03-MB-709.03

Echtzeitbildverarbeitung (11)

Prof. Dr. Udo Frese

RANSAC

Auffrischung Stochastik

Quadratische Ausgleichsrechnung

Was bisher geschah

- **Geometrische Rekonstruktion**
  - Kalibrierte Kamera ist ein Winkelmessgerät, eine Länge benötigt
  - Meist ist es so, wie die Zahl der Freiheitsgrade suggeriert.
  - Fluchtpunkt von parallelen Geraden hängt nur von Richtung ab.
  - Zwei Winkel zu Landmarken beschränkt die Position auf einen Kreis.
- **Kameragleichung in 3D (Parameter DOF)**
  - Transformation in Kamerakoordinaten $CInW^{-1}$ (6DOF)
  - Perspektive *p* (0 DOF)
  - Verzerrung *d* (üblich: 0-2DOF)
  - Skalierung/Offset (1/3DOF)

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} + \alpha \cdot d_{\kappa_1, \kappa_2}\left(p\left(CInW^{-1} \cdot p_W\right)\right)$$

RANSAC

RANSAC

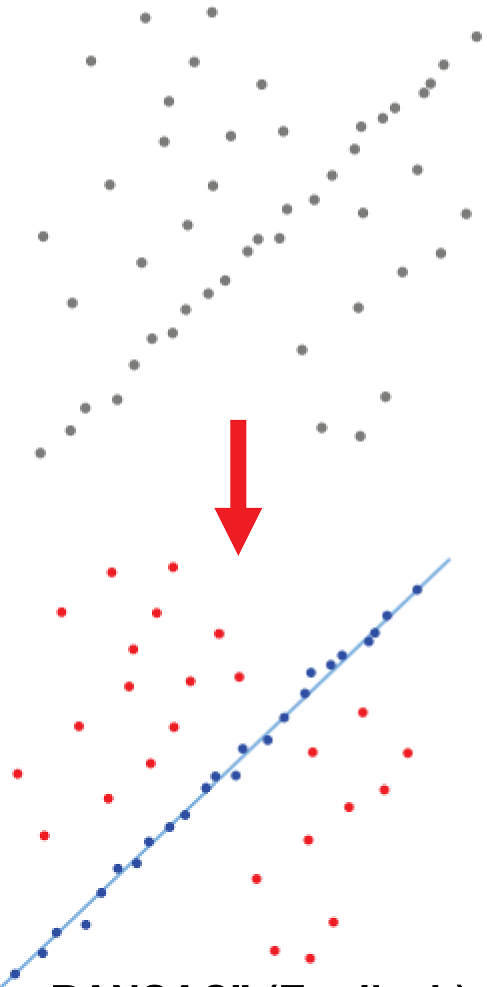
Aufgabe

- **Beobachtung: Geometrische Rekonstruktion einer Größe erfordert Zuordnung Weltpunkten zu Bildpunkten.**
  - oft Zuordnung unbekannt, oder unsicher
  - oft fehlende Bildpunkte
  - oft überzählige Bildpunkte
- **Aufgabe 1: Bestimme Größe durch am besten passende Zuordnungen aus einer Menge potentieller Zuordnungen**
- **Aufgabe 2: Bestimme Größe durch am besten passende Zuordnungen ohne Vorgabe (schwierig!)**
- **Übung: Größe ist Kamerakalibrierung, Zuordnung ist welche Linie des Gitters, keine Vorgabe**
- **Lösung: Random Sample Consensus (RANSAC)**

RANSAC

<u>Beispiel: RANSAC zur Suche nach einer Geraden</u>

- **Eingabe: Punkte**
  - von einer Geraden stammend aber ungenau
  - weitere Punkte (Ausreißer)
- **Ausgabe: Gerade (Größe),**
  - möglichst gut zu den Punkten passt
  - welche Punkte dazugehören (Zuordnung)
- **RANSAC**
  - Ziehe zufällig 2 Punkte
  - Berechne Geraden aus 2 Punkte
  - Zähle Punkte die zur Geraden passen
  - Wiederhole und wähle maximale Anzahl



Quelle: wikipedia „RANSAC" (Englisch)

RANSAC

Random Sample Consensus (RANSAC)

- **Idee: Für eine hypothetische Größe lässt sich prüfen, ob eine Zuordnung zu ihr passt.**
- **Übung: Setze Weltkoordinate für Linie und Kamerakalibrierung in Kameragleichung ein und vergleiche Bildlinie mit Schwellwert**
- **Anzahl der Zuordnungen die passen (*inlier*) ist Indikator für Hypothese**
- **Berechne hypothetische Größe aus zufällig gezogenen Zuordnungen**
  - In jeder Zuordnung so viele Elemente wie für DOF nötig
  - Wiederhole mehrfach
  - Ergebnis ist die Hypothese mit den meisten passenden Zuordnungen

RANSAC

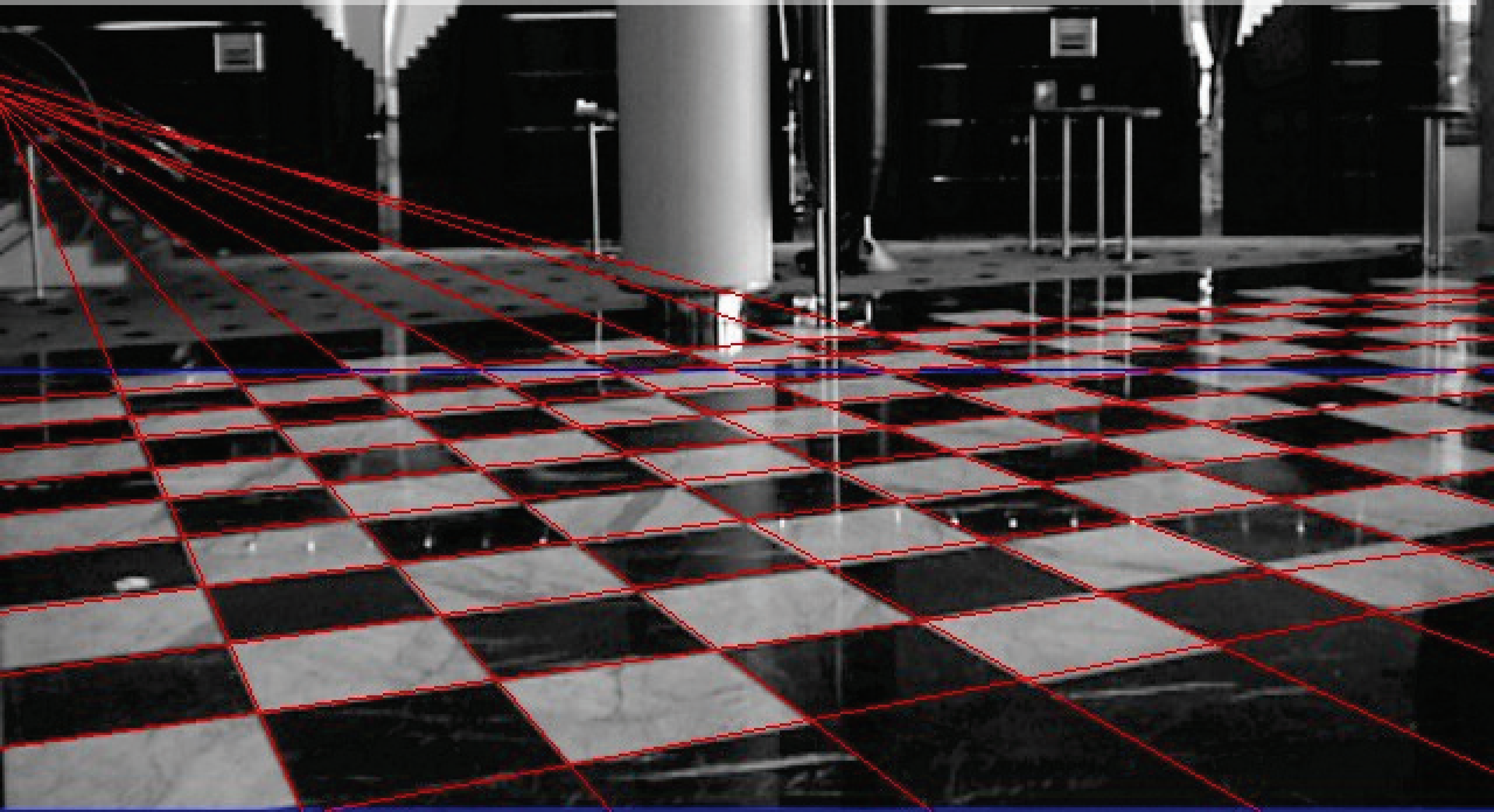
```
int count (vector<Association> ass, Parameters p)
{
   int ctr=0;
   for (int i=0; i<ass.size(); i++)
      if (p.compatibilityError(ass[i])<=THRESHOLD) ctr++;
   return ctr;
}

RANSAC (vector<Association> ass, Parameters& p)
{
   int maxCount = -1;
   for (int i=0; i<NSAMPLES; i++) {
      vector<Association> sample;
      for (int j=0; j<Parameters::NASSPERSAMPLE; j++)
         sample.push_back (ass[rand()%ass.size()]);
      Parameters hypP;
      hypP.computeFrom (sample);
      int cnt = count (ass, hypP);
      if (cnt>maxCount) {maxCount=cnt; p=hypP;}
   }
}
```

Beispiel: RANSAC zur Suche nach einer Geraden

```
typedef Point2D Association;

//!  Line in Hesse normal form (x·cos(α) + y·sin(α) - d=0)
class Parameters {
   //!  Angle of normal and signed distance to origin
   double alpha, d;

   //!  2 points needed for defining a line
   enum {NASSPERSAMPLE=2};

   //!  Line from two points
   void computeFrom (vector<Point2D> p12) {
      alpha = atan2 (p[1].y-p[0].y, p[1].x-p[0].x)+M_PI/2;
      d     = p[0].x*cos(alpha) + p[0].y*sin(alpha);
   }

   //!  Distance p to *this
   double compatibilityError (Point2D p) {
     return fabs(cos(alpha)*p.x + sin(alpha)*p.y - d);
   }
};
```

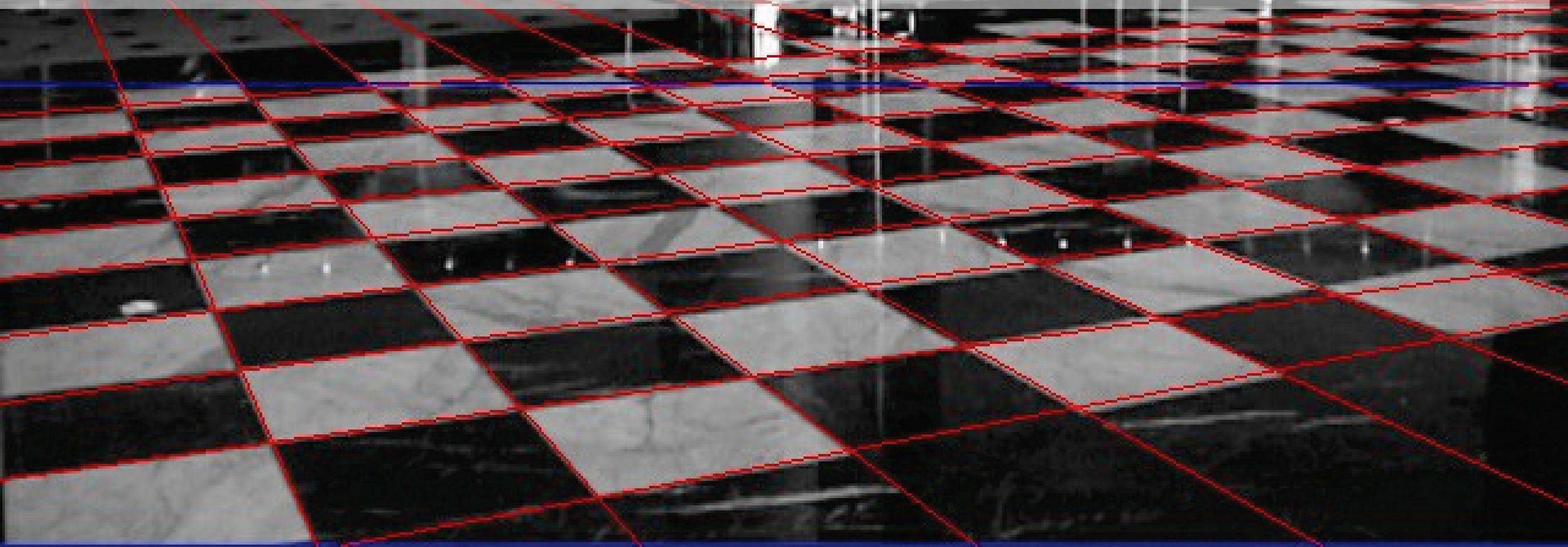
Frage an das Auditorium: Wie setzt man RANSAC auf das Problem Gittergeraden→ Kalibrierung (Pose, Brennweite) an?

- Was ist eine Assoziation?
- Wie viele muss man ziehen?
- Wie definiert man inlier?



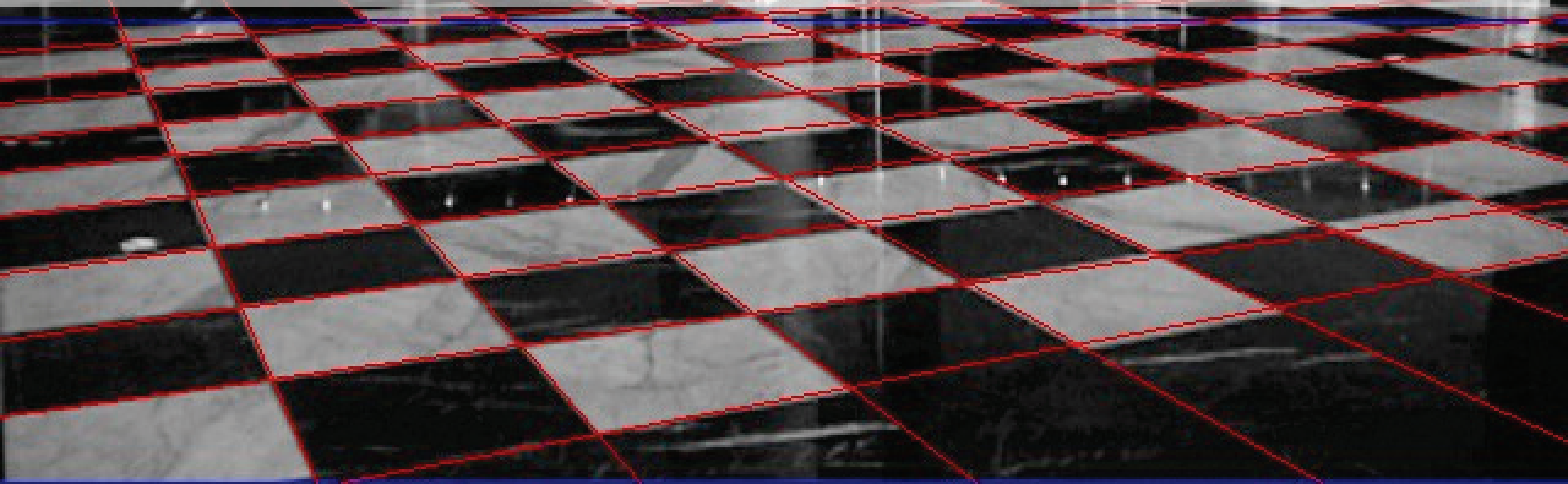
<u>Frage an das Auditorium</u>: Wie setzt man RANSAC auf das Problem Gittergeraden→ Kalibrierung (Pose, Brennweite) an?

- **Was ist eine Assoziation?**
  - Paar aus Bildgerade und Weltgerade (Z=0, X=i*d oder Y=i*d, $i \in Z$)
- **Wie viele muss man ziehen?**
  - 4 Geraden → 4 Schnittpunkte → Kalibrierung
- **Wie definiert man inlier?**
  - Weltgerade der Assoziation mit Kalibrierung ins Bild projizieren
  - Schwellwert in α und d, z.B. den in d als Bilddiagonale*Schwellwert α
- **Wo liegt das Problem?**



Frage an das Auditorium: Wie setzt man RANSAC auf das Problem Gittergeraden→ Kalibrierung (Pose, Brennweite) an?

- **Was ist eine Assoziation?**
  - Paar aus Bildgerade und Weltgerade ($Z=0$, $X=i*d$ oder $Y=i*d$, $i \in Z$)
- **Wie viele muss man ziehen?**
  - 4 Geraden → 4 Schnittpunkte → Kalibrierung
- **Wie definiert man inlier?**
  - Weltgerade der Assoziation mit Kalibrierung ins Bild projizieren
  - Schwellwert in α und d, z.B. den in d als Bilddiagonale*Schwellwert α
- **Wo liegt das Problem?**
- **Wahrscheinlichkeit für 4 richtige Zuordnung ca. $<16^{-4}$ =15E-6**



RANSAC

Verbessertes Ziehen der Zuordnungen bei Kalibrierung

- Wahl des Koordinatenursprungs ist willkürlich
- Ziehe Linie als X=Z=0 Achse
- Ziehe Linie als Y=Z=0 Achse
- finde Nachbarlinien als Z=0, X= bzw. Y=d Linie
- Wahrscheinlichkeit für richtige Wahl <¼
- Wahrscheinlichkeit für gute Wahl etwas kleiner

RANSAC

RANSAC ohne Vorgabe von potentiellen Zuordnungen

- **übernehme Idee, modifiziere Ziehen und Zählen geschickt.**
- **Beispiel Kamerakalibrierung von Liniengitter (Übung)**
  - ziehe zufällig vier Bildlinien (jeweils eine wirklich zufällig und deren Nachbarin)
  - berechne 4 Schnittpunkte
  - berechne Kamerakalibrierung aus den 4 Schnittpunkten
  - laufe durch alle Linien X=-7..+7 und Y=-7..+7
  - projiziere Linie mit Kamerakalibrierung
  - ist eine Bildlinie ähnlich genug (Schwellwert Distanz, Winkel) so passt sie
  - passen mehrere, wähle ähnlichste
  - wähle Kalibrierung bei der am meisten passen

Auffrischung Stochastik

Auffrischung Stochastik

- **Zufallsexperiment: Ein Vorgang der einen nicht vorhersagbaren beobachtbaren Ausgang hat. Hier pragmatisch aufgefasst. Z.B. das Werfen zweier Würfels.**
- **Zufallsvariablen X, Y sind Größen, die von dem Ausgang eines Zufallsexperimentes abhängen. Z.B. Augensumme.**
- **p(X=x) ist die Wahrscheinlichkeit(-sdichte) dafür, dass die Zufallsvariable X, den Wert x hat. Es ist also eine Funktion von x.**
- **Wichtige Unterscheidung: X ist eine Zufallsvariable, also eine Größe, die vom Zufallsexperiment abhängt. x ist einfach nur eine Zahl. So wie p(X=0), p(X=7)**

Auffrischung Stochastik

- p(X=x, Y=y) ist die Wahrscheinlichkeit(-sdichte), dass X den Wert x hat und Y den Wert y hat.
- p(X=x|Y=y) = p(X=x, Y=y) / p(Y=y) ist die Wahrscheinlichkeit(-sdichte), dass X den Wert x hat, wenn Y schon den Wert y hat.
- Zwei Zufallsvariablen sind stochastisch unabhängig, wenn p(X=x, Y=y) = p(X=x) p(Y=y)
- Kurzschreibweise p(x) statt p(X=x), etc.
- Schreibweise: Bei kontinuierlichen Zufallsexperimenten p(…) für Wahrscheinlichkeitsdichten, bei diskreten P() für Wahrscheinlichkeiten.

Auffrischung Stochastik

- **Zufallsexperiment: Werfen zweier Würfel (einer weiß, einer rot)**
- **Zufallsvariable: X Augen des roten Würfel, Y Augen des weißen Würfel, Z=X+Y: Augensumme**

<u>Frage an das Audi-torium</u>: Welche Wahr-scheinlich-keiten haben die verschie-denen Ereig-nisse?

P(X=x, Y=y)=

	P(X=_)	P(Y=_)	P(Z=_)	P(Z=_ |X=3)	P(Z=_ |Y=4)	P(Z=_ |X=4,Y=3)
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						

Auffrischung Stochastik

- **Zufallsexperiment: Werfen zweier Würfel (einer weiß, einer rot)**
- **Zufallsvariable: X Augen des roten Würfel, Y Augen des weißen Würfel, Z=X+Y: Augensumme**

	P(X=_)	P(Y=_)	P(Z=_)	P(Z=_ |X=3)	P(Z=_ |Y=4)	P(Z=_ |X=4,y=3)
1	1/6	1/6	0	0	0	0
2	1/6	1/6	1/36	0	0	0
3	1/6	1/6	2/36	0	0	0
4	1/6	1/6	3/36	1/6	0	0
5	1/6	1/6	4/36	1/6	1/6	0
6	1/6	1/6	5/36	1/6	1/6	0
7	0	0	6/36	1/6	1/6	1
8	0	0	5/36	1/6	1/6	0
9	0	0	4/36	1/6	1/6	0
10	0	0	3/36	0	1/6	0
11	0	0	2/36	0	0	0
12	0	0	1/36	0	0	0

P(X=x, Y=y)=1/36, für alle x,y∈[1..6] sonst 0

Quadratische Ausgleichsrechnung

Quadratische Ausgleichsrechnung

Aufgabe

- **bestimme Parameter eines Messmodells so, dass Vorhersagen möglichst gut mit Messungen übereinstimmen.**
- **Ziel: Mehr Messungen sollten die Genauigkeit verbessern (Ausgleichsrechnung), z.B. gegenüber RANSAC**
- **Anwendung Kalibrierung:**
  - bestimme Kamerapose *CInW*, Verzerrung κ und Brennweite f_{eff} so, dass Projektionen bekannter Punkte möglichst gut mit Bildpunkten übereinstimmen.
- **Anwendung Kamera / Objektlage:**
  - wie oben, aber κ und f sind fest.
- **Anwendung 3D Rekonstruktion :**
  - bestimme Kameraposen *C2InW* und Punkte, so dass deren Projektion möglichst gut mit Bildpunkten übereinstimmen.
  - heißt auch Simultaneous Localization and Mapping (SLAM), Structure from Motion (SfM) oder Bundle Adjustment

Quadratische Ausgleichsrechnung

Praxis

- die Theorie ist hilfreich, aber nicht ganz leicht zu verstehen.
- deshalb zuerst: Was rechnen wir am Ende praktisch?

$$\hat{x} = \arg\min_x \sum_i \frac{\left(z_i - f_i(x)\right)^2}{\sigma_i^{\,2}}$$

- wahrer Parameter X gesucht
- Messungen z_i
- Messfunktion(-en) f_i, d.h. „wenn die Parameter x wären, müssten wir eigentlich $Z_i=f_i(x)$ messen“
- σ_i Unsicherheit der Messung Z_i
- x^ ist Schätzung für X
- Frage an das Autitorium: Könnt Ihr die Formel in Worte fassen?

Quadratische Ausgleichsrechnung

Praxis

$$\hat{x} = \arg\min_x \sum_i \frac{(z_i - f_i(x))^2}{\sigma_i^2}$$

- **wahre Parameter X, gesucht**
- **x^ Schätzung für X**
- **Messungen Z_i**
- **Messfunktion(-en) f_i, d.h. „wenn der Zustand x wäre, müssten wir eigentlich $Z_i=f_i(x)$ messen"**
- **σ_i Unsicherheit der Messung Z_i**
- **Formel in Worten**
  - Klammer ist Differenz zwischen, was wir bei Parameter x hätten messen sollen und was wir gemessen haben.
  - Division durch σ_i^2 macht die Differenz relativ zur Ungenauigkeit des Sensors.
  - dadurch Maß für Plausibilität von x unter der Messung Z_i.
  - Summe ist Gesamtplausibilität von x unter allen Messungen.
  - wir suchen den plausibelsten Zustand.

Quadratische Ausgleichsrechnung

Ansatz

- **Modelliere Messvorgang als Zufallsexperiment, bei dem auf den „wahren“ Messwert der sich aus den „wahren“ Parametern ergibt eine zufällige Störung addiert wird. Frage dann nach den *wahrscheinlichsten* Parametern gegeben die Messungen.**

$$Z_i = f_i(X) + N_i$$

- **wahre Parameter X, Messungen Z_i, Messfunktion(-en) f_i, Störung N_i**

Quadratische Ausgleichsrechnung

Herleitung

$$Z_i = f_i(X) + N_i$$

- **bekannt: $Z_i = z_i$, f_i Unbekannt: X, N_i**
- **verschiedene f_i für verschiedene Messwerte möglich.**
  - verschiedene Teile einer Messung, z.B. X- und Y- Koordinate eines Bildpunktes
  - verschiedene Sensoren, z.B. Winkelpeilung und Radarentfernung
  - abhängig von Parameter $f_i(X) = f(p_i, X)$, z.B. Messung von Bildpunkten verschiedener Raumpunkte p_i.
- **Annahme: Störungen (Messungen) stochastisch unabhängig.**

Quadratische Ausgleichsrechnung

Herleitung

- Gesucht: arg $\max_x$ p(X=x|Z=z),
- also der Parametersatz, mit der höchsten Wahrscheinlichkeit unter der Bedingung dass das gemessen wurde, was gemessen wurde.
- Berechnung über Bayes Theorem:

$$p(X=x|Z=z)\,p(Z=z) = p(X=x, Z=z) = p(Z=z|X=x)\,p(X=x)$$

Wahrscheinlichkeit, dass X den Wert x hat, gegeben, dass Z den Wert z hat

Wahrscheinlichkeit, dass Z den Wert z hat

Wahrscheinlichkeit, dass X den Wert x hat und dass Z den Wert z hat

Quadratische Ausgleichsrechnung

A-posteriori Wahrscheinlichkeit

$$\arg\max_x p(X = x | Z = z) = \arg\max_x \frac{p(Z = z | X = x)\, p(X = x)}{p(Z = z)}$$

$$= \arg\max_x p(Z = z | X = x)\, p(X = x) = \arg\max_x p(Z = z | X = x)$$

Keine a-priori Info

P(Z=z) ist gleich für alle *x*

Wie wahrscheinlich ist es, *Z*=z zu messen, wenn *X*=*x* wäre? (likelihood von x)

Wie wahr-scheinlich ist *x* grundsätzlich?

Quadratische Ausgleichsrechnung

$$\arg\max_x p(Z = z | X = x) = \arg\max_x p(f(X) + N = z | X = x)$$

$$= \arg\max_x p(N = z - f(X) | X = x) = \arg\max_x p(N = z - f(x))$$

- **Anschaulich:**
  - für ein hypothetisches *x* wissen wir, was bei den Messung hätte herauskommen sollen, nämlich *f*(*x*).
  - wir wissen, was herausgekommen ist, nämlich *Z*=z.
  - wir kennen den Meßfehler: *z*- *f*(x).
  - wie wahrscheinlich ist solch ein Meßfehler?
- **Messungen stochastisch unabhängig:**

$$= \arg\max_x p(N_i = z_i - f_i(x) \forall i) = \arg\max_x \prod_i p(N_i = z_i - f_i(x))$$

Quadratische Ausgleichsrechnung

- **Modell des Meßrauschens:**
- **n_i hat Gaußverteilung mit Mittelwert μ_i=0 und Standardabweichung σ_i.**

$$p(N_i = n_i) = \frac{1}{\sigma_i \sqrt{2\pi}} e^{-\frac{(n_i - \mu_i)^2}{2\sigma_i^2}}$$

$$= \arg\max_x \prod_i e^{-\frac{(z_i - f_i(x))^2}{2\sigma_i^2}} = \arg\min_x \sum_i \frac{(z_i - f_i(x))^2}{\sigma_i^2}$$

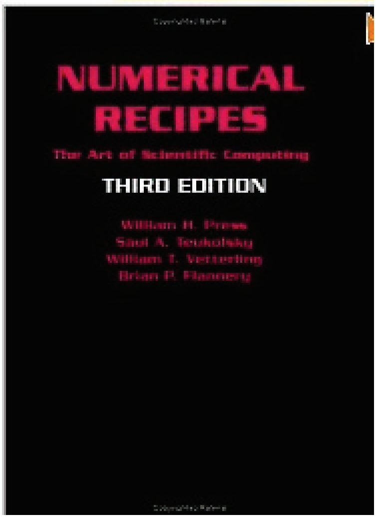
Quadratische Ausgleichsrechnung

```
class Measurement {
   double z; // Messwert
   double sigma; // Messungenauigkeit (std. Abweichung)
   double f(const vector<double>& x) const; // Messfunktion
};

double lsError (vector<Measurements>& meas, vector<double>& x){
  double sum=0;
  for (int i=0; i<meas.size(); i++) {
      double delta = (meas[i].z-meas[i].f(x))/sigma[i];
      sum += delta*delta;
  }
  return sum;
}
```

Quadratische Ausgleichsrechnung

Nichtlineare Optimierung

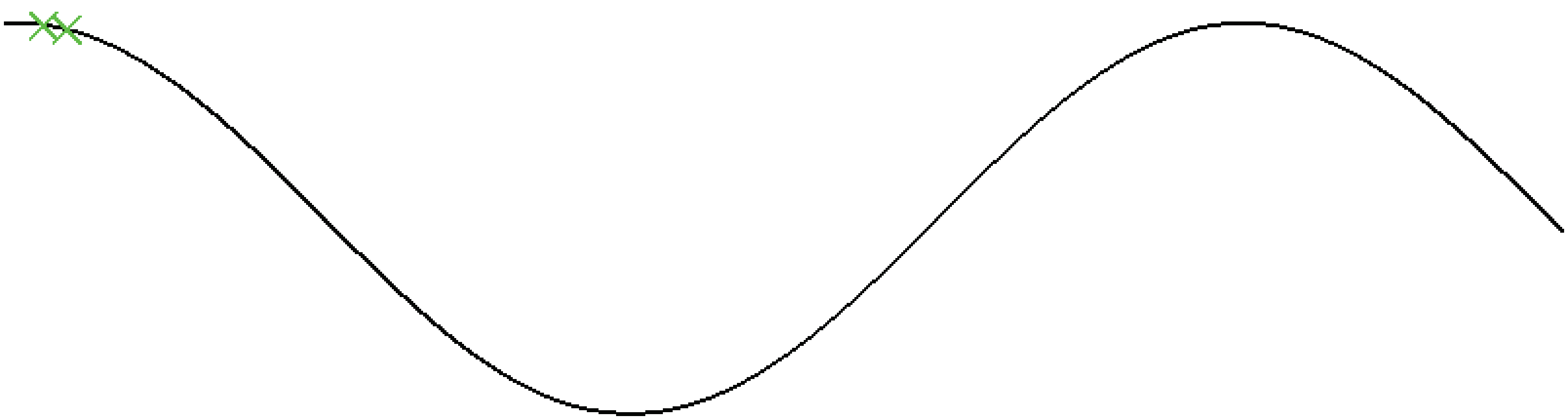
- **Gegeben Funktion $g: R^n \rightarrow R$ (+meist Startwert x_0)**
- **Gesucht $\text{argmin}_x\, g(x)$**
- **Forschungsgebiet für sich [1, Kap. 10, 15]**
  - nichtlineare Programmierung, NLP
- **hier extrem einfache, sehr ineffiziente Lösung**
- **besser: Levenberg-Marquard [1, Kap. 15.5]**
- **SLoM Bibliothek, www.openslam.org/slom/**
- **[1] Numerical Recipes 3rd Edition: The Art of Scientific Computing von William H. Press, Saul A. Teukolsky, William T. Vetterling und Brian P. Flannery von Cambridge University Press, 2007**



Quadratische Ausgleichsrechnung

Idee für einen SEHR einfachen Optimierer

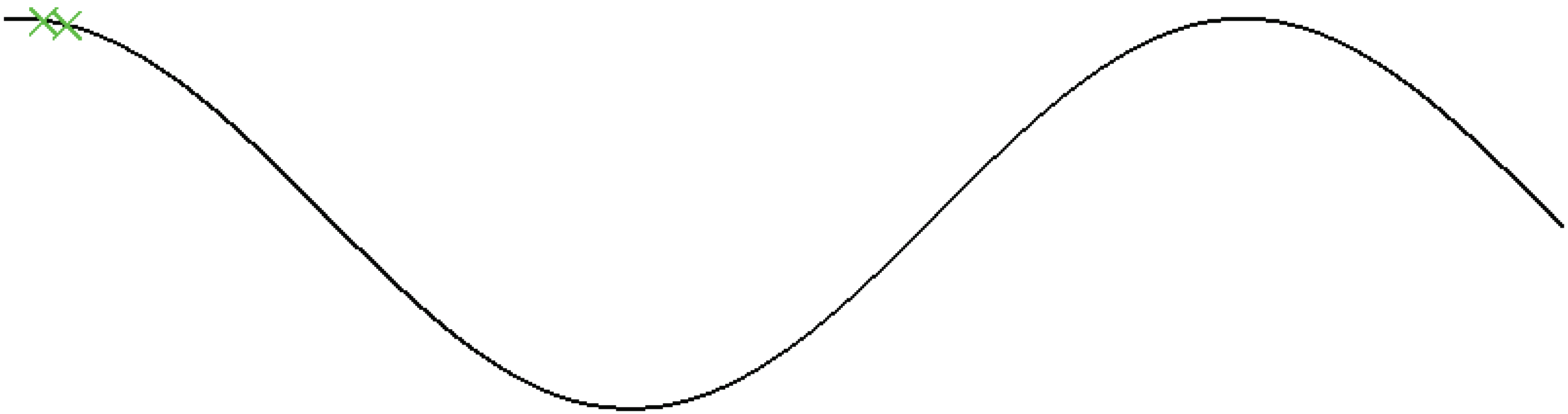
- **Probiere abwechselnd Schritte in Koordinatenrichtungen**
- **Wenn Erfolg (Funktionswert kleiner)**
  - weiter so $\Rightarrow$ Schrittweite erhöhen
- **Wenn Misserfolg (Funktionswert größer)**
  - dieser oder letzter Schritt war zu weit $\Rightarrow$ umdrehen und Schrittweite verkleinern



Quadratische Ausgleichsrechnung

Idee für einen SEHR einfachen Optimierer

- **Probiere abwechselnd Schritte in Koordinatenrichtungen**
- **Wenn Erfolg (Funktionswert kleiner)**
  - weiter so $\Rightarrow$ Schrittweite erhöhen
- **Wenn Misserfolg (Funktionswert größer)**
  - dieser oder letzter Schritt war zu weit $\Rightarrow$ umdrehen und Schrittweite verkleinern



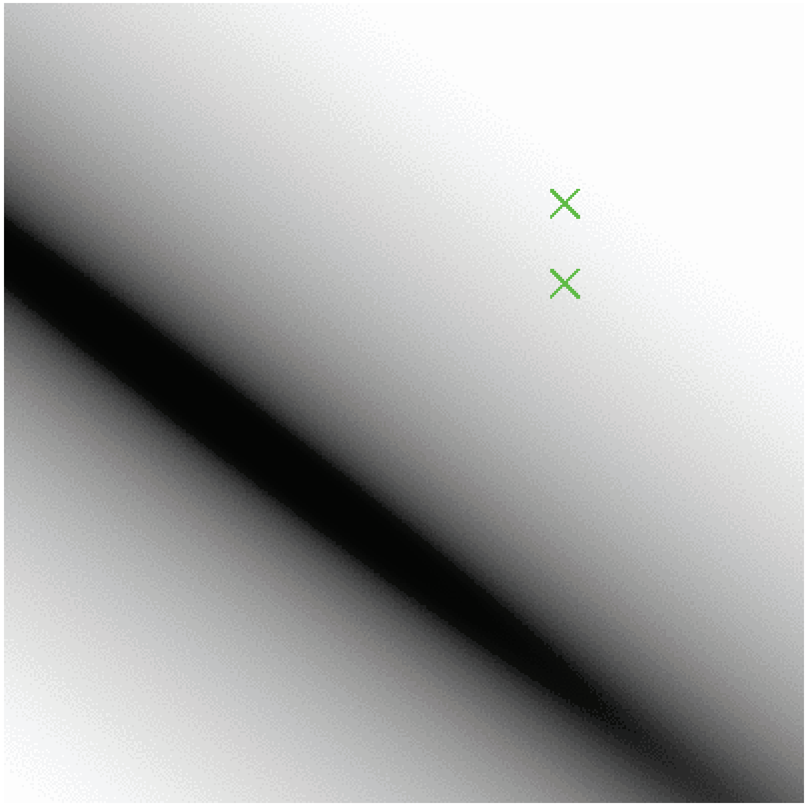
Quadratische Ausgleichsrechnung

```
//! find a minimum of f starting with x performin iterations
double leastSquares ( const vector<Measurement>& meas,
                      vector<double>& x, int iterations) {
   vector<double> step (x.size(), INITIALSTEP);
   double fX = lsError(meas, x);
   for (int ctr=0; ctr<iterations; ctr++) { // oder Abbruchkriterium
      for (int i=0; i<x.size(); i++) {
         x[i] += step[i];
         double fXTry = lsError(meas, x); // Try a step in direction i
         if (fXTry<fX) { // success
            fX = fXTry;
            step[i] *= 2; // try faster
         }
         else { // failure
            x[i] -= step[i];
            step[i] *= -0.5; // try slower and other direction
         }
      }
   }
   return fX;
}
```

Quadratische Ausgleichsrechnung

Problemfall für einfache Optimierer

- **"Schlechte Konditionierung"**
  - hohe Krümmung in einer Richtung
  - niedrige Krümmung quer dazu
- **Algorithmus findet schnell Tal**
- **Läuft nur in kleinen Schritten hinunter**
- **Hohe Krümmung begrenzt Schrittweite**



Zusammenfassung

- **RANSAC**
  - finde passende Zuordnungen aus einer Menge von Zuordnungen
  - generiere Hypothese aus zufällig gezogenen Zuordnungen
  - zähle, wie viele Zuordnungen dazu passen
- **Quadratische Ausgleichsrechnung**
  - Berechnung der wahrscheinlichsten Parameter
    x = argmax_x P(X=x|Z=z) als = argmax_x P(Z=z | X=x).
  - *„Die wahrscheinlichsten Parameter, gegeben, dass ich gesehen habe, was ich gesehen habe sind die, bei denen es am wahrscheinlichsten ist, dass zu sehen, was ich gesehen habe.“*
  - Gauß´sches Messrauschen führt zu quadratischer Minimierung.

$$\hat{x} = \arg\min_x \sum_i \frac{\left(z_i - f_i(x)\right)^2}{\sigma_i^{\ 2}}$$

  - Optimierung mit
    - *langsamen aber einfachem Algorithmus aus der Vorlesung*
    - *Levenberg-Marquardt*
    - *SLoM*