Diese Gerade l_k ist die duale Grade zum Pkt $p_k = near(p_i, p_j)$ $l_k = D(p_k)$

- Dieses Verfahren wenden wir auf alle Knoten p_{ij} (die representativ für die Gerade durch p_i und p_j stehen) an und erhalten so $near(p_i, p_j)$ $\forall (i,j)$.

Laufzeit:
$\Theta(n^2)$ da linear für jede Fläche

$\Rightarrow$ Lineare Suche auf der Menge $\Delta(p_i, p_j, near(p_i, p_j))$ liefert eine Lsg in Zeit $O(n^2)$.

6.5 Berechnung des Arrangements von n Geraden

6.5.1 Verfahren I: Plane Sweep

- Laufzeit:
  $\Theta(n^2 \log n)$

6.5.2 Verfahren II: Inkrementelles Verfahren

Geg.: n Geraden $l_1, \dots, l_n$
Ges.: Berechne Folge von Arrangements $A_1, \dots, A_n$ wobei A_i alle Geraden $\{l_1, \dots, l_i\}$ enthält.

- Algorithmus:
  - A_1: trivial (enthält nur die Gerade l_1)
  - Konstruktion von A_i durch Hinzufügen von l_i zu A_{i-1}:
    1) • Starte in bel Schnittpkt $p = l_i \cap l_j$ $(j < i)$
    2) • Durchlaufe von p aus (nach links und rechts) die Flächen von A_{i-1} (genauer: die Kanten der Flächen) bis Gerade l_i wieder getroffen wird. Dort ist der nächste Schnittpkt
    3) • Update der Datenstruktur ($\hat{=}$ planarer Graph), dh Einfügen neuer Knoten und Kanten.

- Lemma:
  Sei $m_i :=$ Kosten für Einfügen von l_i in A_{i-1}. Dann gilt:
  - m_i ist proportional zur # der in 2.) durchlaufenen Kanten.
  - $m_i \leq 5 \cdot i$

- Satz:
  Das Arrangement von n Geraden kann in Zeit $O(n^2)$ berechnet werden.
  Beweis:
  Inkrementeller Algorithmus, dh das Hinzufügen von l_i kostet $O(i)$

$$\left.\begin{array}{l} m_1 \text{ kostet } O(1) \\ m_2 \text{ kostet } O(2) \\ \vdots \\ m_n \text{ kostet } O(n) \end{array}\right\} \text{n-mal} \Rightarrow O(n^2)$$