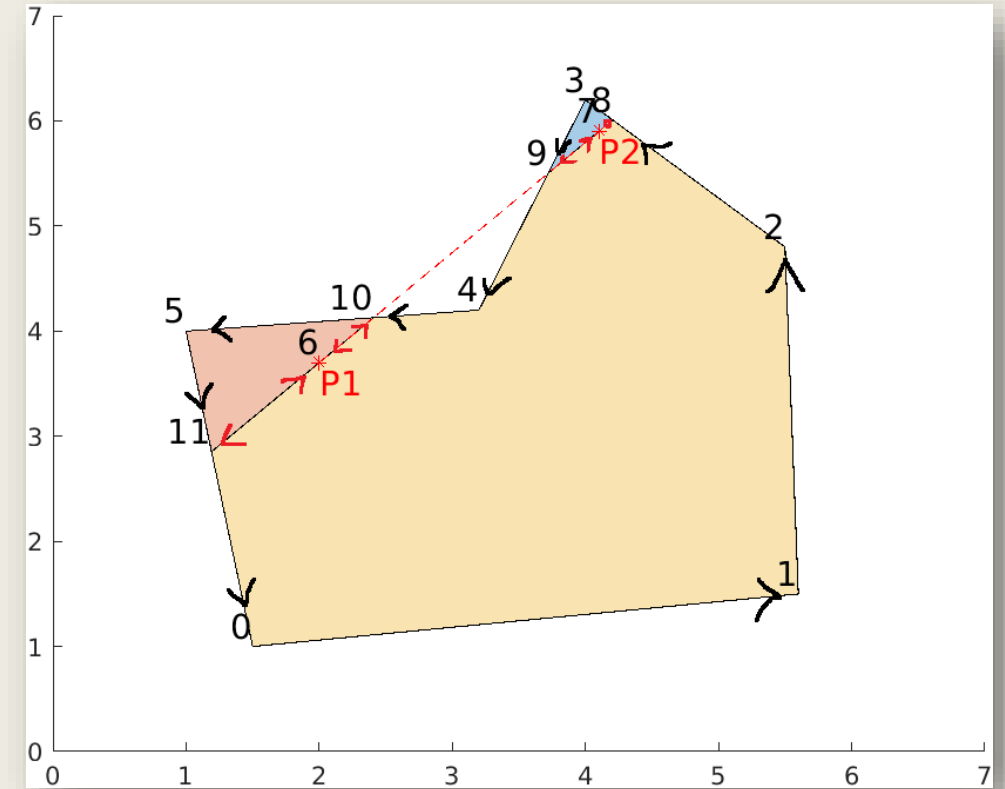
- *SideList* è una mappa con chiave l'intero associato al punto (sul bordo) e con dato un intero nel set $\{1, 0, -1\}$ che contrassegna la posizione del punto rispetto alla retta (sinistra della retta, sulla retta o a destra della retta)

SideList	Chiave	0	1	2	8	3	9	4	10	5	11
	Dato	-1	-1	-1	0	1	0	-1	0	1	0

- *InOutList* è una mappa con chiave l'intero associato al punto sulla retta e con dato un intero nel set $\{1, 0, -1\}$ che contrassegna il fatto che la retta sia rispettivamente entrante, uscente o nessuna delle due (caso di default assegnato agli estremi del segmento tagliante) rispetto al punto

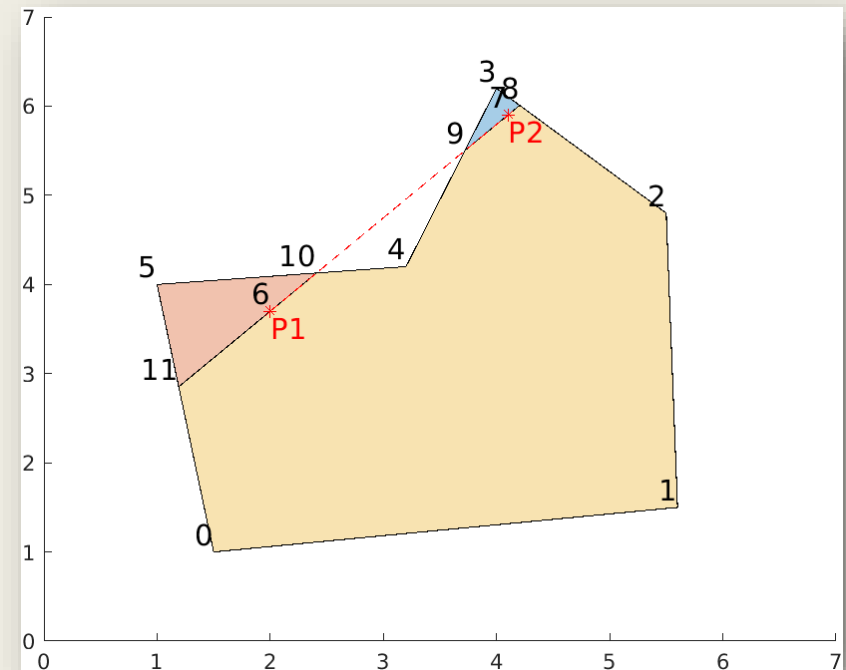
InOutList	Chiave	11	6	10	9	7	8
	Dato	1	-1	0	1	-1	0

- *Graph* è una mappa con chiave l'indice dei punti del problema e con dato una lista di adiacenza contenente:
 - il punto successivo sulla retta o sul bordo, se il punto in questione è un punto del poligono
 - il punto precedente sulla retta e il punto successivo sulla retta o sul bordo (se esistono), se mi trovo su un punto della retta
 - Al momento della creazione del grafo vengono esclusi collegamenti tra punti che creerebbero segmenti al di fuori del poligono grazie alla mappa InOutList



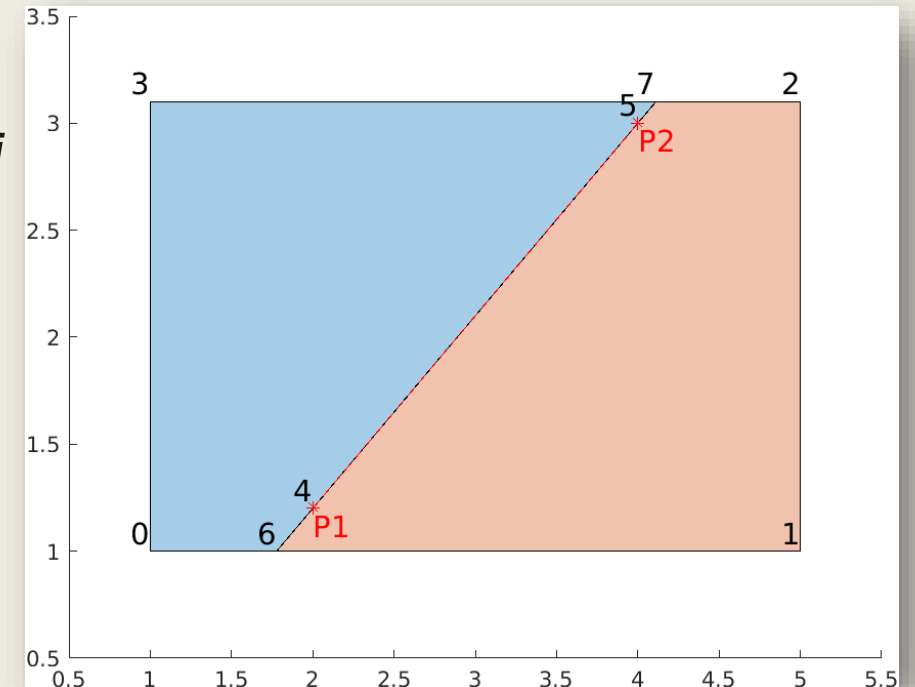
Graph	Key	0	1	2	3	4	5	6	7	8	9	10	11
	Data	1	2	8	9	10	11	10	8	7	7	6	6
								11	9	3	4	5	0

- Ciclando su tutti i punti del bordo, scegliamo il punto di partenza di un metodo ricorsivo per trovare i subpoligoni;
- se il punto considerato non appartiene alla retta tagliante ed il suo successivo sul bordo appartiene alla retta, allora il punto in questione è un punto di partenza.
- Per trovare i sottopoligoni si utilizza un metodo ricorsivo, che sfrutta la visita in profondità del grafo:
 - *se il nodo non è già stato visitato, allora viene aggiunto al vettore dei nodi visitati e continuo la visita se il nuovo punto non si trova dal lato opposto della retta rispetto al primo punto visitato*
 - *se il nodo è già stato visitato, coincide con il punto di partenza e il vettore dei visitati è maggiore o uguale a tre, aggiungo il poligono al vettore di subpoligoni*



Taglio del poligono convesso

- Il taglio del poligono convesso avviene in maniera analoga a quella del poligono concavo, con ottimizzazioni:
 - *dal momento che in questo caso è possibile avere al massimo due intersezioni, una entrante e l'altra uscente, non è necessaria l'uso della mappa InOutList*
 - *risparmio di memoria da allocare per i vettori LinePoints e BorderPoints e arresto anticipato del ciclo di ricerca delle intersezioni dovuto a una conoscenza aprioristica del numero massimo di punti sulla linea (i 2 estremi e, al peggio, 2 intersezioni distinte da questi)*

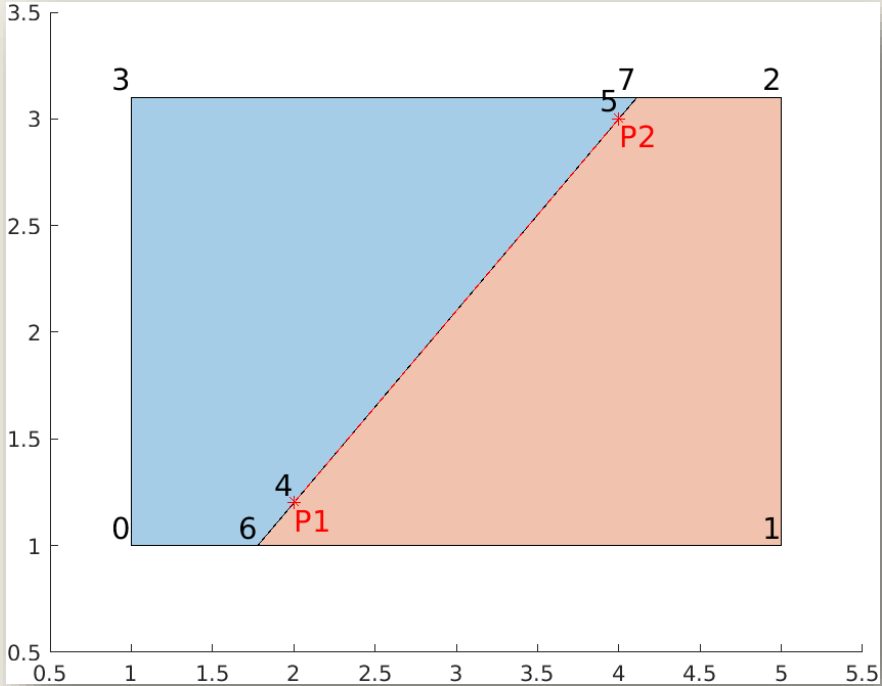


BorderPoints	0	6	1	2	7	3
--------------	---	---	---	---	---	---

LinePoints	6	4	5	7
------------	---	---	---	---

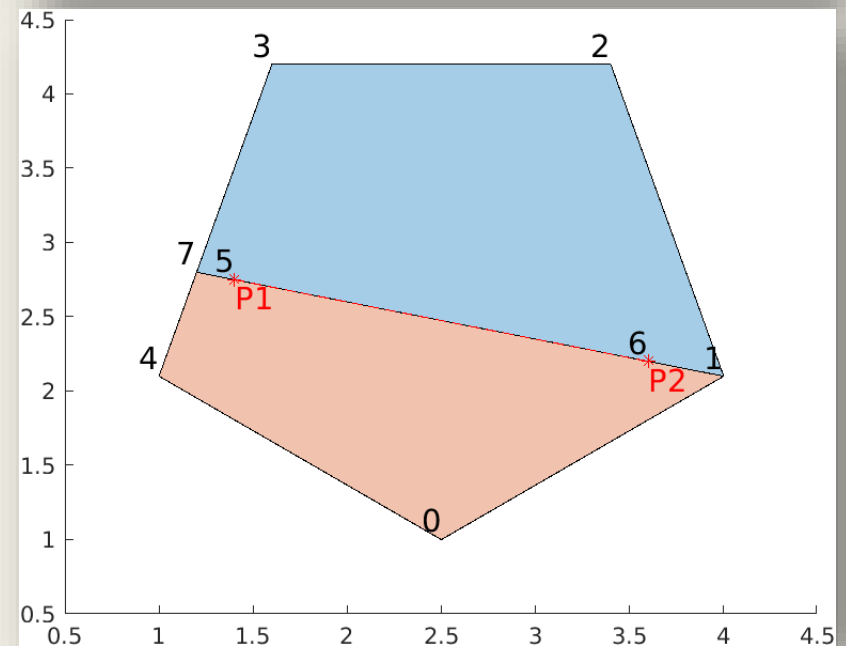
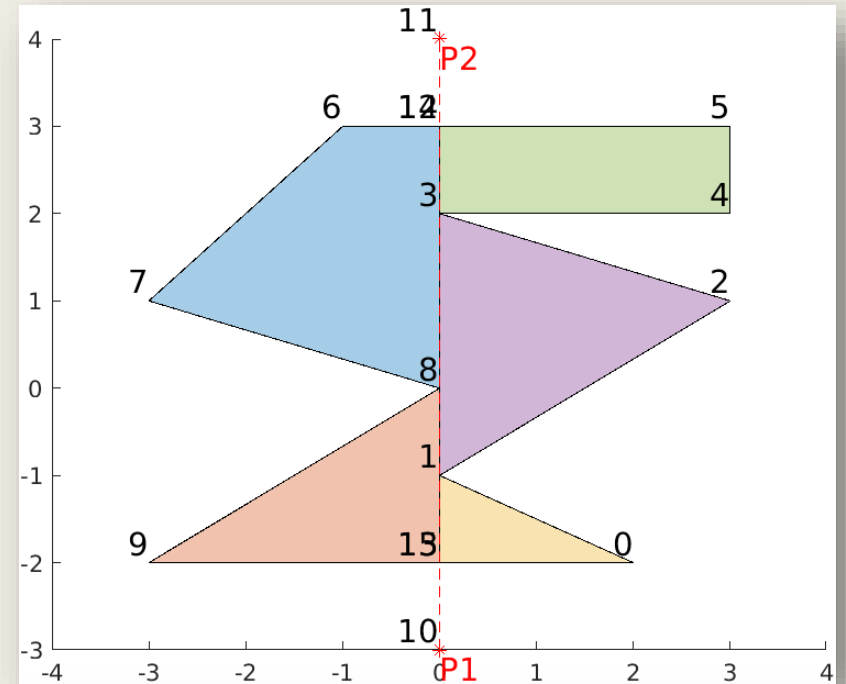
SideList	Chiave	0	6	1	2	7	3
	Dato	1	0	-1	-1	0	1

Graph	Chiave	0	1	2	3	4	5	6	7
	Dato	6	2	7	0	5	7	4	5
						6	4	1	3



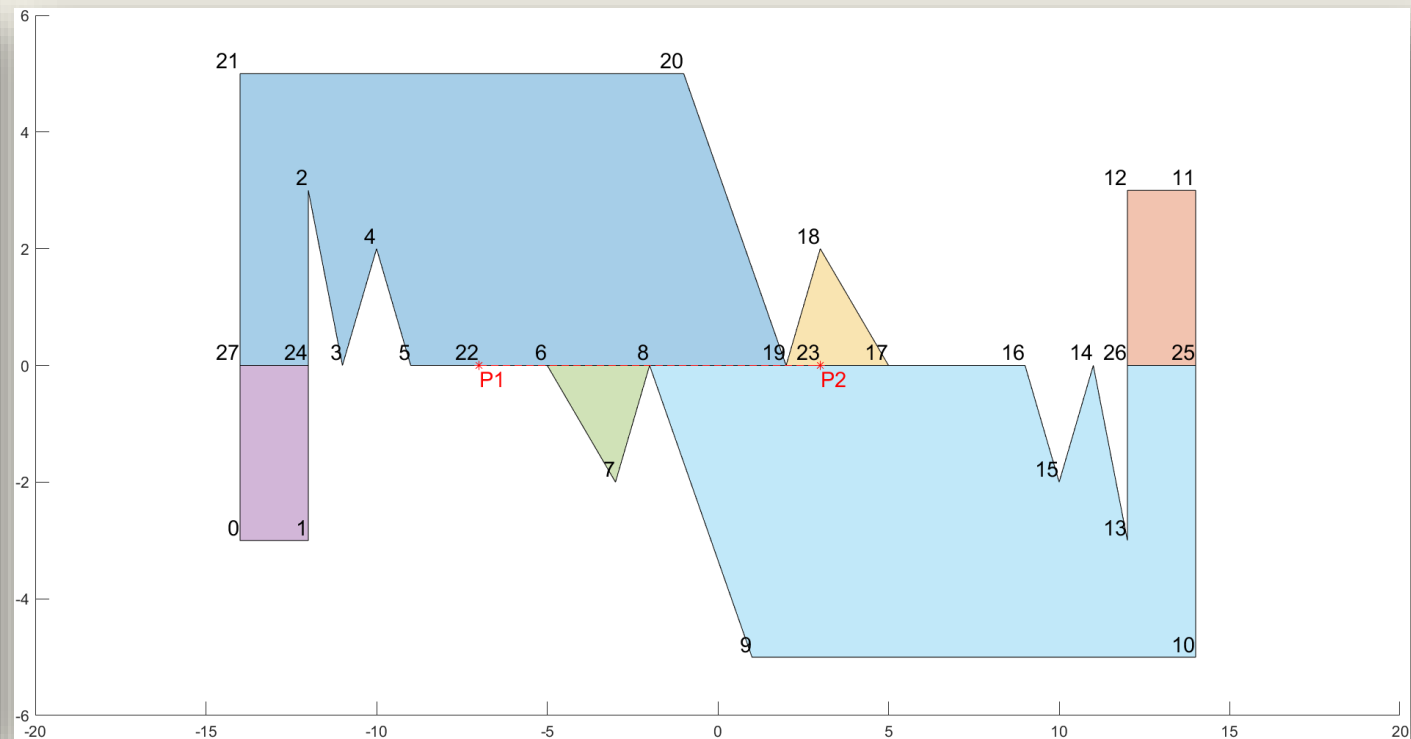
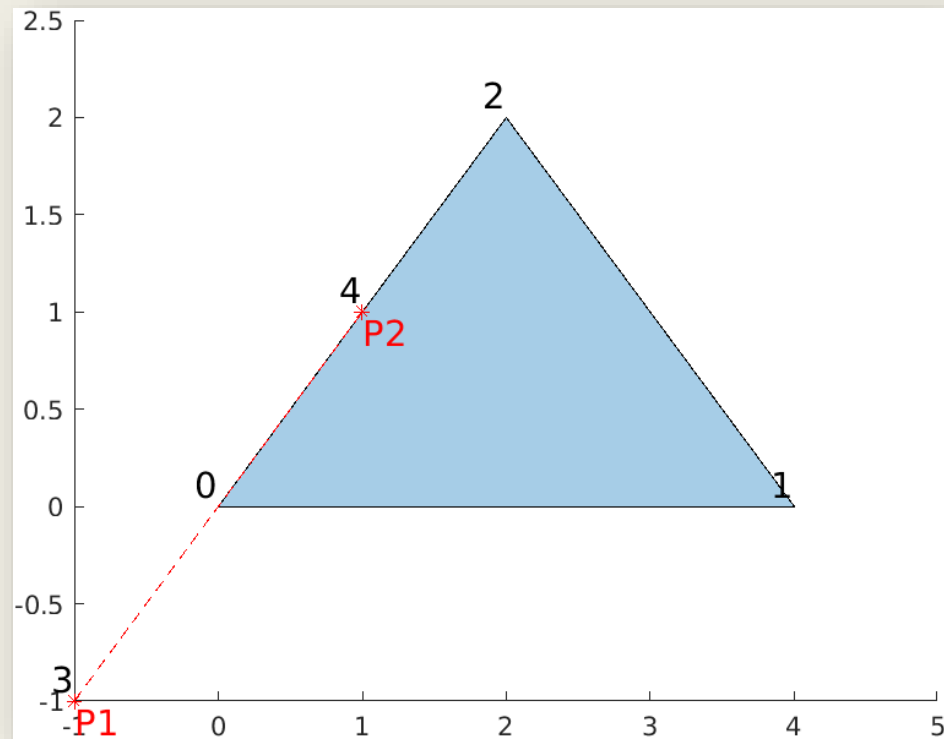
Casi particolari

- Caso di intersezione al vertice:
 - *se l'intersezione coincide con un vertice del poligono allora non vengono creati duplicati in BorderPoints;*
- Caso di retta tagliante esterna al poligono:
 - *Aggiunge il poligono nel vettore dei suoi subpoligoni senza chiamare alcun cutter;*



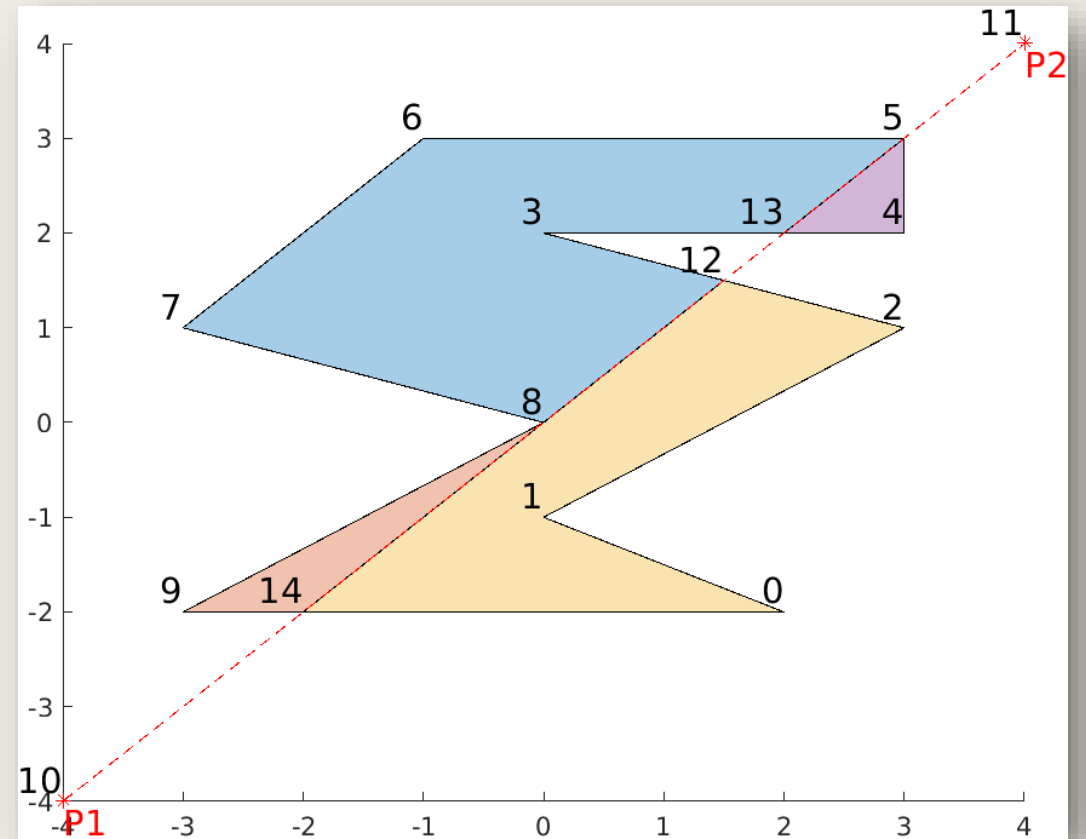
■ Caso di parallelismo:

- se uno o entrambi gli estremi del segmento tagliante cadono all'interno di un lato, allora essi sono aggiunti ai punti del bordo



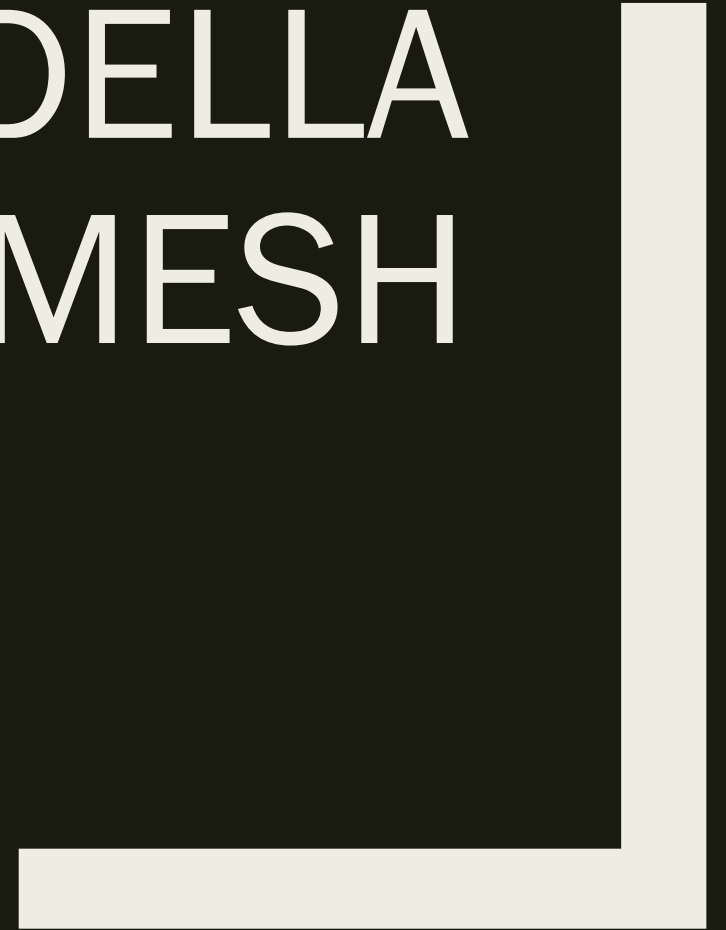
Stampa del poligono

- La factory contiene tutti i dati del problema necessari per creare dei file su Matlab, che permettono di plottare:
 - *il poligono iniziale con la numerazione di ciascun vertice*
 - *il segmento tagliante*
 - *i sottopoligoni generati a seguito del taglio*



COSTRUZIONE DELLA MESH

PARTE 2

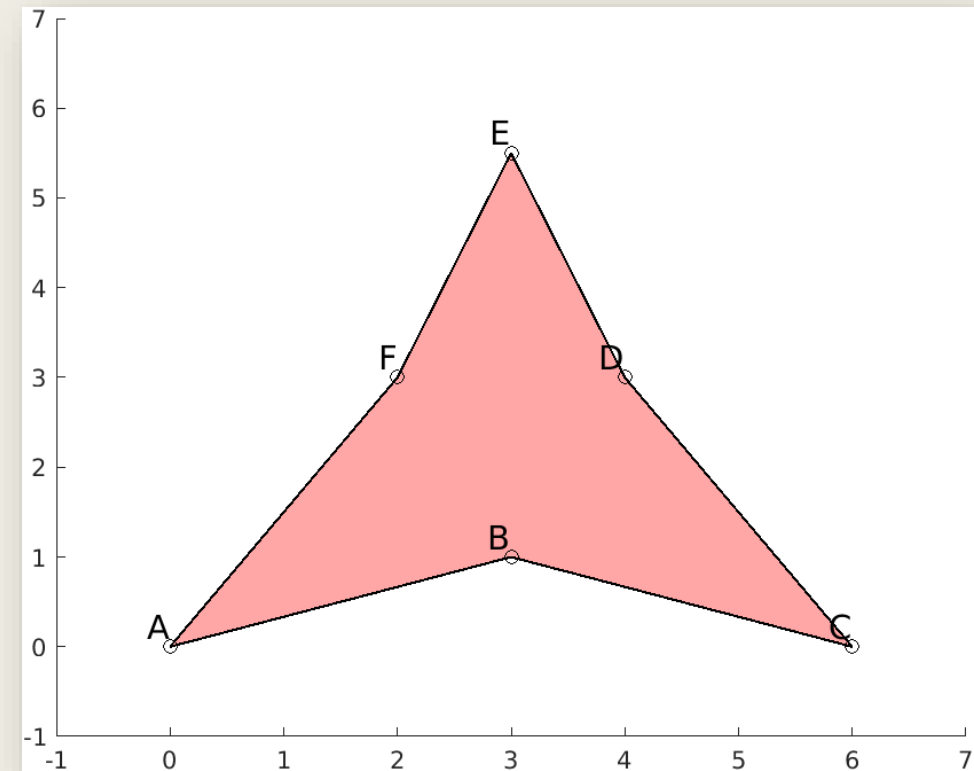
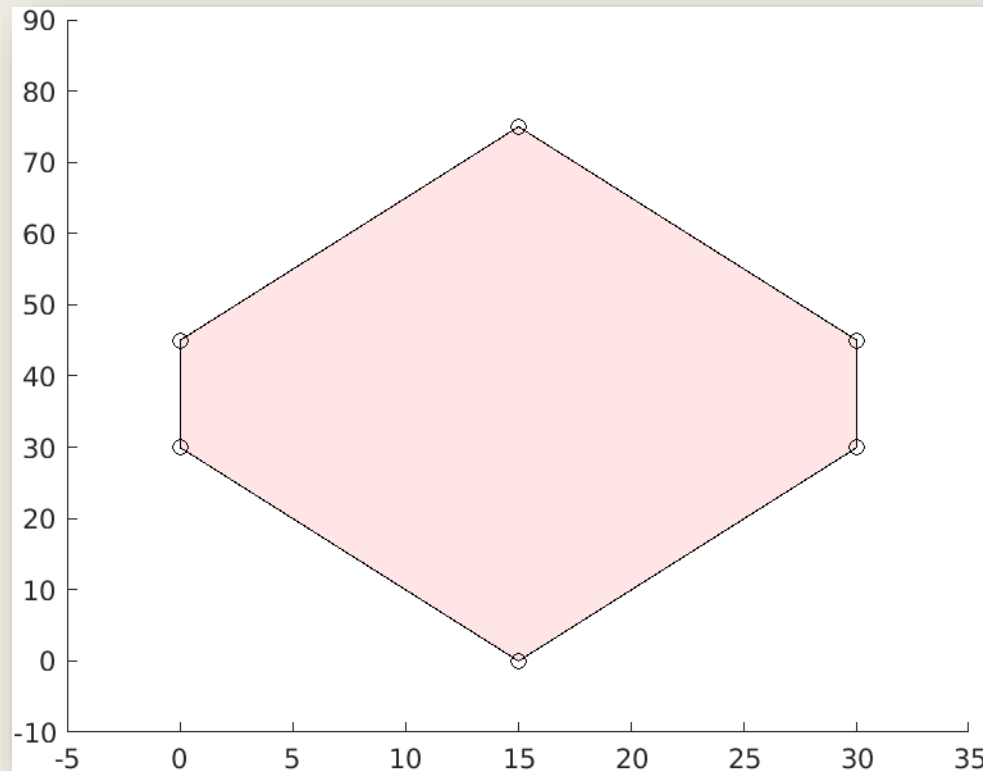


Ipotesi

- Dominio e poligono iniziale sono forniti in input da codice stesso e hanno i vertici orientati in senso antiorario
- Il poligono che fa da dominio è assunto come convesso per facilitare le operazioni di costruzione della mesh;
- Non ci sono ulteriori assunzioni riguardo a concavità, convessità, parallelismo della retta con i segmenti o parallelismo di segmenti tra loro

Costruzione del dominio

- La costruzione della mesh avviene a partire da un dominio convesso per ipotesi e da un elemento di riferimento, costruito a partire da un poligono qualsiasi
- Il dominio è un poligono convesso definito come un vettore di punti



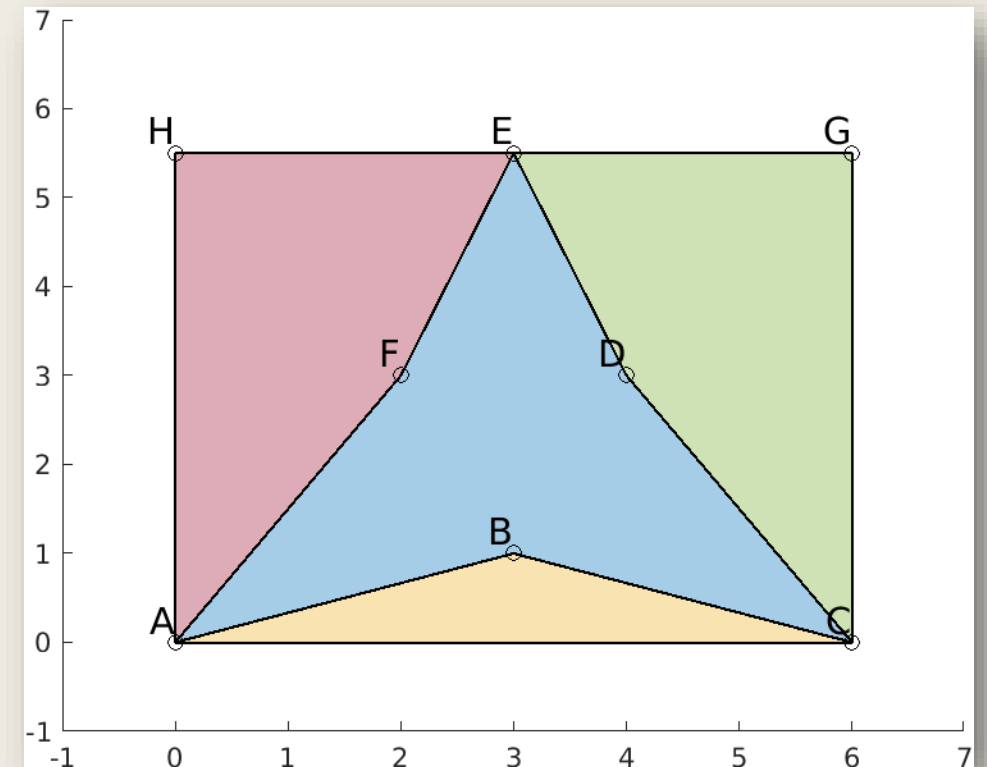
Costruzione della cella

- L'elemento di riferimento si identifica come un vettore di poligoni racchiusi in un bounding box:
 - si costruisce a partire da un poligono centrale (*PolygonPoints*), di cui si calcola il minimo rettangolo che lo racchiude, sfruttando le coordinate estremali

PolygonPoints
A
B
C
D
E
F

- si calcolano i punti di intersezione del poligono centrale con il bounding box e si costruisce il vettore ordinato di punti che costituiscono il bordo (*BorderPoints*)

BorderPoints
A
C
G
E
H



- *si costruisce una mappa (Linker) con chiave l'indice del punto del bordo e con dato l'indice del punto corrispondente sul poligono, se esiste, altrimenti un valore di default (-1)*

Linker	Chiave	0	1	2	3	4
	Dato	0	2	-1	4	-1

- *ciclando sui punti del bordo si parte da ogni punto di intersezione tra poligono e bounding box tale che il successivo non sia anch'esso di intersezione e si procede con una visita in profondità*
- *a partire dal punto iniziale si visitano i punti del bordo fino a trovare un altro punto di intersezione*
- *sfruttando il Linker si passa al corrispettivo punto sul poligono e si procede in senso orario fino a trovare il punto iniziale, chiudendo il poligono*

Costruzione della mesh

- A partire dal dominio si costruisce il minimo rettangolo che lo contiene (bounding domain) e considerando le dimensioni della cella si calcola il numero minimo di celle, lungo gli assi x e y, tali che ricoprono il dominio
- Si trasla l'elemento di riferimento in modo che il suo vertice in basso a sinistra coincida con il vertice in basso a sinistra del bounding domain
- Per ogni cella contenuta nel bounding domain:
 - *se la cella completamente fuori dal dominio allora non faccio niente*
 - *se la cella è contenuta completamente all'interno del dominio, aggiungo alla mesh tutti i suoi poligoni*
 - *se la cella è parzialmente contenuta nel dominio, per ogni lato del dominio si tagliano i poligoni nella cella mantenendo i sottopoligoni sinistri*

Stampa della mesh

- Una classe ad hoc permette la creazione di file su Matlab che atti a plottare:
 - *il poligono costituente il dominio della mesh*
 - *la mesh in uno stato precedente al taglio*
 - *la mesh finale a seguito del taglio*

