

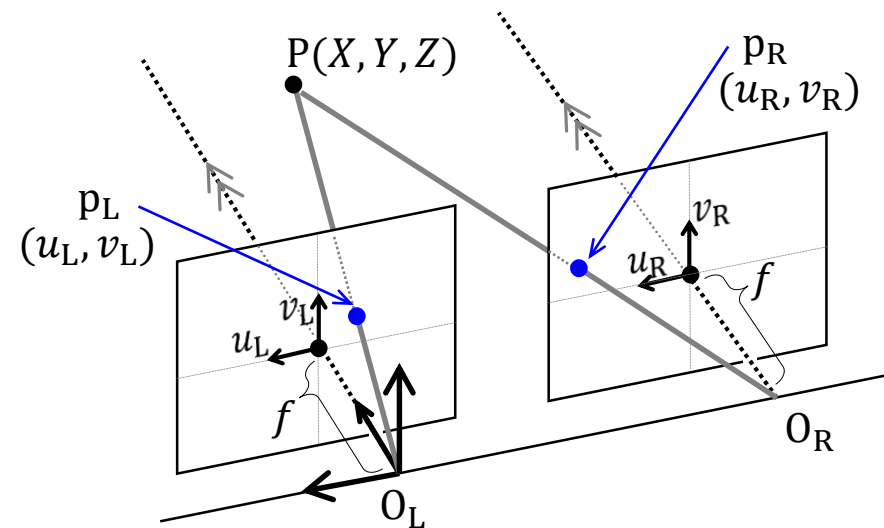
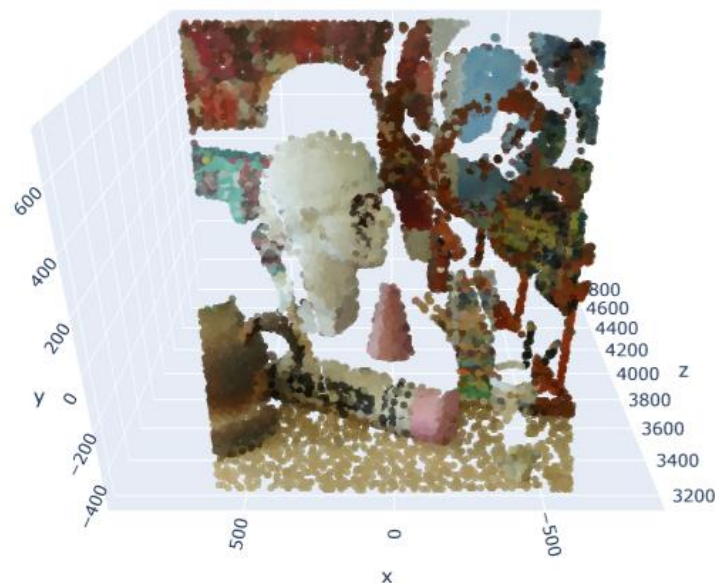
実験2日目

「逆透視変換」, 「ポイントクラウド」, 「RANSAC」
inverse perspective mapping point cloud random sample consensus

これらについて, What? / Why? / How? の
あらゆる疑問を解消する文書(実験レポート)を期待しています.

3次元計測

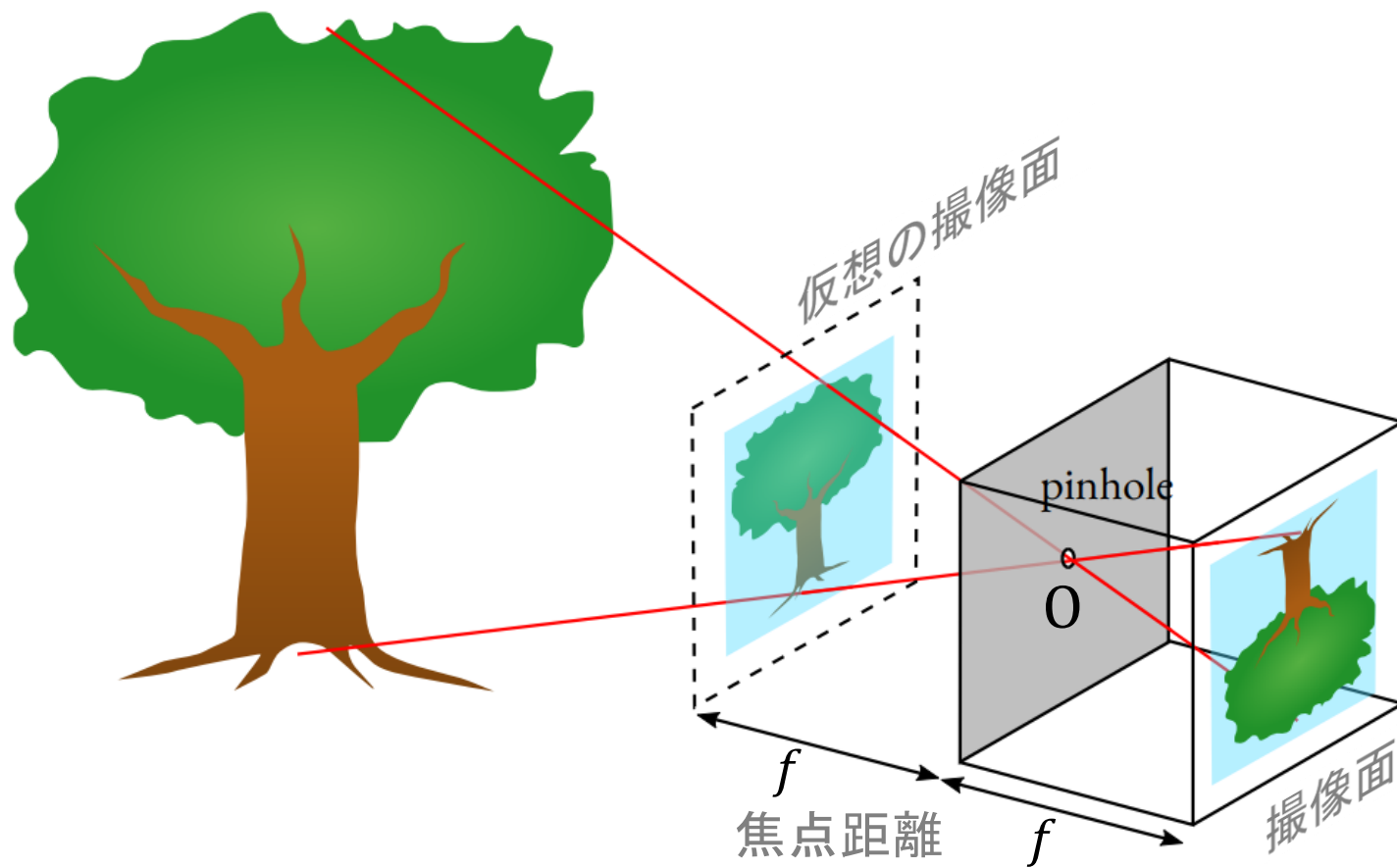
✓ 3次元化する



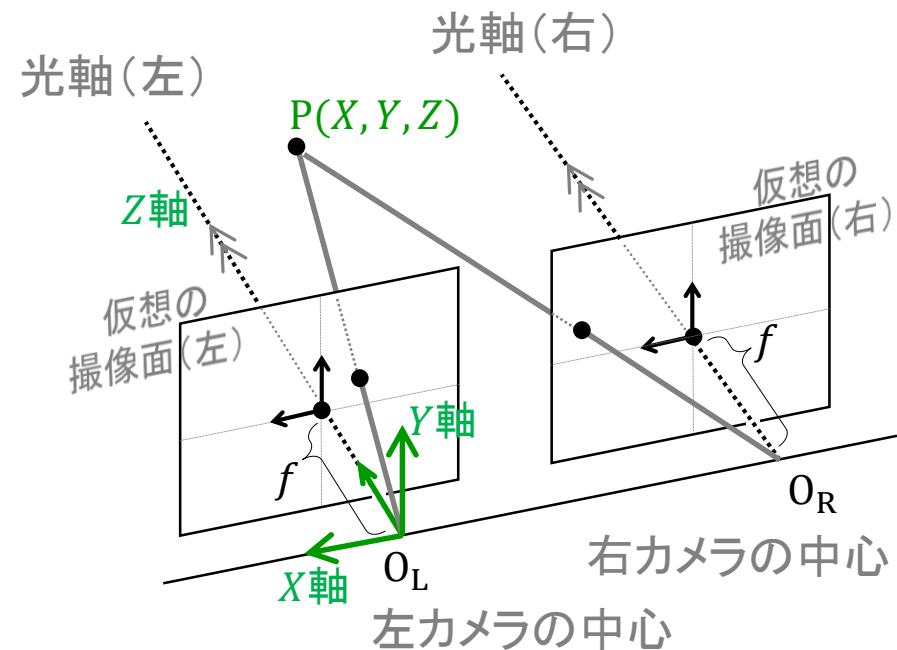
←
「視差」を測る
(ブロックマッチング)



画像とカメラの座標系



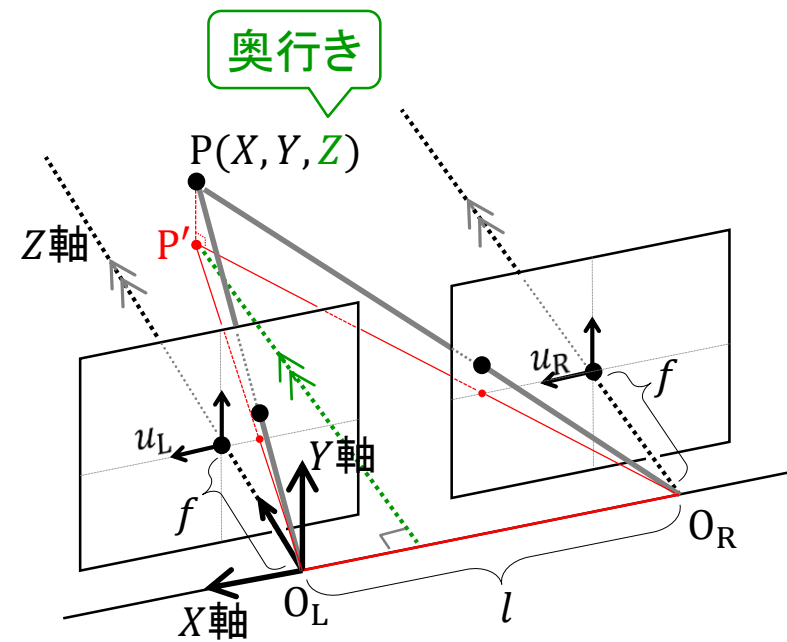
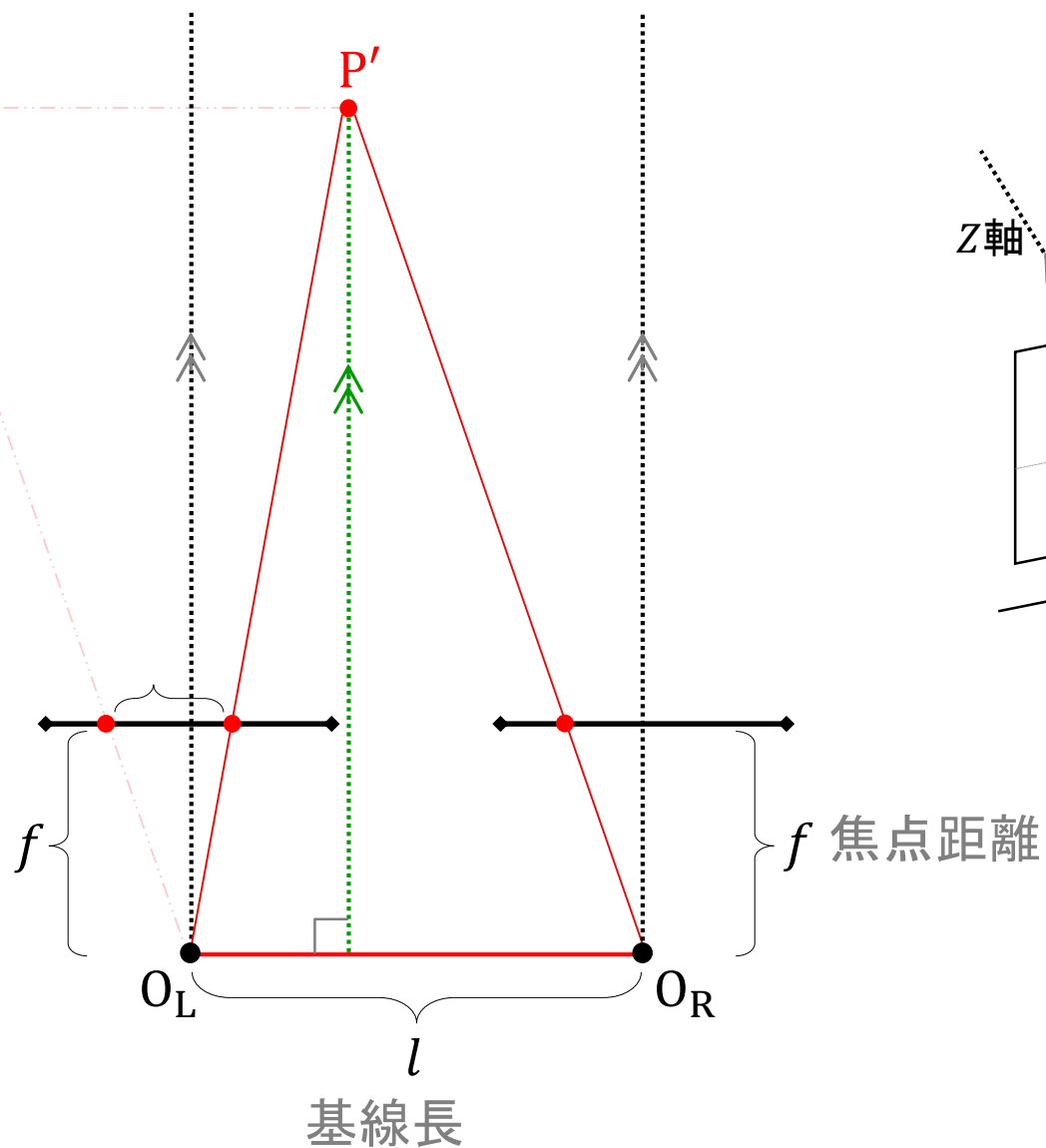
透視投影モデル



2台のカメラを
平行に設置した場合

視差 d と奥行き Z の関係

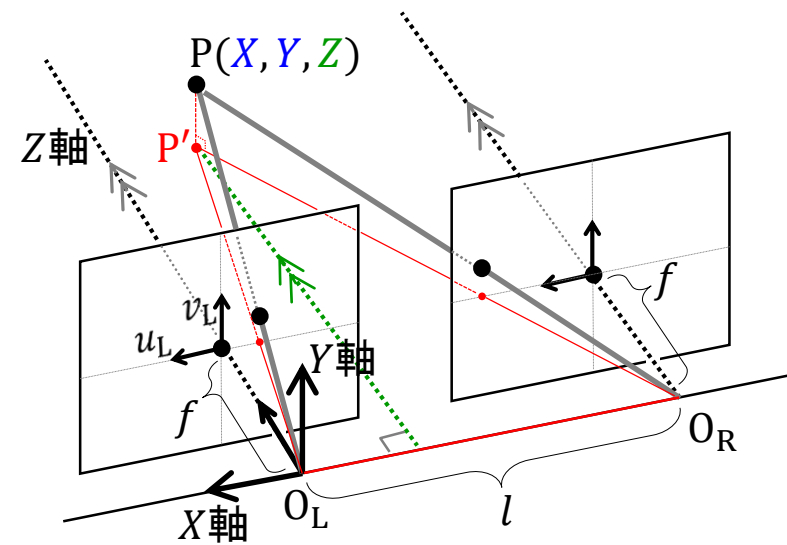
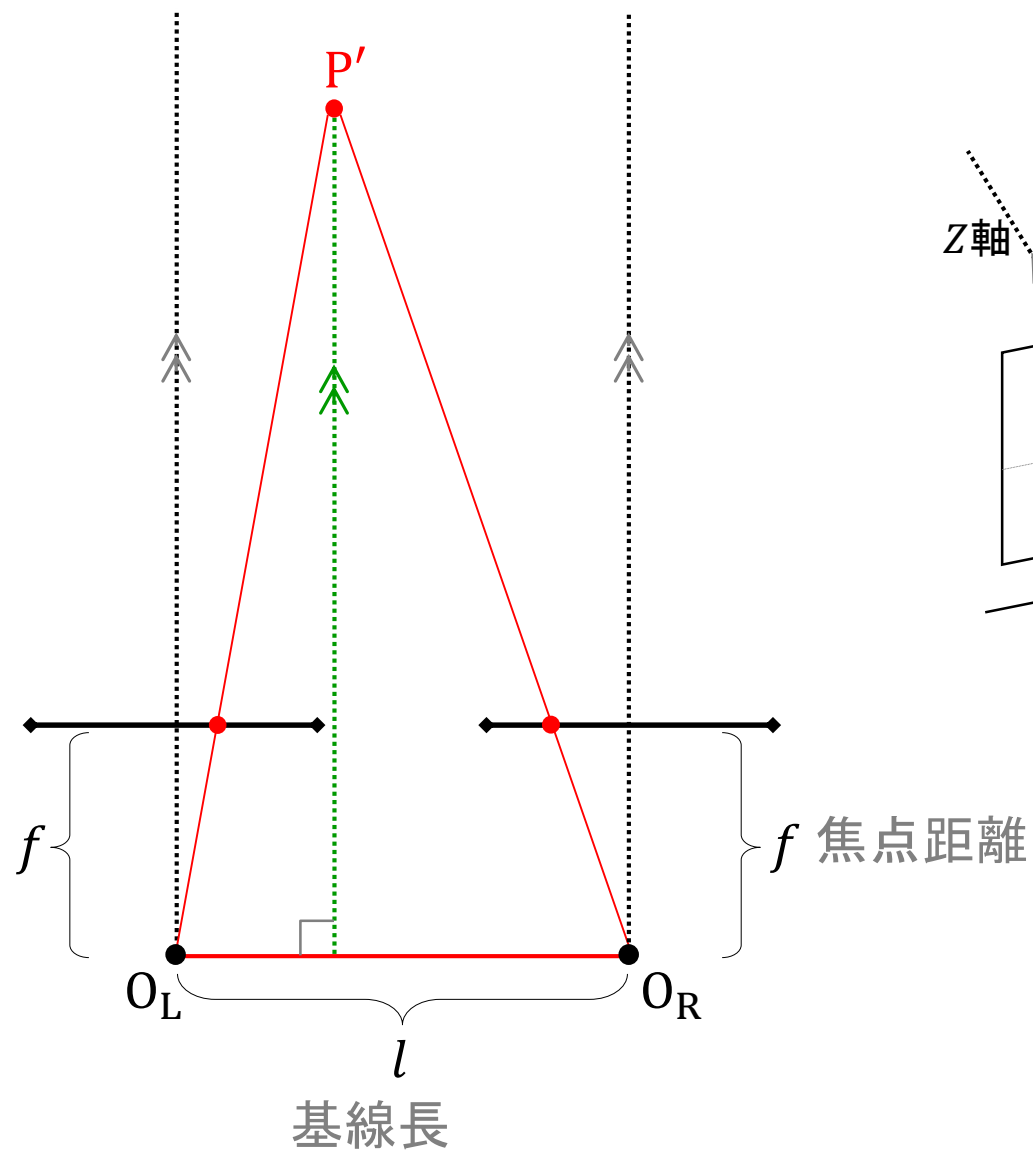
Z = 【視差 d , 焦点距離 f , 基線長 l で表せ】



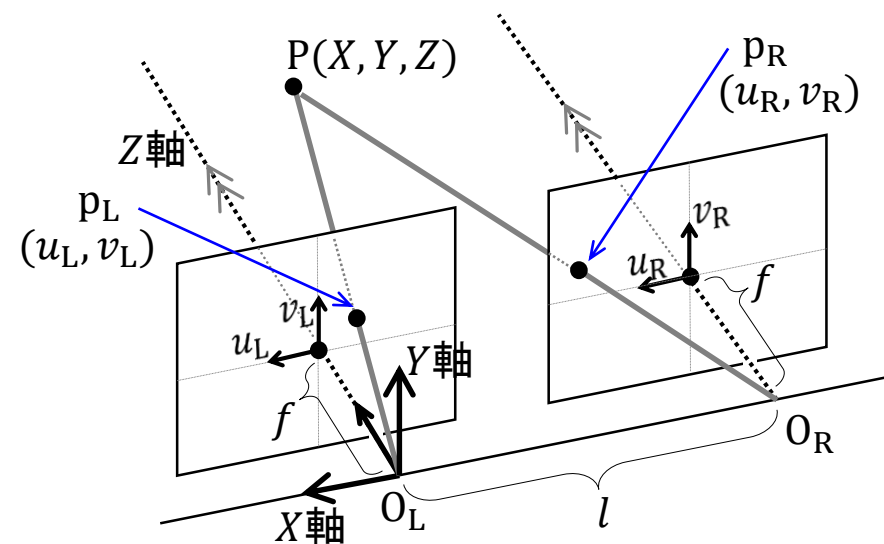
$$\text{視差 } d = |u_R - u_L|$$

X と Y も計算できる

$X = [Z, u_L, f \text{ で表せ}], Y = [Z, v_L, f \text{ で表せ}]$



3次元座標の計算手順



手順1 $p_L(u_L, v_L)$ に対応する $p_R(u_R, v_R)$ を見つける。

手順2 視差を測る。 $d = |u_R - u_L|$

手順3 深度(奥行き)に換算する。 $Z = \frac{f}{d} l$

手順4 3次元座標を得る。 $X = \frac{u_L}{f} Z$, $Y = \frac{v_L}{f} Z$ $(X, Y, Z) = \left(\frac{u_L}{d} l, \frac{u_R}{d} l, \frac{f}{d} l \right)$

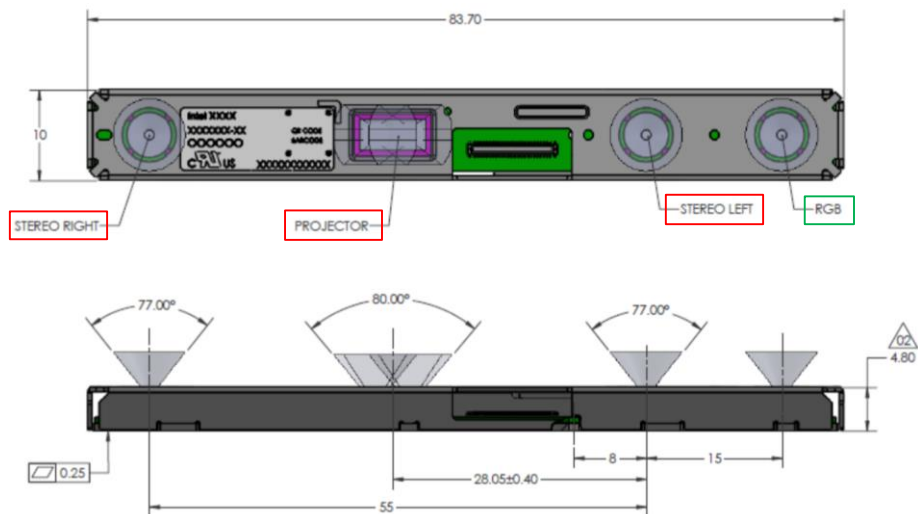
RGB-Dカメラを使ってみよう



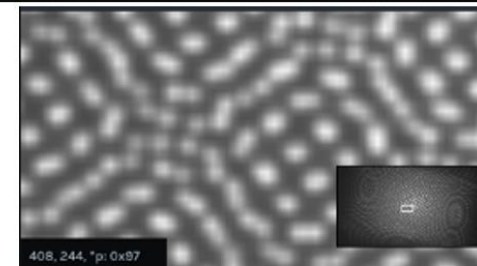
カラー画像 (RGB)



深度画像 (Depth)



D415 projector



深度から3次元座標を計算する



視差画像

手順1 $p_L(u_L, v_L)$ に対応する $p_R(u_R, v_R)$ を見つける。

手順2 視差を測る。

$$d = |u_R - u_L|$$

手順3 深度(奥行き)に換算する。

$$Z = \frac{f}{d} l$$

手順4 3次元座標を得る。

$$X = \frac{u_L}{f} Z, \quad Y = \frac{v_L}{f} Z$$

手順1～3の処理を経た
深度画像 $Z(u_L, v_L)$ を取得する。

3次元座標をたくさん調べたら・・・！

点群 (point cloud)



手順1 $p_L(u_L, v_L)$ に対応する $p_R(u_R, v_R)$ を見つける。

手順2 視差を測る。

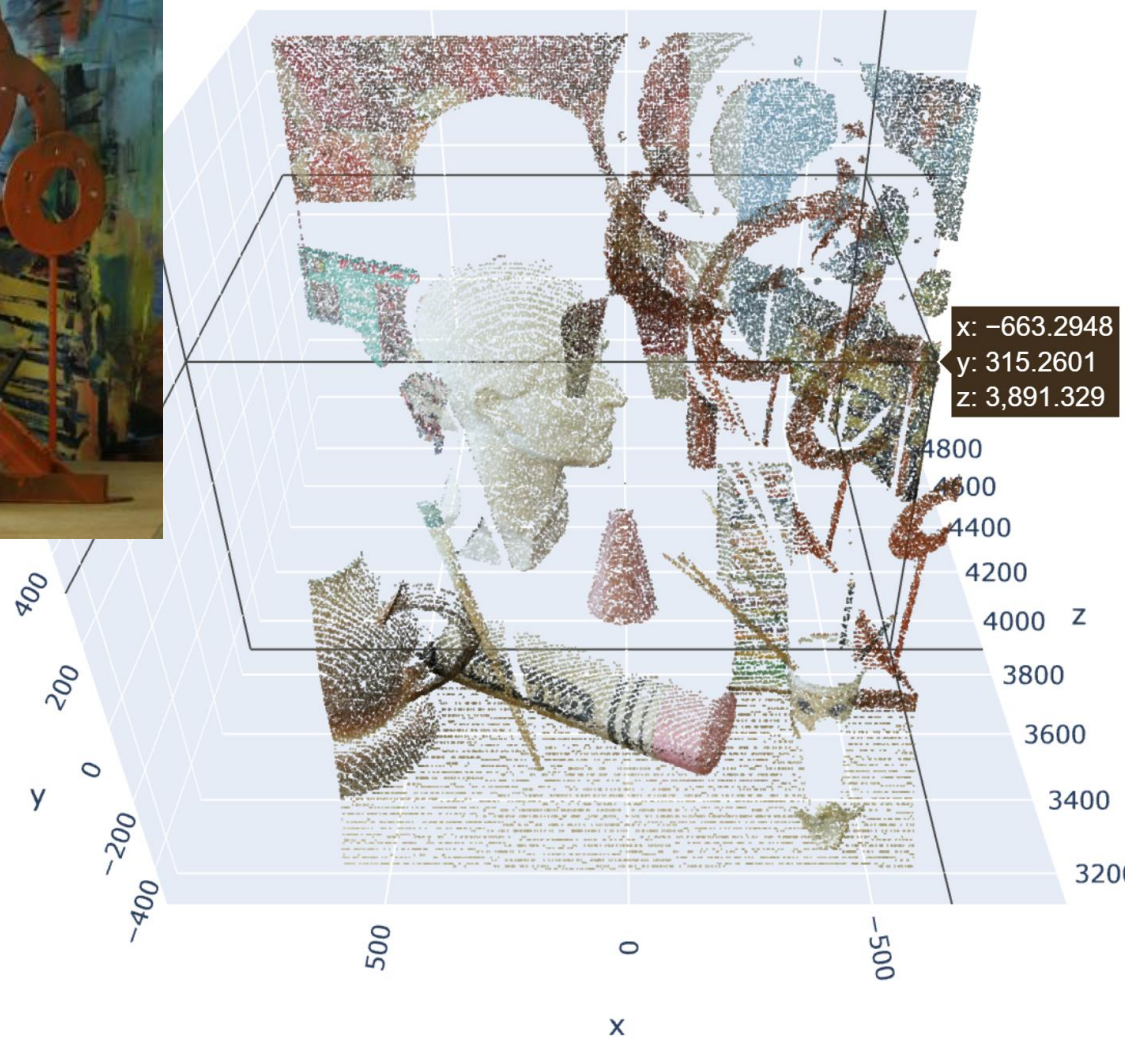
$$d = |u_R - u_L|$$

手順3 深度(奥行き)に換算する。

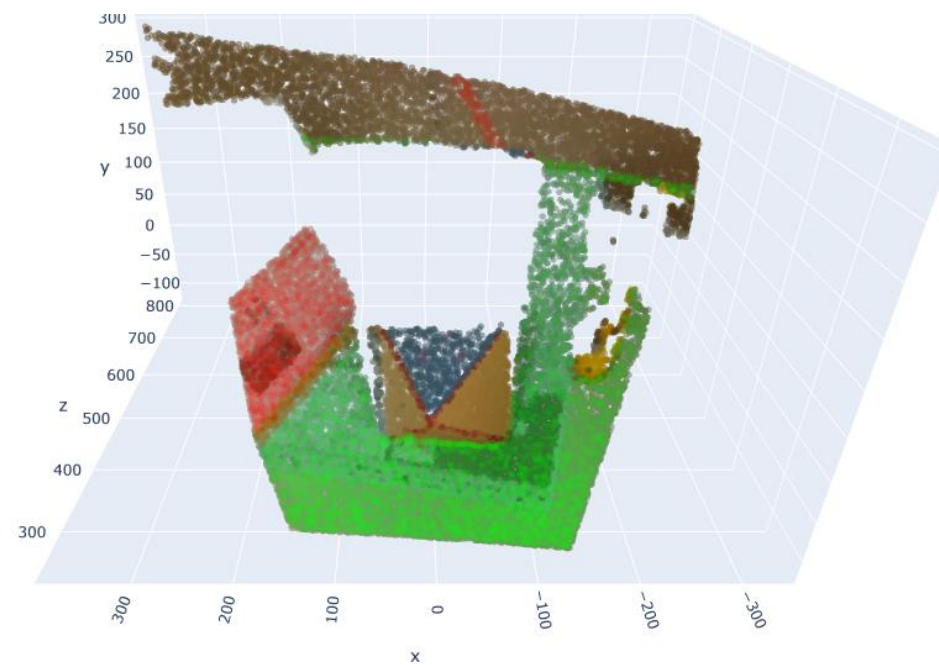
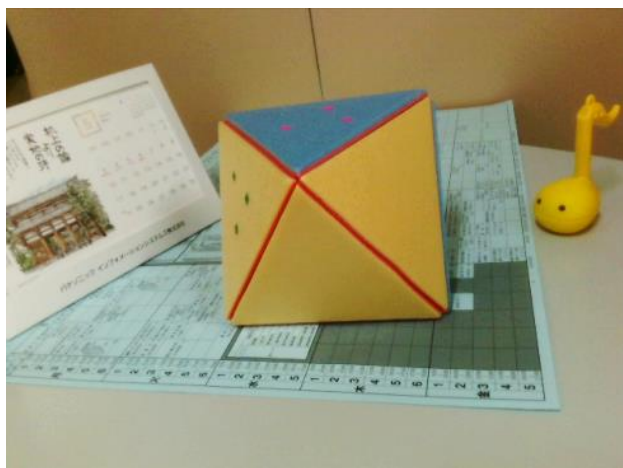
$$Z = \frac{f}{d} l$$

手順4 3次元座標を得る。

$$X = \frac{u_L}{f} Z, \quad Y = \frac{v_L}{f} Z$$

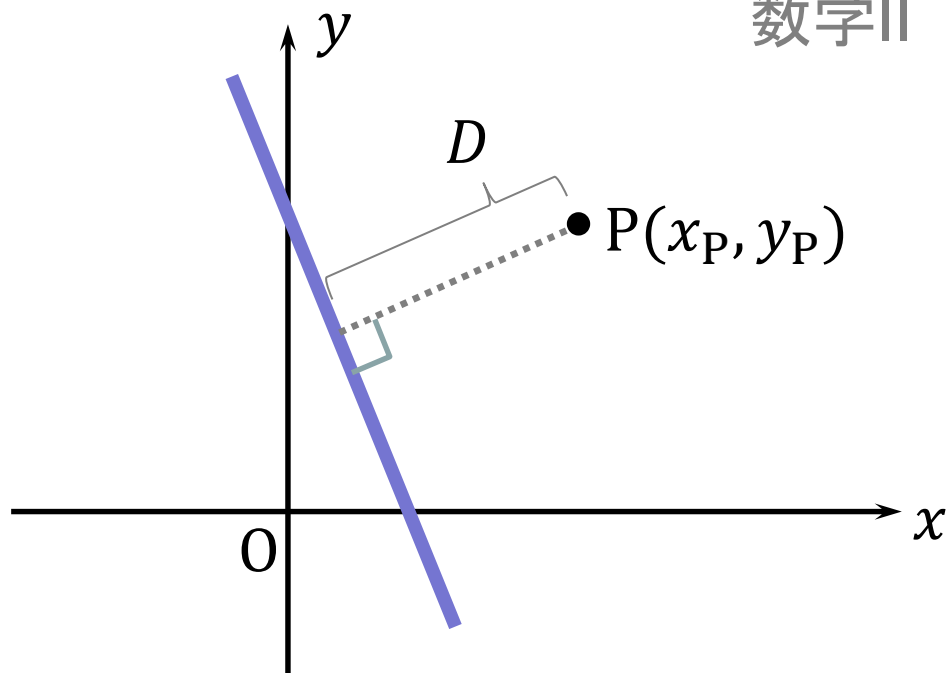


平面の自動検出



点と平面の距離

数学II

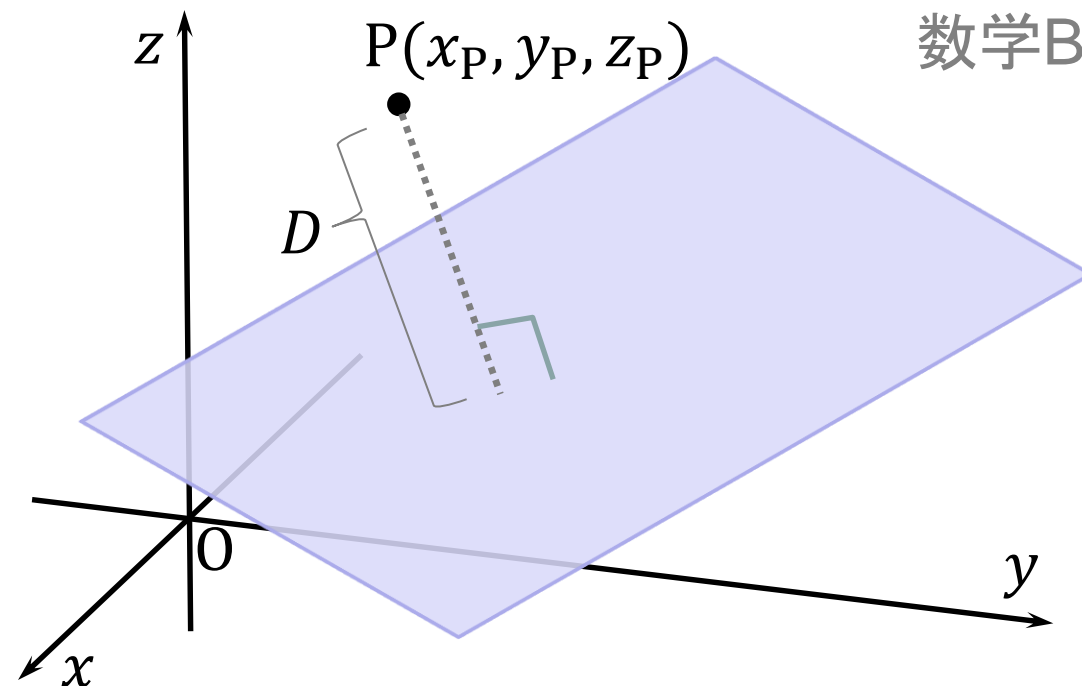


直線の方程式 $ax + by + c = 0$

点 $P(x_P, y_P)$ と直線の距離

$$D = \frac{|ax_P + by_P + c|}{\sqrt{a^2 + b^2}}$$

数学B



平面の方程式 $ax + by + cz + d = 0$

点 $P(x_P, y_P, z_P)$ と平面の距離

$$D = \frac{|ax_P + by_P + cz_P + d|}{\sqrt{a^2 + b^2 + c^2}}$$

平面を自動検出する

RANSAC

【平面を検出するRANSAC】 [F. T-Kurdi et al., 2008]

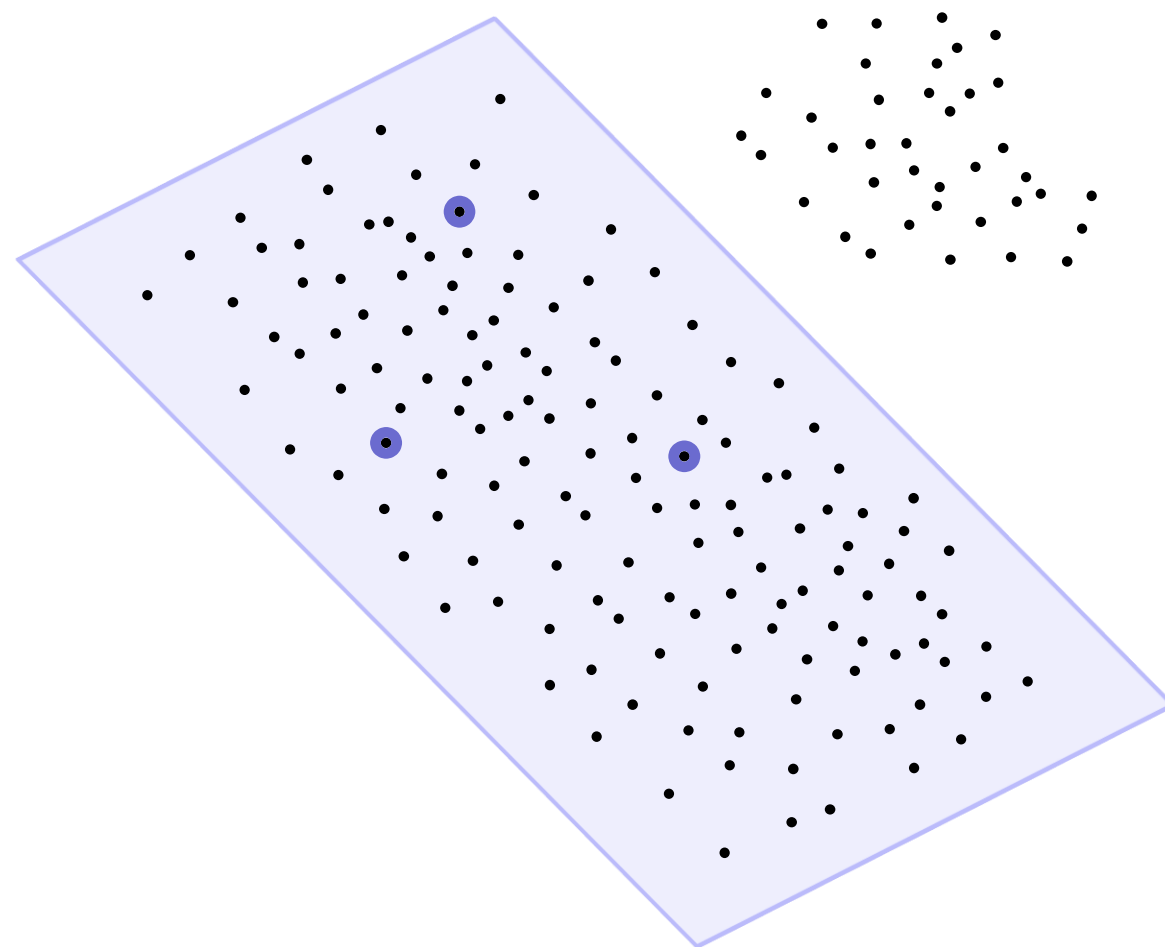
以下の手順1～4を繰り返す。

手順1 点群からランダムに3点を選ぶ。

手順2 3点を通る平面の方程式を作る。

手順3 平面にとっても近い点を数える。

手順4 点の数が記録更新なら,
その平面を覚えておく。



平面を自動検出する

【平面を検出するRANSAC】 [F. T-Kurdi et al., 2008]

以下の手順1～4を繰り返す。

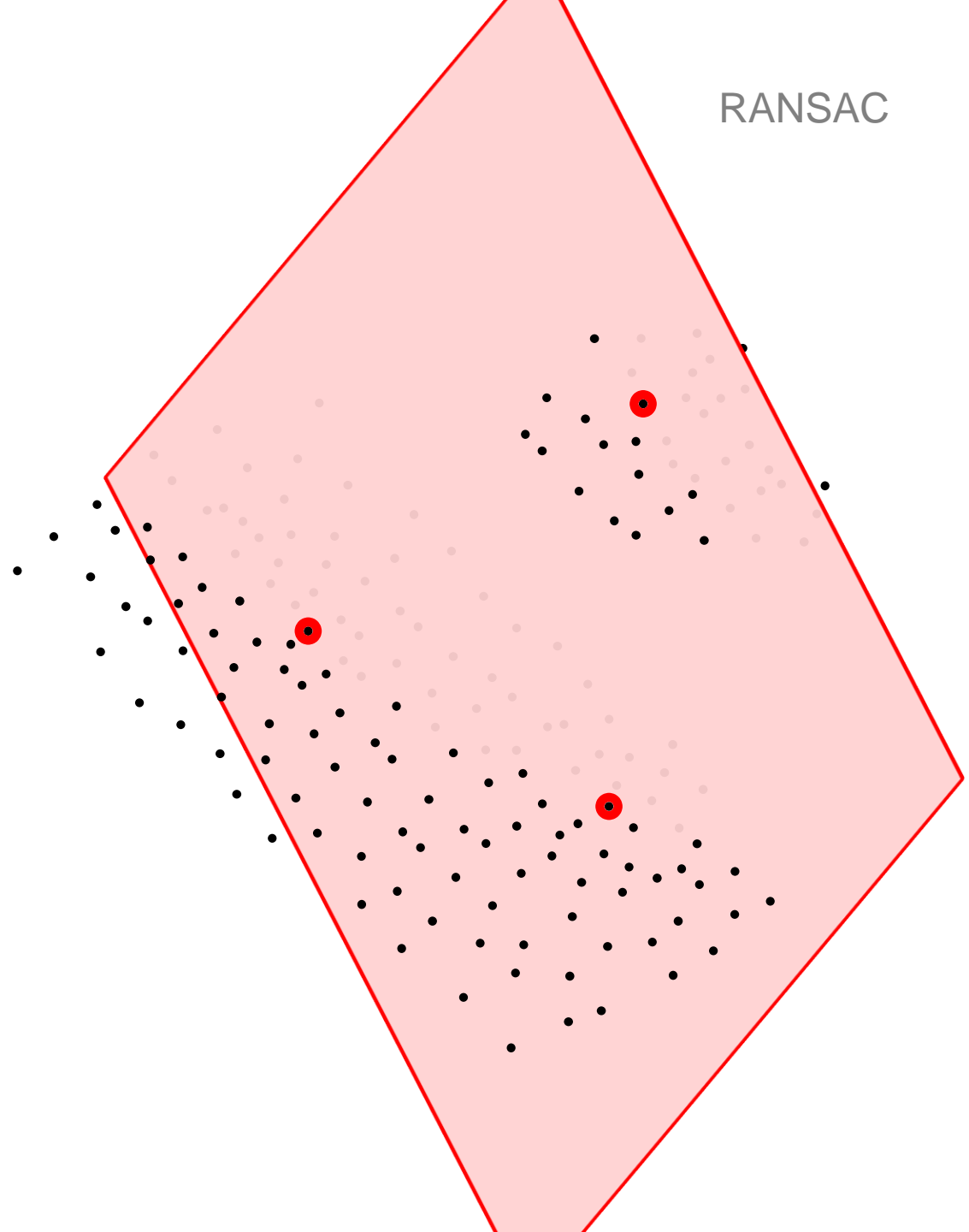
手順1 点群からランダムに3点を選ぶ。

手順2 3点を通る平面の方程式を作る。

手順3 平面にとっても近い点を数える。

手順4 点の数が記録更新なら,
その平面を覚えておく。

RANSAC



補遺: 平面と符号付き距離

(後期「パターン認識と機械学習」でも活躍します)

位置ベクトル

- 法線 \mathbf{w} , 通る点 \mathbf{c} の超平面: $\mathcal{P}(\mathbf{w}, \mathbf{c}) = \{ \mathbf{x} \mid \mathbf{w} \cdot (\mathbf{x} - \mathbf{c}) = 0 \}$

超平面 (hyperplane): 2次元空間の直線, 3次元空間の平面, ...

$$\mathbf{w} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

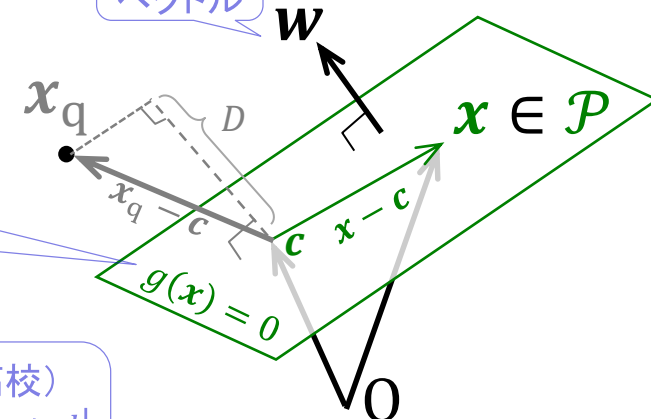
法線
ベクトル

• $g(\mathbf{x}) = \mathbf{w} \cdot (\mathbf{x} - \mathbf{c}) = -\mathbf{w} \cdot \mathbf{c} + \mathbf{w} \cdot \mathbf{x} = w_0 + w_1 x_1 + \dots + w_n x_n = 0$

x の一次関数

定数なので
 w_0 と置く

例 ($n = 3$ 次元): 平面の方程式
 $ax + by + cz + d = 0$



位置ベクトル

- 点 \mathbf{x}_q と超平面 $\mathcal{P}(\mathbf{w}, \mathbf{c})$ の符号付き距離 (signed distance)

$$D(\mathbf{x}_q, \mathcal{P}) = \left(\frac{\mathbf{w}}{\|\mathbf{w}\|} \right) \cdot (\mathbf{x}_q - \mathbf{c}) = \frac{\mathbf{w} \cdot (\mathbf{x}_q - \mathbf{c})}{\|\mathbf{w}\|}$$

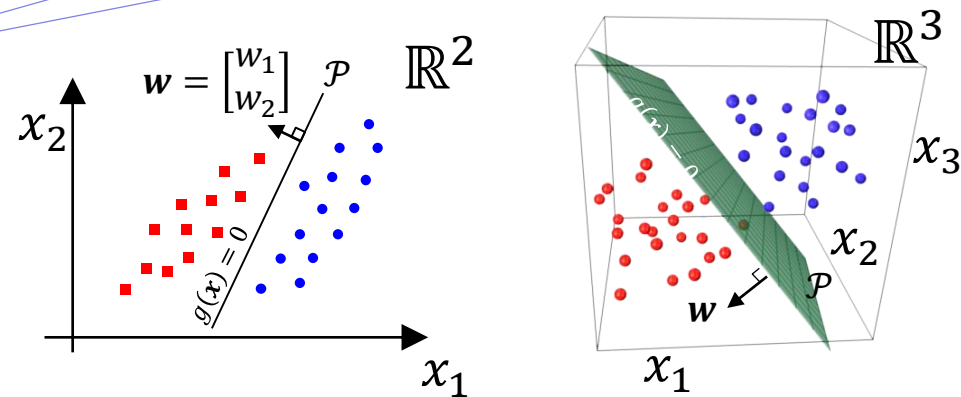
点と平面の距離 (高校)

$$D = \frac{|ax_q + by_q + cz_q + d|}{\sqrt{a^2 + b^2 + c^2}}$$

• $D(\mathbf{x}_q, \mathcal{P}) > 0 \Leftrightarrow g(\mathbf{x}_q) > 0 \Leftrightarrow \mathbf{x}_q$ は \mathbf{w} の正の側

$D(\mathbf{x}_q, \mathcal{P}) = 0 \Leftrightarrow g(\mathbf{x}_q) = 0 \Leftrightarrow \mathbf{x}_q \in \mathcal{P}$ (面上)

$D(\mathbf{x}_q, \mathcal{P}) < 0 \Leftrightarrow g(\mathbf{x}_q) < 0 \Leftrightarrow \mathbf{x}_q$ は \mathbf{w} の負の側



高次元データを符号で「識別」できます。

最後のメッセージ

情報科学・データ科学



『見る・聞く・考える』を『数学』に翻訳！

コンピュータ

(ﾟдﾟ)ウマー ⇒ 明日今日から本気出す！