Hauptstudiumspraktikum
Fingerabdruck-Erkennung

Universität Ulm 2003
Abteilung Neuroinformatik

Martin Braun
Markus Kirschmann

Erfahrungsbericht zum **F**inger**P**rint**V**erifikator

1. Einleitung

Ziel des Praktikums war es, die in [1] beschriebene Architektur zu implementieren und gegebenenfalls Optimierungen vorzunehmen. Desweiteren sollten die in [1] angegebenen Ergebnisse verifiziert und die Parametrisierung soweit möglich verfeinert werden.

Die in [1] beschriebe Klassifikator -Schicht des Systemes wurde durch ein Multi-Layer-Perzeptron ersetzt.
Als Lernmethoden wurden zwei Batch-Lernalgorithmen realisiert.

2. Architektur

2.1. Fingerabdruck-Datenbank

Es wurden die im Praktikum aufgenommenen Fingerabdrücke[1] verwendet.
Die Bilder lagen als Graustufen - Bitmaps der Grösse 384 x 320 Pixel vor. Von allen Teilnehmern wurden vom jedem Finger mindestens zwei Scans angefertigt. Insgesamt lagen damit 240 Bilder als Trainingssatz und 140 als Testsatz vor.

2.2 Vorverarbeitung

Auf die Rohdaten wurde zuerst einen parametrisierbaren LoG-Filter[2] angewendet. Dannach wurde aus den gefilterten Bilder ein zetraler Ausschnitt mit Grösse 64 x 64 (DB-Bilder) und ein Ausschnitt mit Grösse 104 x 104 (Eingabe-Bilder) abgespeichert. Die Grössen der Ausschnitte können ebenfalls über Parameter gesteuert werden.

2.3 Mustergenerierung

Dem Netz sollen sowohl Matches als auch Mismatches präsentiert werden. Aus den abgespeicherten Mustern werden diese nun für das Training und den Test des Netzes generiert. Dabei werden die kleineren DB -Bilder auf den Eingabe -Bildern positioniert. Dies geschieht durch das pixelweise Verschieben des DB -Bildes über dem Eingabebild und gleichzeitiger Berechnung des maximalen Korrelationskoeffizienten.
Die resultierenden Musterpaare (64 x 64) können zur Reduzierung des Rechenaufwandes noch weiter verkleinert werden. Als letzter Schritt der Mustergenerierung werden die Daten dann auf Werte zwischen 0 und 1 normalisiert.

---

1 Aufgenommen mit dem Fingerabdruckscanner TST, BiRD 2

2 Matlab -Filter : Laplacian of Gaussian

2.4 Lernen

Das Lernen findet im Batch -Modus statt, d.h. dem Netz werden vor dem Anpassen der Parameter (Gewichte und Schwellen der Filter, Gewichte und Schwelle des Klassifikator-Neurons) sämtliche Musterpaare des Trainingssatzes präsentiert.
Auf die beiden Bilder des jeweiligen Musterpaares werden die adaptiven Filter angewendet. Die beiden Ergebnisse jedes Musterpaares und Filters werden per quadratischer Differenz zusammengeführt und dem Klassifikator -Neuron übergeben.
Als Lernalgorithmen wurden das Error -Backpropagation -Verfahren (BACKPROP) und das Resilient -Backpropagation - Verfahren (RPROP) implementiert.

3. Ergebnisse und Erfahrungen

Grundsätzlich klassifiziert das Netz nach dem Training zwar die zum Training verwendeten Muster relativ gut, werden dem Netz jedoch neue, unbekannte Muster zur Klassifizierung übergeben, so ergibt sich für diese eine wesentlich schlechtere Erkennungsrate.

Durch das Vergrössern der Ausschnitte der Eingabe -Bilder konnte das Ergebnis deutlich verbessert werden, da z.B. zusammengehörige Bilder des Testdatensatzes stärker gegeneinander verschoben waren.

Durch das Ausschliessen von Matches mit einem zu kleinen Korrelationskoeffizienten konnte das Ergebnis des Trainings optimiert werden. Wurde der minimale Korrelationskoeffizient jedoch zu hoch angesetzt, so verschlechterte sich das Ergebnis auf den Testdaten.

Um eine möglichst optimale Klassifikation zu erreichen, wurde zur Ausgabe des Klassifikator-Neurons zusätzlich noch der Korrelationskoeffizient addiert. Durch diese Maßnahme konnte das Ergebnis in einigen Fällen berichtigt werden.

Die in [1] vorgeschlagenen Parameter erwiesen sich als weitestgehend passend. Versuche mit mehreren adaptiven Filtern ergaben zwar eine bessere Erkennungrate auf den Trainingsdaten, jedoch verschlechterte sich dadurch die Erkennungsrate auf den Testdaten enorm.

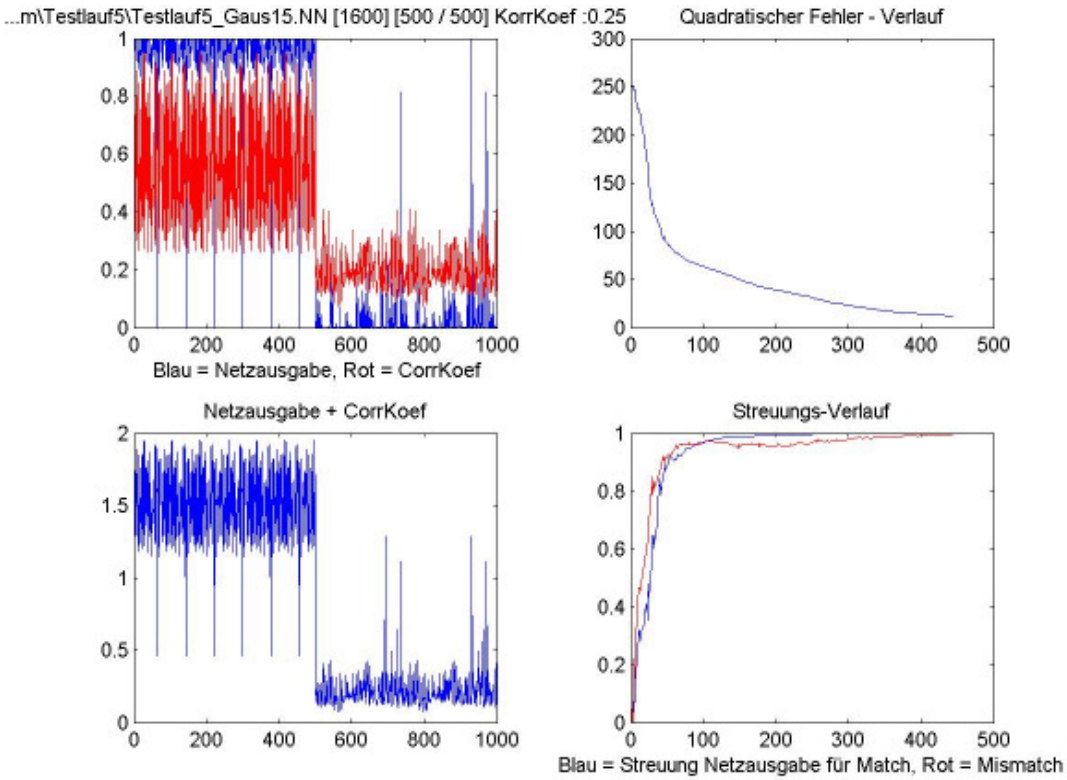
4.Verbesserungsvorschläge

- Geschwindigkeitsoptimierung des Alignments durch „intelligentere“ Such-Methoden, z.B. Diamond-Search oder Korrelation auf verschiedenen Skalen
- Aufwändigere Vorverarbeitung, z.B. durch Signalrestauration
- Kopplung der Ausgaben mehrerer Netze, die unabhängig von einander und unterschiedlich parametrisiert trainiert wurden
- Batchbetrieb anstatt der interaktiven Oberfläche zur Ermittlung der optimalen Parameter

5.Exemplarische Testläufe

5.1 Testlauf a)

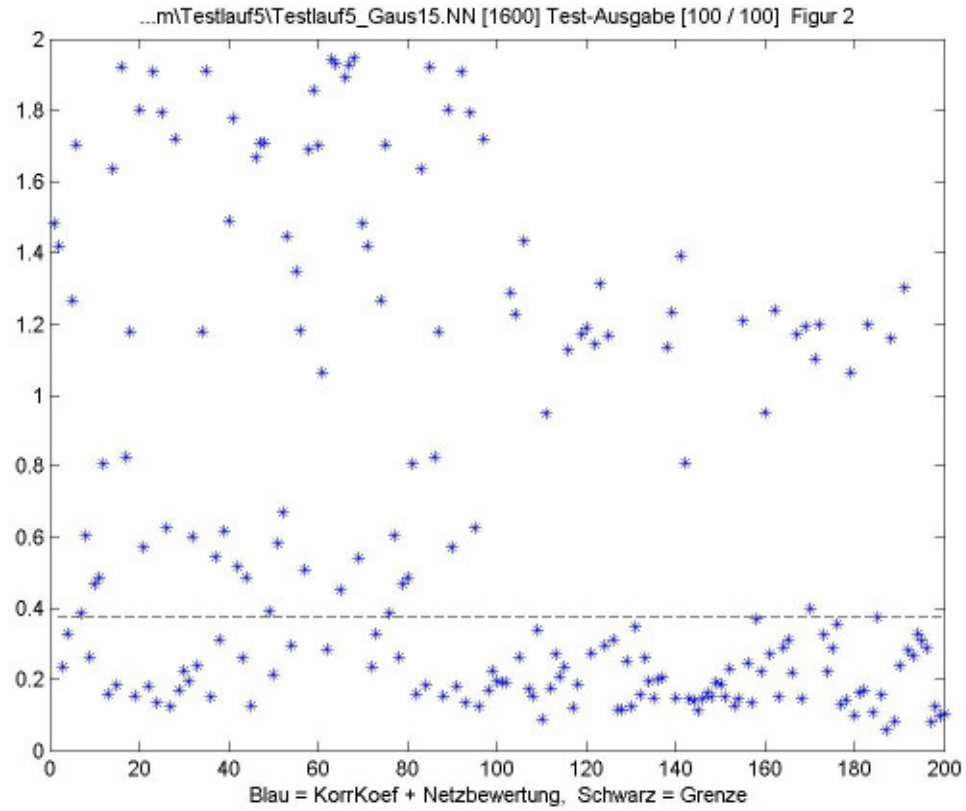
```
Besondere Parameter :
Vorverarbeitung:
      Gaus-Grösse:            15
      Gaus, Sigma:            2

Training:
      Muster:                 500 Match / 500 Mismatch
      Filter:                 2 x (7 x 7)
      Min. Koeff[3]:          0.25
```




```
o.l.  vertikal          : Netzausgabe
      horizontal        : Muster (1..500 Match)
o.r.  vertikal          : Quadratischer Fehler über alle Muster
      horizontal        : Lernepochen
u.l.  vertikal          : Netzausgabe
      horizontal        : Muster (1..500 Match)
u.r.  Vertikal          : Streuung
      horizontal        : Lernepochen
```

3 Minimaler Korrelationskoeffizient des Musterpaares beim Alignment, ab dem das Musterpaar noch als Match beim Training verwendet wird.

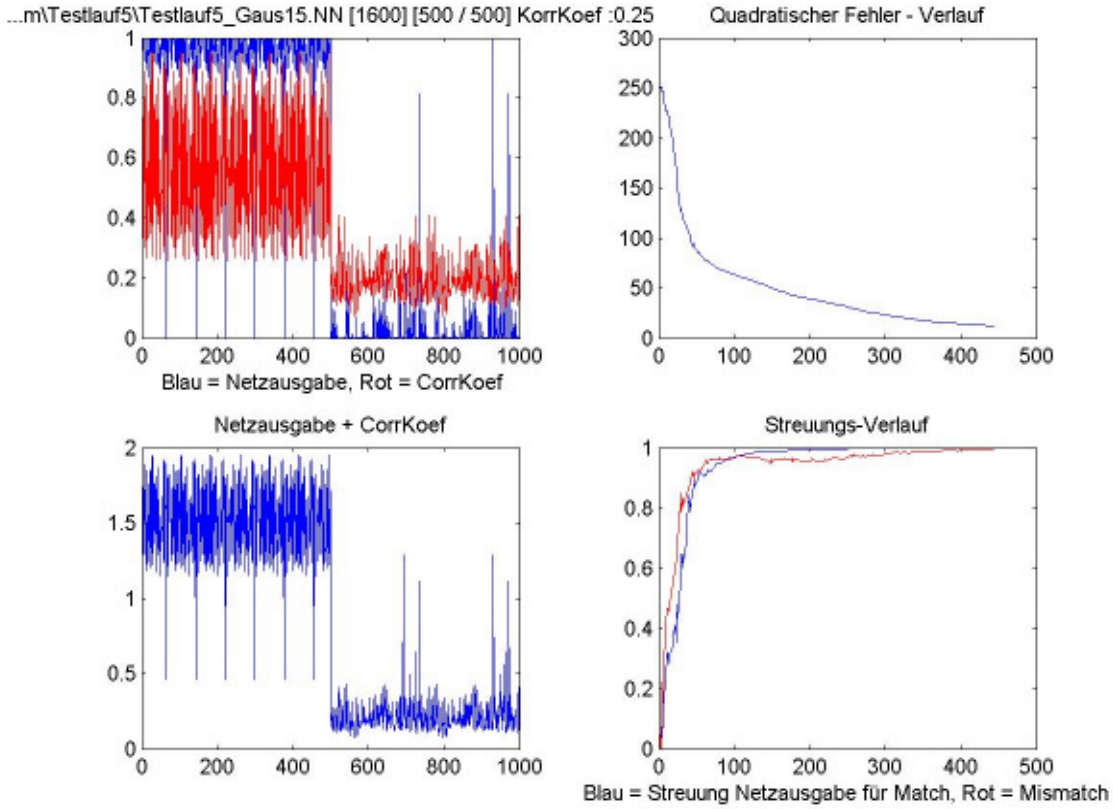
```
Ergebnis auf Testdatensatz
      Fehler Matches:          32
      Fehler Missmatches:      26
      Grenze:                  0.37
```




```
vertikal          : Netzausgabe
horizontal        : Muster - (1..100 Matches)
```

5.2 Testlauf b)

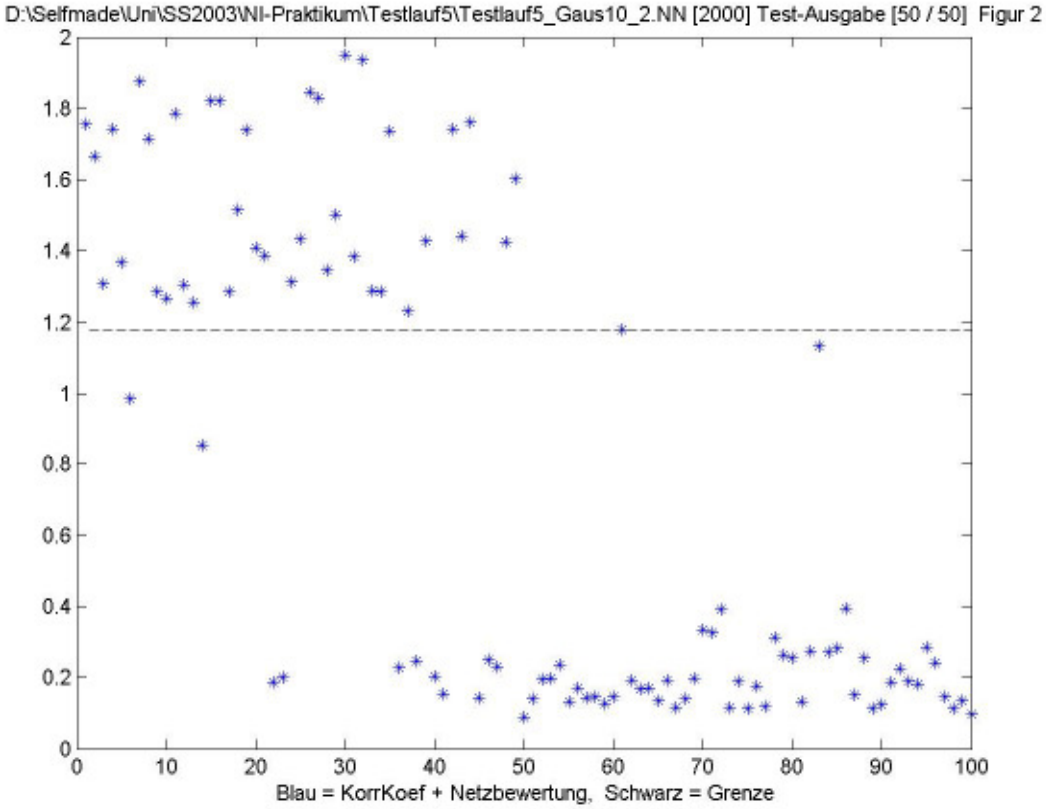
```
Besondere Parameter :
Vorverarbeitung:
      Gaus-Grösse:            10
      Gaus, Sigma:            2


Training:
      Muster:                 500 Match / 500 Mismatch
      Filter:                 2 x (7 x 7)
      Min. Koeff[4]:          0.25
```




```
o.l.  vertikal          : Netzausgabe
      horizontal        : Muster (1..500 Match )
o.r.  vertikal          : Quadratischer Fehler über alle Muster
      horizontal        : Lernepochen
u.l.  vertikal          : Netzausgabe
      horizontal        : Muster (1..500 Match )
u.r.  Vertikal          : Streuung
      horizontal        : Lernepochen
```

4 Minimaler Korrelationskoeffizient des Musterpaares beim Alignment, ab dem das Musterpaar noch als Match beim Training verwendet wird.

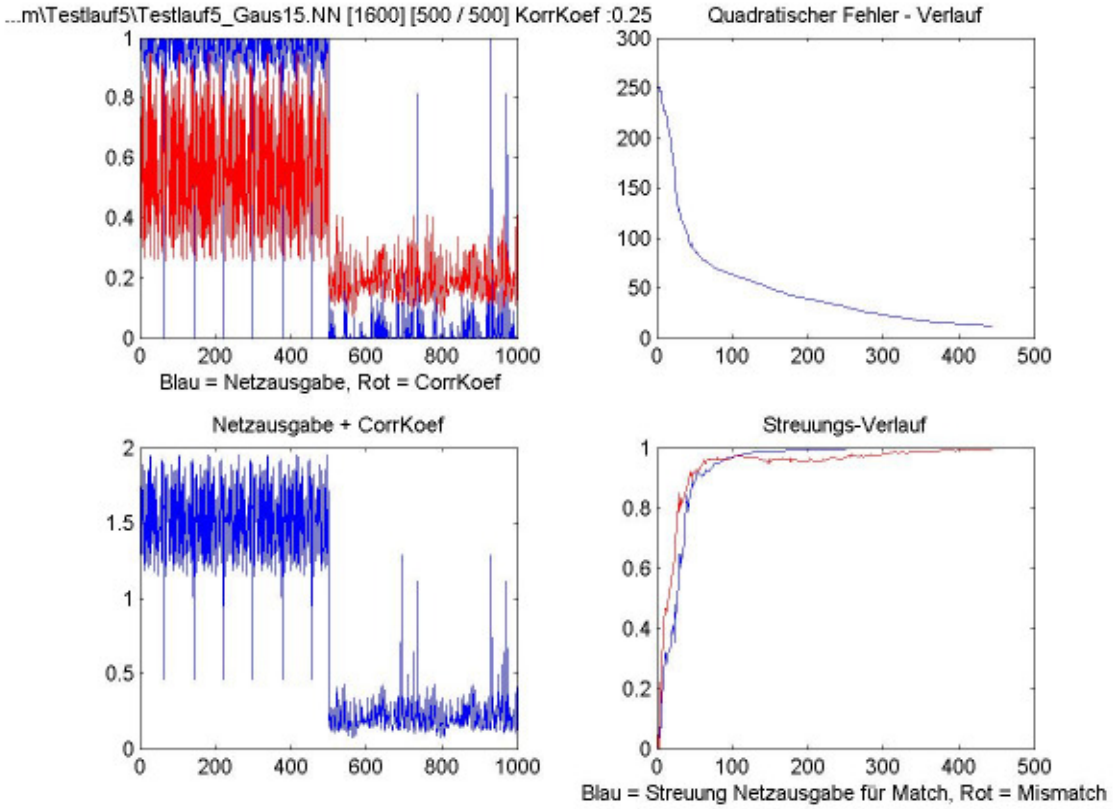
```
Ergebnis auf Testdatensatz
      Fehler Matches:         12
      Fehler Missmatches:     0
      Grenze:                 1.17
```




```
vertikal          : Netzausgabe
horizontal        : Muster - 1..50 Matches
```

5.3 Testlauf c)

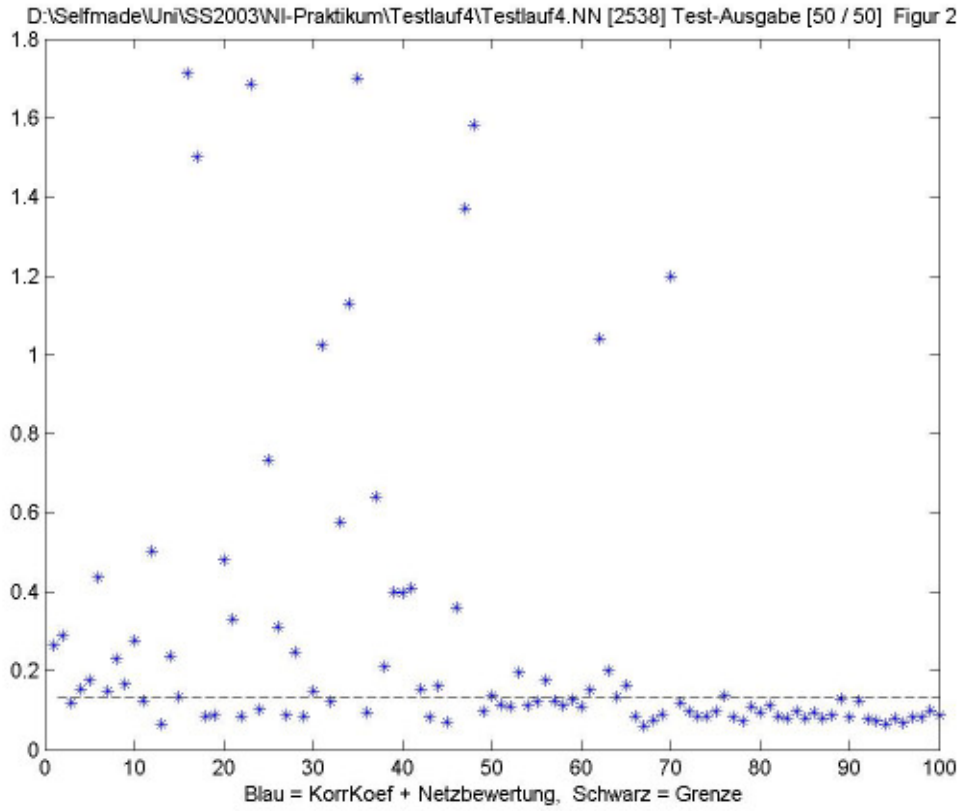
```
Besondere Parameter :
Vorverarbeitung:
      Gaus-Grösse:            10
      Gaus, Sigma:            2

Training:
      Muster:                 500 Match / 500 Mismatch
      Filter:                 2 x (7 x 7)
      Min. Koeff:             0.5
```




```
o.l.  vertikal          : Netzausgabe
      horizontal        : Muster (1..500 Match )
o.r.  vertikal          : Quadratischer Fehler über alle Muster
      horizontal        : Lernepochen
u.l.  vertikal          : Netzausgabe
      horizontal        : Muster (1..500 Match )
u.r.  Vertikal          : Streuung
      horizontal        : Lernepochen
```

Optimales „Auswendig-Lernen“ der Trainingsdaten.
Auf Grund der Einschränkungen durch „min Koeff“ und der begrenzten Anzahl der Trainingsdaten sind nur wenige unterschiedliche Matches verfügbar. Dies verursacht die sich wiederholende Struktur der Netzausgabe im Bereich der Matches.
Daher sollte der „min Koeff“ möglichst klein gewählt werden, um auch „schlechte“ Matches zum Training zu zulassen und dadurch die Erkennung des Netz robuster zu machen.

```
Ergebnis auf Testdatensatz
      Fehler Matches:   14
      Fehler Missmatches:     8
      Grenze:                 0.13
```




```
vertikal          : Netzausgabe
horizontal        : Muster - 1..50 Matches
```

Das Netz hat zwar die Trainingsdaten optimal auswendig gelernt, dadurch wird die Erkennungsrate auf dem Testdatensatz jedoch verschlechtert.

6. Quellen

[1] „Neuronal Networks for Fingerprint Recognitation“.
Pierre Baldi, Yves Chauvin