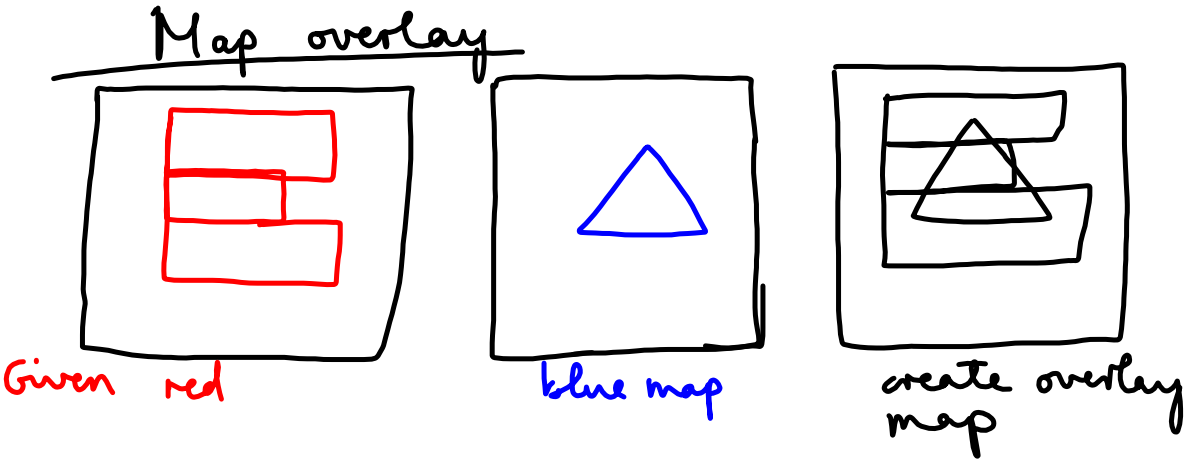
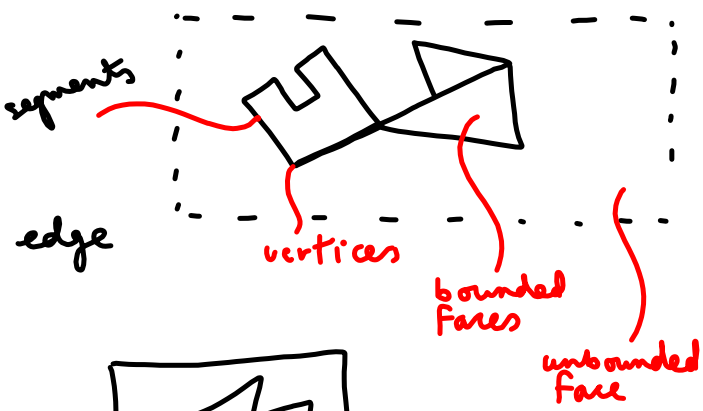
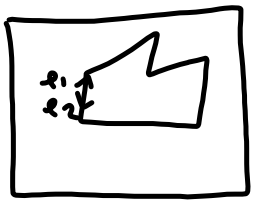

Map overlay
Given red
blue map
create overlay map

Map (planar subdivision) - embedding of a graph into plane $\mathbb{R}^2$

- Store planar subdivision in a dcel : doubly connected edge list.
- In this approach, orientation of edges is important:





DCEL: 3 Tables

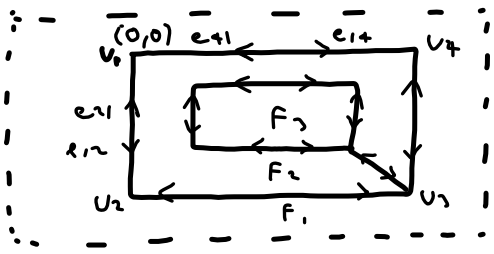
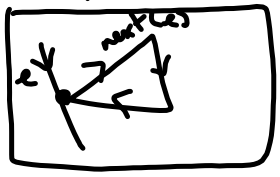


Table for vertices

Name of vertex	Co-ordinate	Edge originating at vertex
V_1	(0,0)	e_{12}

Also Tables for edges, faces

Example of next(e)



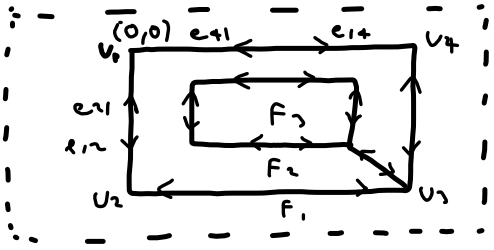


Table for edges

Name of edge	Origin (vertex)	Twin	Next edge	Previous edge	Adjacent face
e_{12}	v_1	e_{21}		e_{41}	f_2

- Adjacent face : face to left of oriented edge
- Next(e) - origin is endpoint of e,
  - next(e) has same adjacent face as e,
- no edge between e & next(e) has these two properties.

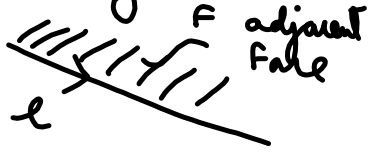
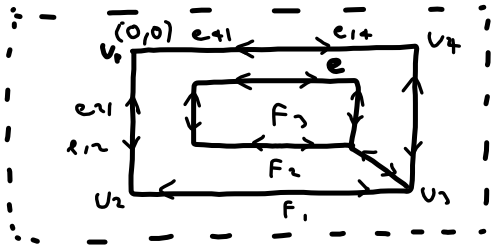


Table for faces

Name of face	1 edge on outer cycle	1 edge on each inner cycle
f_2	e_{12}	
f_1	none	e_{21}



- Cycle: sequence $(e_1, \dots, e_n)$ of edges with $next(e_i) = e_{i+1}$ & $next(e_n) = e_1$.
- It is a cycle of F if $adj(e_i) = F$ for any (all) e_i.
- Outer cycle of f if edges e_i lie on outer boundary of f.
- Inner cycle otherwise.

Complexity of planar subdivision/DCEL is no. of vertices + no of edges + no. of faces.

Exercise:
Using dcel, calculate all edges with origin v in clockwise order.

Algorithm for map overlay

S_1 red map .... D_1 dcel

S_2 blue map .... D_2 dcel

$S = \text{Overlay}(S_1, S_2)$ – calculate dcel D for overlay S.

(see Fig 3.1 E-learning)

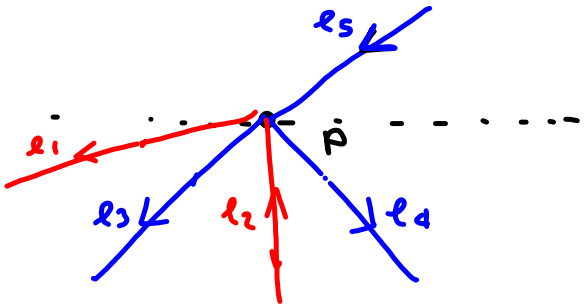
(Detailed version in Jantun's thesis - see E-learning)

Algorithm has 3 steps:

1. Put tables for vertices & edges of D_1 & D_2 into single table D (record colour of edges).
2. Using segment intersection alg., modify tables for vertices & edges.
3. Finally, create table for faces. For each face $f \in D$, find red face f_1 & blue face f_2 in which f lies.

Algorithm involves:
- "event queue" - a bin. balanced tree
- bin bal tree T of line segments (with colour),
- For each $p \in Q$, sets $L(p)$, $C(p)$, $U(p)$ of coloured line segments on which p lies (as in last week's algorithm)
- Step ① of algorithm is straightforward - here we will describe steps ② & ③.
- For Step ②, we begin by adding all endpoints of segments to Q.
- Several cases to consider at event point $p \in Q$.

Given $p \in Q$,
- update Q & T as in segment intersection alg.,
  but keeping track of colours of segments.
- If $C(p) = \emptyset$,
  in tables for vertices & edges do not add any new vertices or edges - only update next & previous.



Original table
$next(\ell_2) = \ell_1$

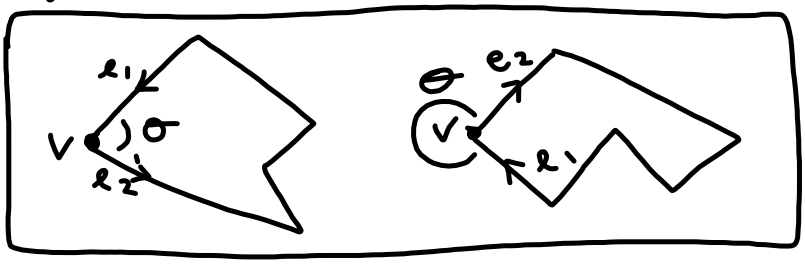
New table
$next(\ell_2) = \ell_3$

For $C(p) \neq \emptyset$, see Fig 3.5 in E-learning.

- For Step ③, we must calculate faces of DCEL.
- Idea :- each bounded face has a unique outer cycle.
  - unbounded face has no outer cycle.

  Can define faces of D to be
  Outer cycles ∪ {c_∞} — imaginary outer cycle for unbounded region.
- Which cycles of D are inner & which are outer?

- Calculate cycles of D using tables for edges & vertices since cycles are determined by edges, vertices & next pointer.
- At a cycle c, choose leftmost vertex V

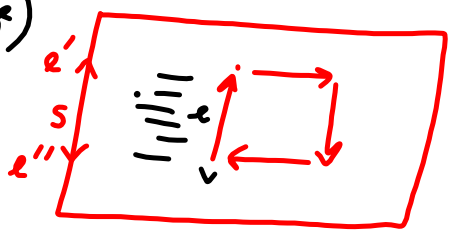


- let e_1, e_2 be edges going into & out of V.
- let θ be angle bet. e_1, e_2 measured over adjacent region.
- $\theta < 180°$ ~ outer cycle
- $\theta > 180°$ ~ inner cycle ~ $\theta < 180° \sim \det\begin{pmatrix} e_{1x} & e_{1y} \\ e_{2x} & e_{2y} \end{pmatrix} > 0$

- Draw a graph of all cycles (incl. c_∞)
- Each connected component of graph will contain precisely one outer cycle (poss. c_∞) & then we will be able to fill in the face information using this.

The graph

- Choose inner cycle c (with edge e)
- Find leftmost vertex of cycle.
- Find closest segment s to its left.
- Determines two edges e', e'', one of which has same adj. face as e, in this case e''.
- Call cycle on which e' lies k & draw edge from e to k in the graph.



- If k, outer, stop.
- If k inner, repeat.
- If nothing to left of v, we connect e to c_∞.
- For graph example, see Fig. 3.8 in E-learning.
- Complete face table for D:
  - adj(e) = the outer cycle it is connected to (c_∞ or).
  - Outercycle(f) = f or if $f = c_\infty$, then ϕ.
  - Innercycles(f) = those to which f is connected in graph.

Identification of Faces - see E-learning.

Complexity

- S_1 planar subdiv. of comp n_1
- S_2 - - . - - - - - n_2

& let $n = n_1 + n_2$

Complexity of overlay alg.

$O((n+k)\log n)$ — k complexity of overlay.