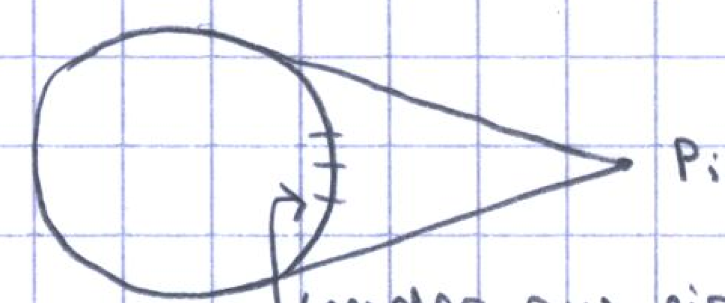
6.2.1.3 Bem:

Am Ende bleibt also:

6.2.1.4. Laufzeit:

Im $\mathbb{R}^2$:



werden nur einmal entfernt + Kosten für zwei Tangenten.

im $\mathbb{R}^3$:

Rand ist nicht konstant ($\Rightarrow$ mehr als zwei Tangenten)

$O(n)$ möglich!

Im schlechtesten Fall: Laufzeit: $O(n^2)$.

In der Praxis wird der Alg. dennoch eingesetzt, da er für viele Eingaben doch relativ schnell ist.

6.2.1.5 Bem:

⊕ einfach, praktisch, effizient (für best Eingaben)

⊖ im schlechtesten Fall: $O(n^2)$.

$\rightarrow$ Randomisierung: Sweep in zufällige Richtung

$\Leftrightarrow$ Drehung der Punktmenge in zufälligen Winkel.

Resultat: Erwartete Laufzeit: $O(n \log n)$.

6.2.2. Divide & Conquer-Alg:

6.2.2.1. Algorithmus:

1. Sortiere $S \rightarrow \{p_1 .. p_n\}$
2. Zerlege in S_1 und S_2 (zwei Hälften gemäß sortieren)
   $S_1 := \{p_1 .. p_{n/2}\}$, $S_2 := \{p_{n/2+1} .. p_n\}$.
3. Berechnung rekursiv: $C_1 := CH(S_1) \wedge C_2 := CH(S_2)$.
4. Misch-Schritt: Konstruieren $CH(S)$ durch Einwickeln von C_1 und C_2.

Bem:

- Einwickeln der ganzen Punktmenge (siehe 2d) kostet im schlechtesten Fall $O(n^2)$. vgl. s.o.
- jeder Schritt verläuft genauso wie im 2-dim. Fall mit folgendem Unterschied:
  a) verwende 3d-Orientierungsprädikat
  b) 2 Pkte (d.h. Kante $\overline{ab}$) fest, man sucht Pkt c mit minimalem Winkel.

Wir nutzen aus, dass wir zwei konvexe Polyeder C_1 und C_2 einwickeln.

$\rightarrow$ Tangentialzylinder.

6.2.2.2. Situation (Mischschritt):