

# RECONNAISSANCE DE GESTES

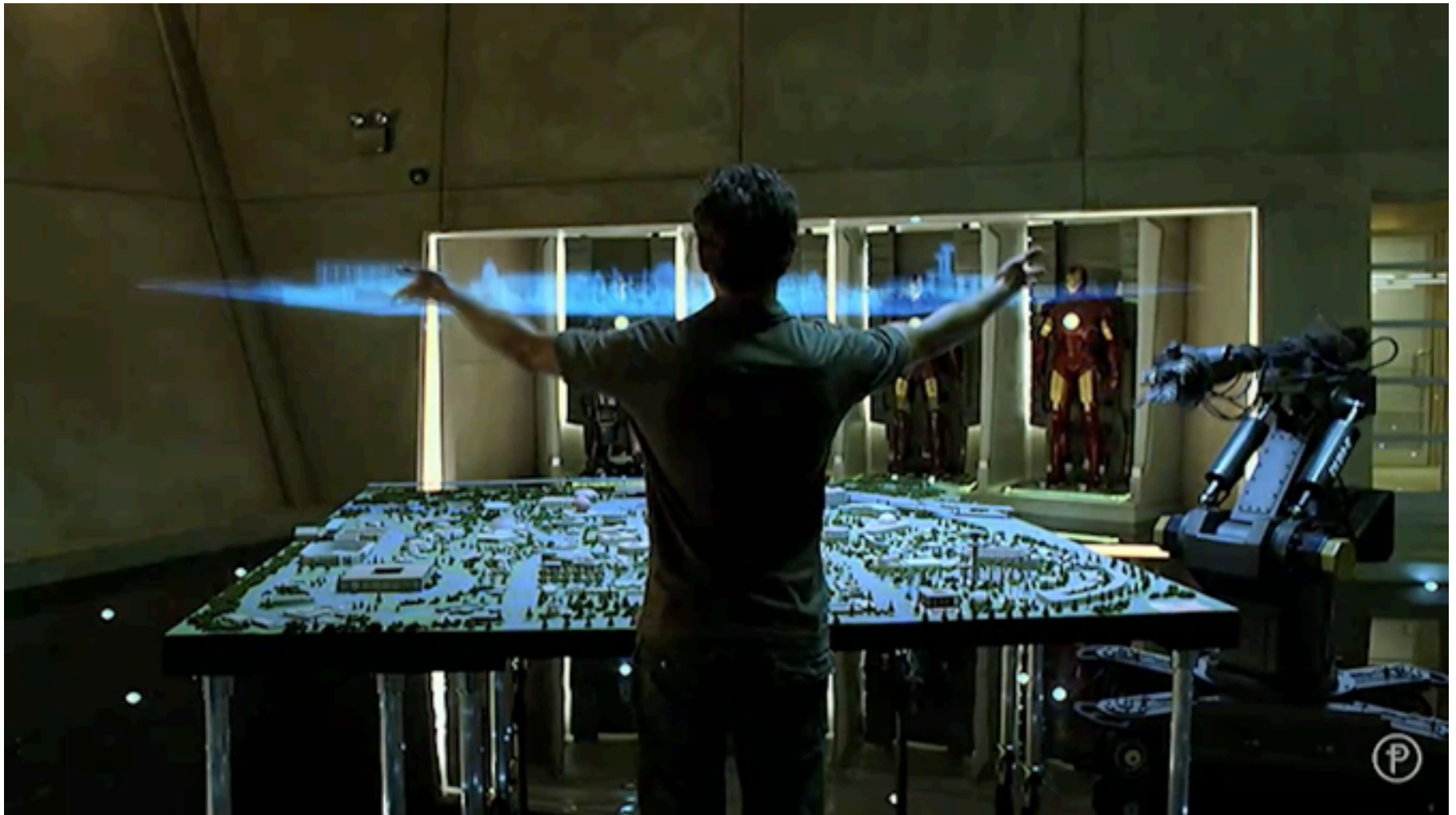
# Minority report

2



# Iron man

3



# Plan

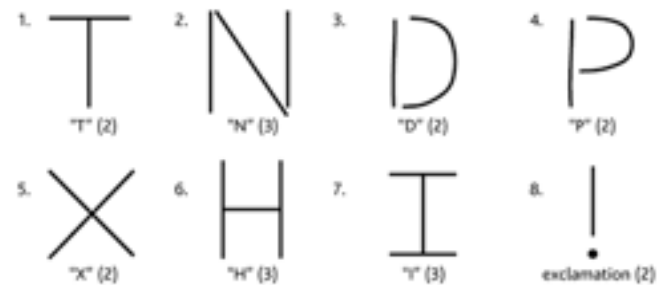
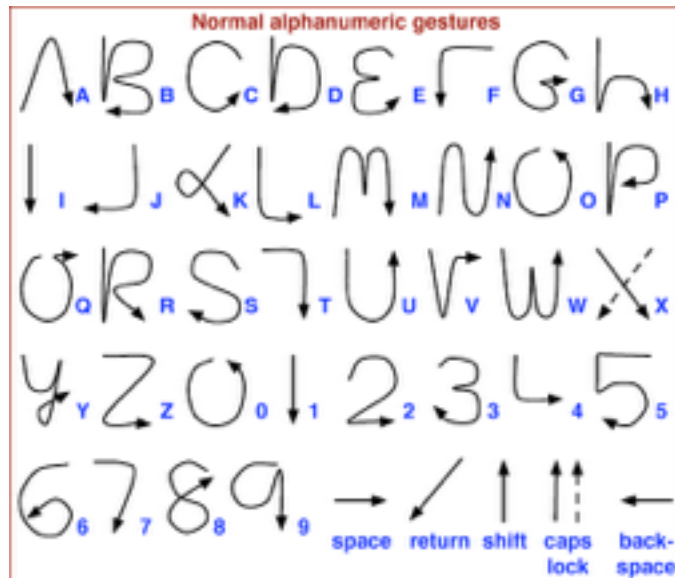
4

- Objectifs
- Technique libstroke
- Technique \$1
- Rubine
- DTW

# Objectifs

5

- Reconnaître des gestes réalisés par l'utilisateur pour les associer à des commandes
- On se limite aux gestes un seul trait (unistroke)



Multistroke

Unistroke - Graffiti (Palm OS)

# Technique de reconnaissance de gestes élémentaires: libstroke (1997)

6

- Librairie disponible dans FVWM <http://etla.net/libstroke/>
- Les gestes sont décrits par des séquences de chiffres

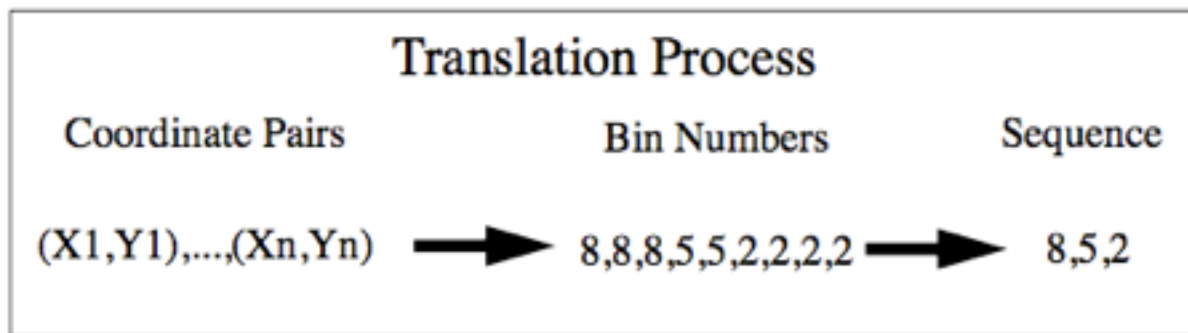
```
# Strokes
#      num      button context mod. action
Stroke 14789    2      A      N      Exec exec xlock -mode blank&
Stroke 258      2      A      N      Exec exec xterm&
Stroke 563214789 2      A      N      Exec exec exmh&
Stroke 7415963  2      A      N      Exec exec netscape&
Stroke 741236987 2      A      N      Destroy
Stroke 1478963  2      A      N      Popup "Apps"
Stroke 74123    2      A      N      Module "winlist" FvwmWinList
Stroke 74159    2      A      N      Move
Stroke 852      2      A      N      Menu "Apps" Nop
```

1	2	3
4	5	6
7	8	9

# Libstroke: algorithme

7

- 1) détermination de la boîte englobante (bounding box) à partir de  $\min_x, \max_x, \min_y, \max_y$
- 2) division de la boîte en 9 cellules
- 3) étiquetage des points
- 4) factorisation
- 5) comparaison aux motifs enregistrés



- Extension Firefox: Mouse Gestures Redox

# Techniques de reconnaissance

8

- Classifieurs statistiques (Rubine, DTW)
- Modèles de Markov cachés
- Réseaux de neurones
- Méthodes ad-hoc (libstroke, \$1)



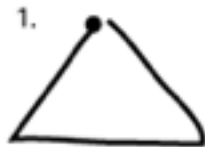
# Techniques de reconnaissance

9

- \$1 recognizer
- Méthode simple à implémenter et donnant des taux de reconnaissance comparables à Rubine et DTW
- 1 seul exemple suffit
- Technique publiée dans l'article de recherche "Gestures without Libraries, Toolkits or Training: A \$1 Recognizer for User Interface Prototypes" à la conférence User Interface Software and Technology (UIST) en 2007 (écrit par Wobbrock, Wilson et Li)

# \$1 recognizer

10



triangle



"x"



rectangle



circle



check



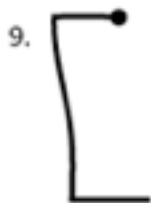
caret



question



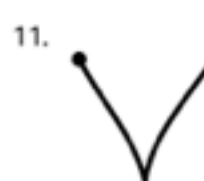
arrow



left square bracket



right square bracket



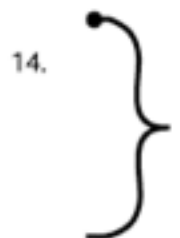
"v"



delete



left curly brace



right curly brace



star



pigtail

# Algorithme

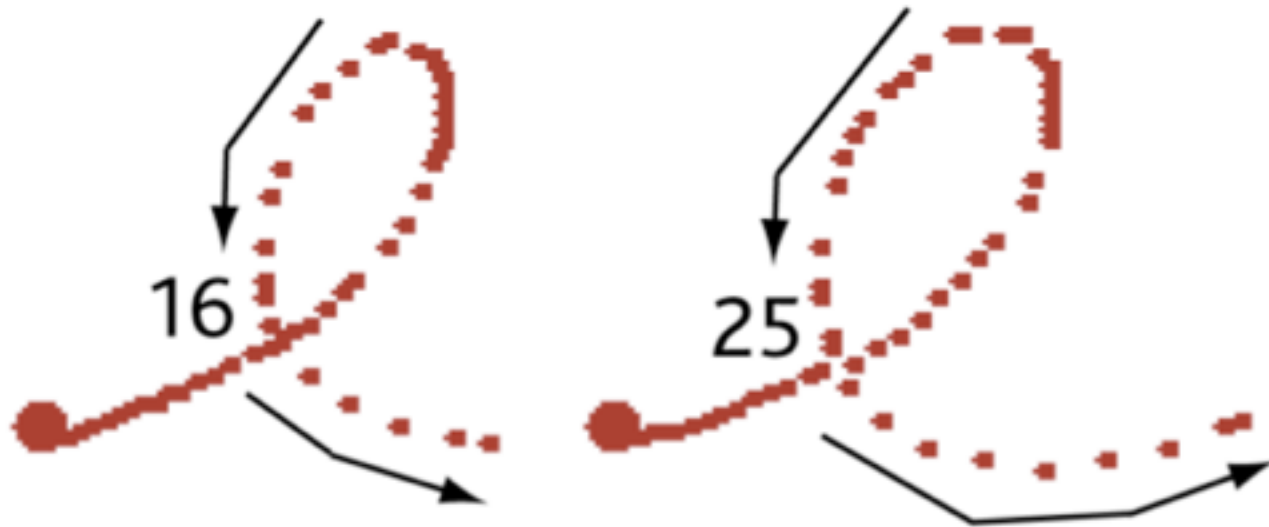
11

- 1) L'utilisateur réalise un geste
  - ▣ Le geste est représenté par une liste ordonnée de points
- 2) Ce geste est comparé à un ensemble de gestes de référence (templates) en utilisant une mesure de distance euclidienne
- 3) Le geste reconnu est celui pour lequel cette distance est minimale

# Problèmes posés

12

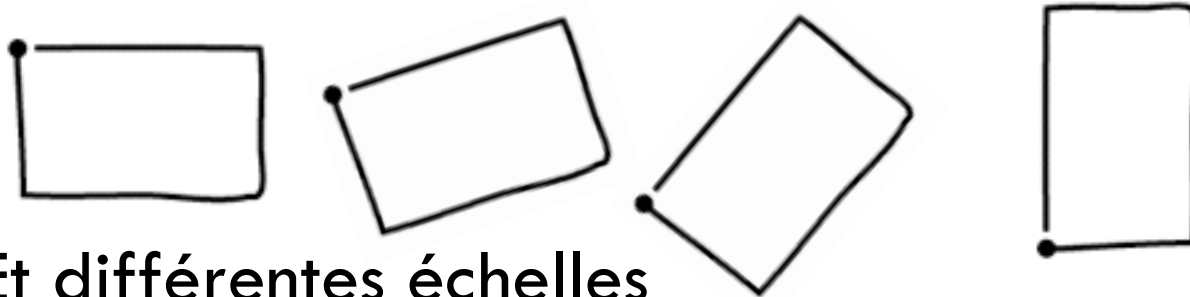
- Le nombre de points pour définir un geste va dépendre de la vitesse d'exécution, de la fréquence d'échantillonnage du périphérique...



# Problèmes posés

13

- Un geste peut être réalisé à différentes positions sur l'interface,
- Suivant différentes orientations



- Et différentes échelles



# 4 étapes

14

- 1) Ré-échantillonnage du geste pour être invariant à la fréquence d'acquisition et la vitesse d'exécution du geste
- 2) Ré-orientation du geste pour être invariant à l'orientation suivant laquelle il est exécuté
- 3) Mise à l'échelle et translation pour être invariant à l'échelle de réalisation du geste et la position à laquelle il est exécuté
- 4) Reconnaissance du geste

# 1<sup>ère</sup> étape: Ré-échantillonnage

15

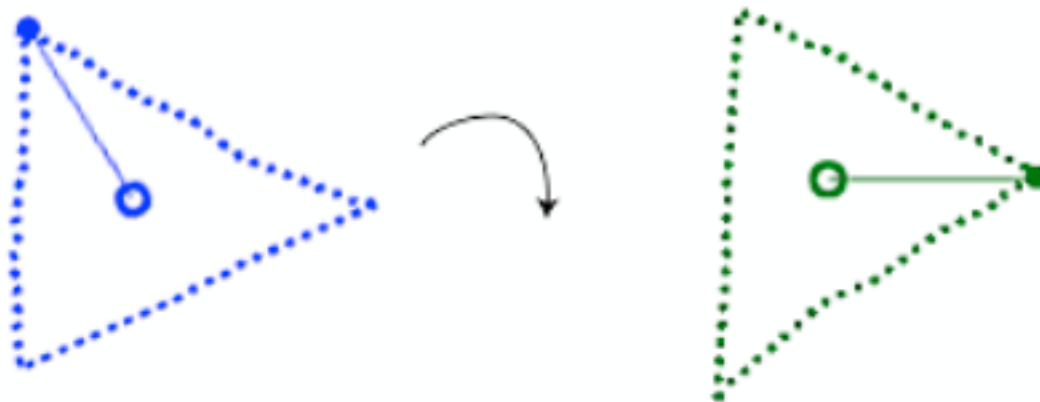
- Le geste est défini par  $M$  points ordonnés
- On veut  $N$  points ordonnés équidistants les uns des autres
- $N = 64$



## 2<sup>e</sup> étape: Rotation « indicative »

16

- 1) Calcul du centre du geste (centroïde)
- 2) Calcul de l'angle entre le centroïde, le premier point et l'horizontale
- 3) Rotation des points en utilisant cet angle

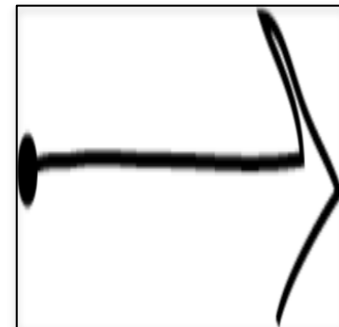
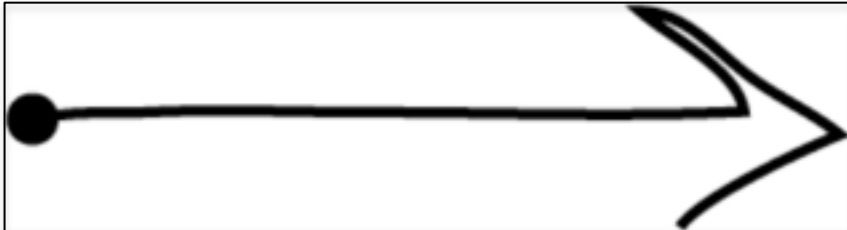




# 3<sup>e</sup> étape: mise à l'échelle et translation

17

- Mise à l'échelle non uniforme: on ramène le geste à un carré de référence
- 1) détermination de la boîte englobante du geste (bounding box)
  - ▣ Détermination de  $\min_x$ ,  $\max_x$ ,  $\min_y$ ,  $\max_y$
- 2) Mise à l'échelle
- 3) Translation à l'origine



## 4<sup>e</sup> étape: reconnaissance

18

- Un geste candidat  $C$  est comparé à chacun des gestes enregistrés (templates)  $T_i$  pour calculer la distance moyenne  $d_i$  entre les points correspondants

$$d_i = \frac{\sum_{k=1}^N \sqrt{(C[k]_x - T_i[k]_x)^2 + (C[k]_y - T_i[k]_y)^2}}{N}$$

- Le template avec la distance moyenne la plus faible est le résultat de la reconnaissance
- La distance est transformée en score entre 0 et 1

$$score = 1 - \frac{d_i^*}{\frac{1}{2} \sqrt{size^2 + size^2}}$$

# Limitations

19

- Pas possible de distinguer un carré d'un rectangle, une ellipse d'un cercle, une flèche vers le bas d'une flèche vers le haut
- Pas possible de reconnaître des gestes « 1D »

# Classifieur statistique

20

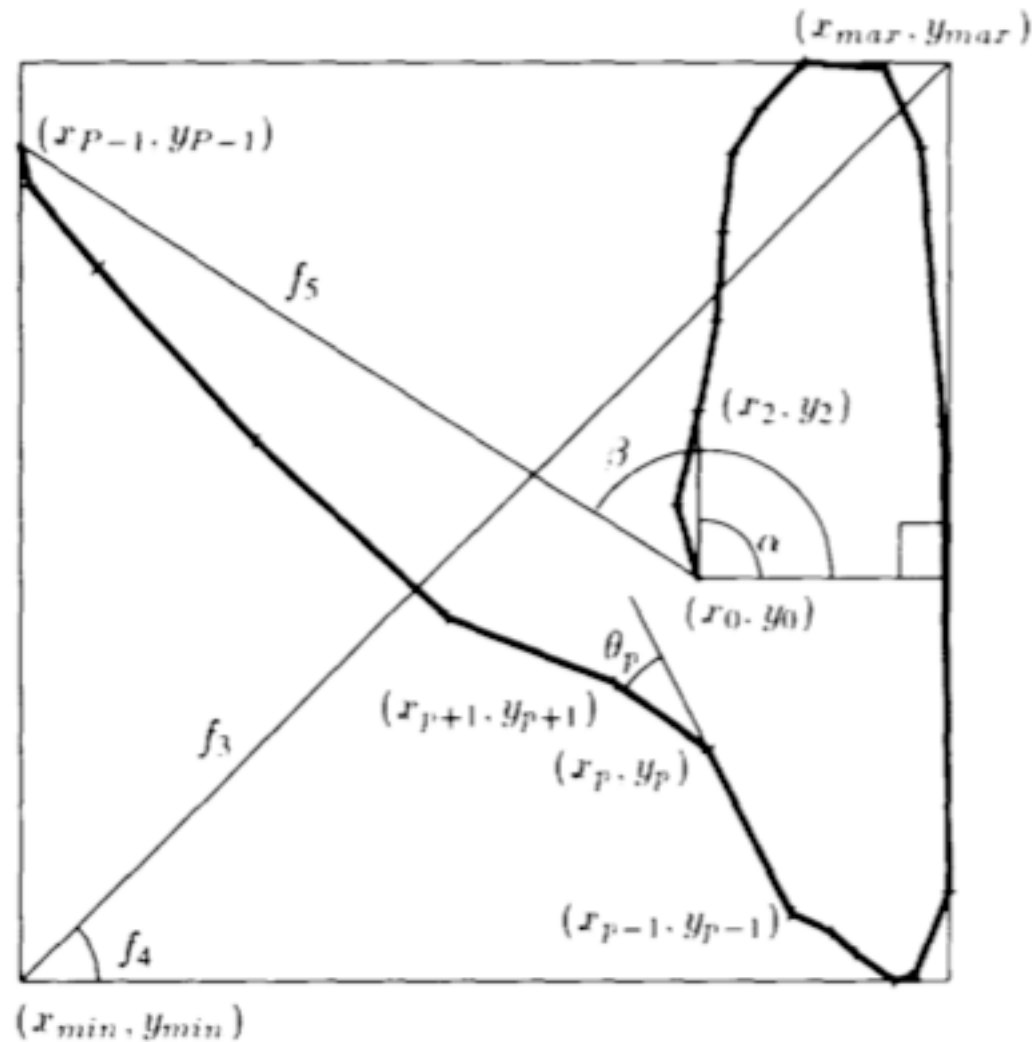
- Rubine
- Reconnaissance de gestes ne comportant qu'un seul tracé (unistroke)
- Traitement statistique de caractéristiques

Dean Rubine. 1991. Specifying gestures by example. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '91). ACM, New York, NY, USA, 329-337. DOI=10.1145/122718.122753 <http://doi.acm.org/10.1145/122718.122753>

- Liste de points en entrée
  - ▣ Traitement des points pour éliminer les points trop proches: tout point dont la distance est inférieure à 3 pixels du point précédent est supprimé
  - ▣ Calcul d'un vecteur de caractéristiques statistiques
  - ▣ Comparaison aux différents gestes de référence. Le geste avec le score le plus grand est renvoyé.
  - ▣ Détection des gestes ambigus

# Caractéristiques

22



# Initialisation

23

- Calcul des statistiques moyennes pour chaque caractéristique de chaque geste
- Calcul de la matrice de covariance moyenne de tous les gestes
- Objectif: trouver les coefficients pondérateurs des caractéristiques statistiques qui permettent de maximiser le score des gestes de chaque classe

# Reconnaissance

24

- Calcul des caractéristiques du geste candidat
- Calcul d'une probabilité de correspondance pour chaque geste de référence
- Le geste avec la probabilité la plus importante est choisi



# Dynamic Time Warping

25

- DTW
- Déformation temporelle dynamique
- Déterminer pour chaque élément d'une séquence, le meilleur élément correspondant dans l'autre séquence relativement à un certain voisinage et à une certaine métrique
- Complexité polynomiale

# Applications

26

- Vidéo, audio, graphique...
- Toutes données qui peuvent être transformées en représentation linéaire en fonction du temps (séries temporelles)
- Echantillons ordonnés par une étiquette de temps
- Reconnaissance vocale
- Reconnaissance de gestes off-line et on-line
- alignement de protéines...

# Principe de base

27

- Séquence de référence  $R = [r_1, r_2, \dots, r_n]$
- Séquence de test  $T = [t_1, t_2, \dots, t_m]$
- Si  $m=n$  alors on peut calculer la distance entre les deux signaux de la façon suivante (pas forcément idéal):

$$D = \sum_{i=1}^n distance(r_i, t_i)$$

- Possibilité de calculer la distance euclidienne ou d'utiliser une autre métrique (fonction qui donne un réel)
- Plus possible d'utiliser cette méthode dès que  $n \neq m$
- Attention: les échantillons des séquences doivent être équidistants en temps

# Principe de base

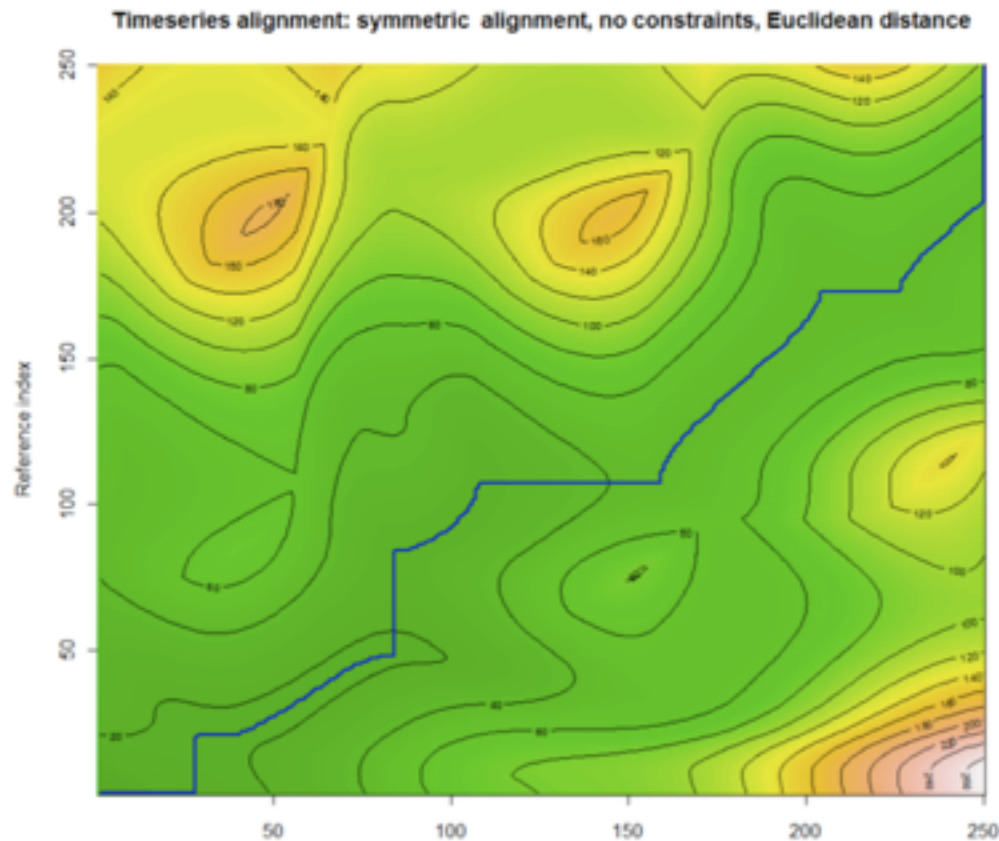
28

- DTW réalise d'abord un alignement non linéaire en recherchant parmi tous les alignements possibles, celui qui minimise une fonction de coût cumulé
- «Time Warping»: Dilation ou compression des séquences pour obtenir le meilleur alignement possible

# Principe de base

29

- Détermination du chemin  $W = [w_1, w_2, \dots, w_k]$  de longueur minimale  $\sum_{i=1}^k distance(w_i)$



# Principe de base

30

## □ Conditions aux frontières

- $w_1 = (r_1, t_1)$

- $w_k = (r_n, t_m)$

## □ Contraintes locales

- Monotonie pour respecter le séquençement des points

- Eviter les sauts dans le temps

- Pour tout couple  $(r_i, t_i)$ , le choix des prédécesseurs est limité à

- $(r_{i-1}, t_i), (r_i, t_{i-1}), (r_{i-1}, t_{i-1})$

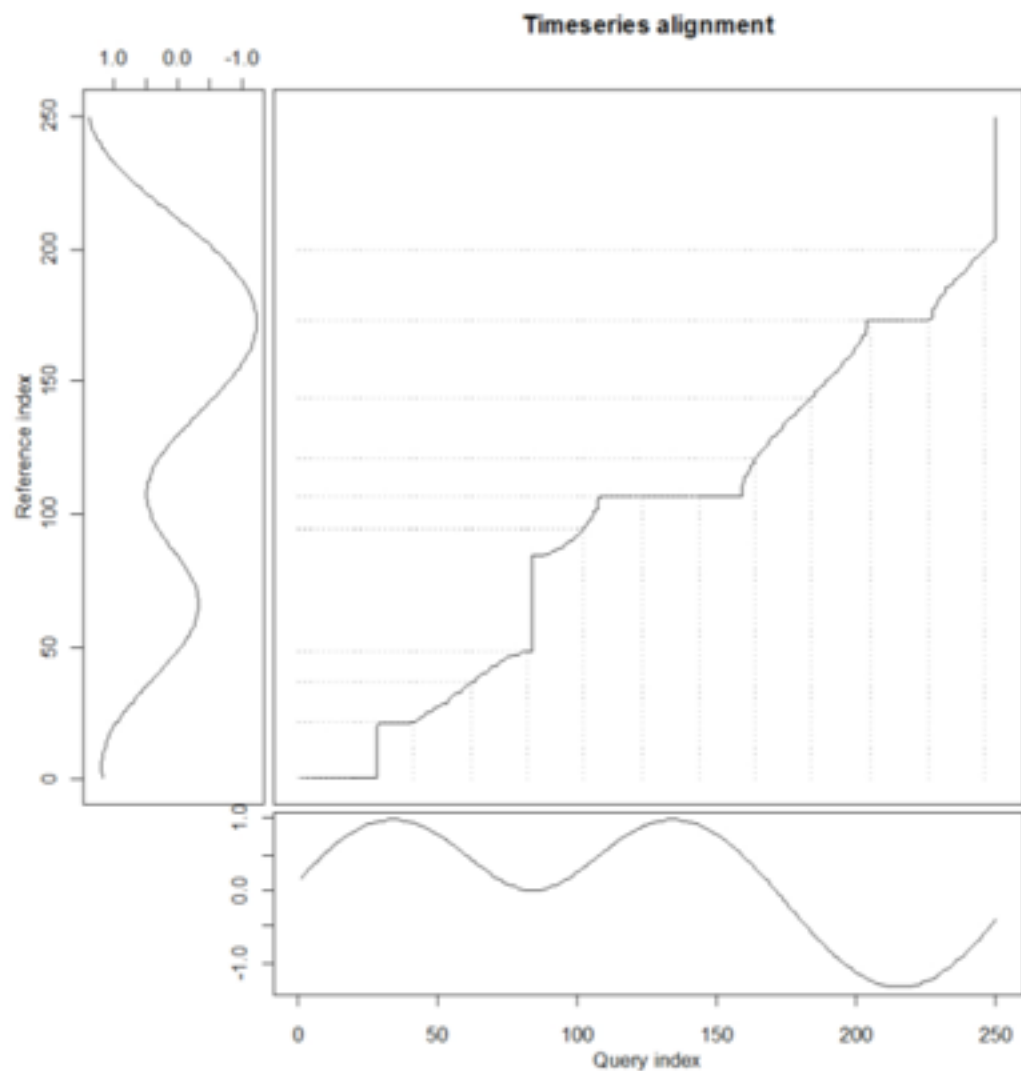
## □ Exhaustivité

- Chaque élément de  $R$  doit être mis en relation avec au moins un élément de  $T$  et vice-versa

- $\max(m, n) \leq k \leq m + n - 1$

# Principe de base

31



# Principe de base

32

- Programmation dynamique
- Fonction d'optimisation:
  - ▣ Soit  $D(i,j)$  la longueur du chemin entre  $(r_1, t_1)$  et  $(r_i, t_i)$
- Récursion:
  - ▣  $D(i,j) = \text{dist}(r_i, t_j) + \min(D(i-1,j), D(i-1,j-1), D(i, j-1))$
  - ▣ condition initiale:  $D(1,1) = \text{dist}(r_0, t_0)$
- Distance minimale entre les deux séquences
  - ▣  $D(n,m)$



# Mise en application

33

- Construction d'une matrice  $D$  de dimensions  $n \times m$  dans laquelle chaque élément  $(i,j)$  contient  $D(i,j)$
- Remplissage de  $D(1,1)$  avec la condition initiale
- Utilisation de la formule récursive pour remplir la matrice ligne par ligne ou colonne par colonne
- Cas particuliers première ligne et première colonne
- Distance minimale donnée par l'élément  $D(n,m)$
- La distance minimale peut être normalisée par la longueur du chemin

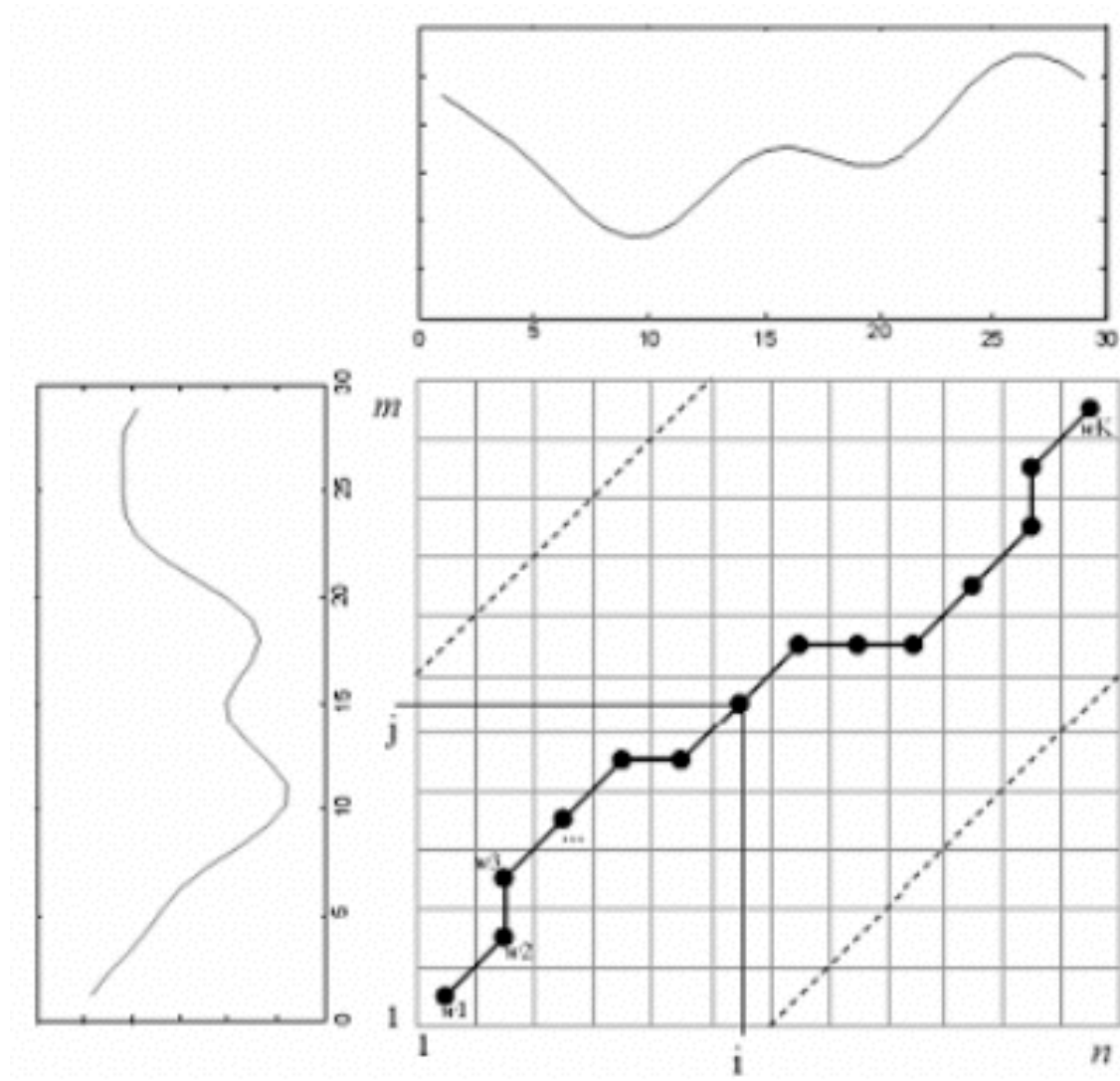
# Algorithme

34

```
n ← |X|
m ← |Y|
dtw[] ← new [n × m]
dtw(0,0) ← 0
for i = 1; i ≤ n; i ++ do
    dtw(i,0) ← dtw(i-1,0) + c(i,0)
end for
for j = 1; j ≤ m; j ++ do
    dtw(0,j) ← dtw(0,j-1) + c(0,j)
end for
for i = 1; i ≤ n; i ++ do
    for j = 1; j ≤ m; j ++ do
        dtw(i,j) ← c(i,j) + min {dtw(i-1,j); dtw(i,j-1); dtw(i-1,j-1)}
    end for
end for
return dtw
```

# Chemin de déformation

35



# Chemin de déformation

36

- A chaque calcul de  $D(i,j)$ , sauvegarde du prédécesseur qui minimise la distance
- Parcours des prédécesseurs en partant de  $D(n,m)$

# Complexité

37

- Complexité:  $O(m*n)$
- Optimisation en limitant la région de recherche

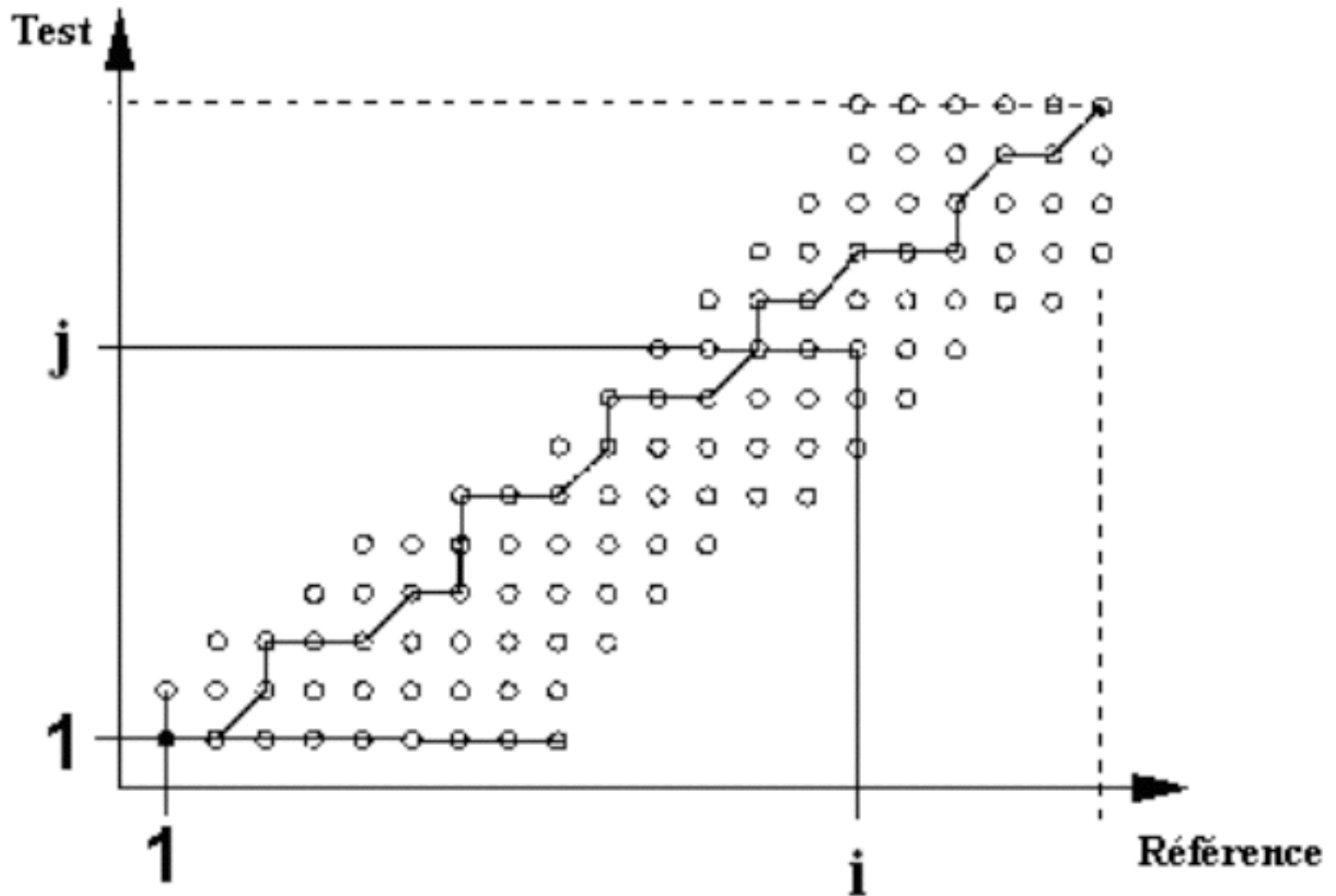
# Optimisation

38

- La zone supérieure gauche et la zone inférieure droite ne sont pas calculées
- Les distances locales associées sont mises à une valeur très élevée afin que le chemin n'y passe pas

# Optimisation

39



# Optimisation

40

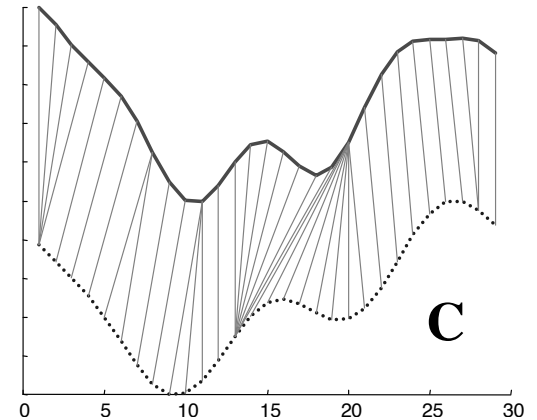
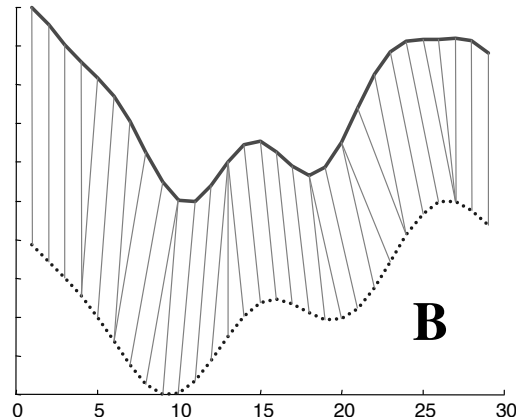
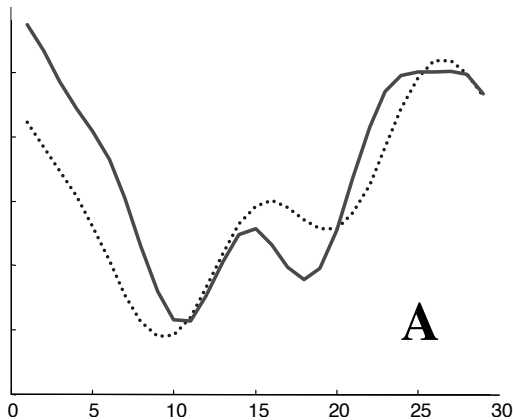
- Mise à l'échelle
  - ▣ Réduire les tailles de R et T



# Singularités

41

- Une singularité apparaît quand un point d'une séquence est associé à de nombreux points de l'autre séquence sans raison valable



# Singularités: technique de fenêtrage

42

- Ajout de contraintes dans les alignements possibles
- Technique de fenêtrage
  - ▣ Donne une borne supérieure à la singularité

```
for  $i = 1; i \leq n; i++$  do      for  $j = \max(1, i-w); j \leq \min(m, i+w); j++$  do
  for  $j = 1; j \leq m; j++$  do
     $dtw(i, j) \leftarrow c(i, j) + \min \{ dtw(i-1, j); dtw(i, j-1); dtw(i-1, j-1) \}$ 
  end for
end for
```

# Singularités: slope weighting

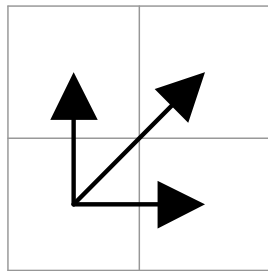
43

- Pondérer les déplacements suivant les directions:
  - ▣  $D(i,j) = \text{dist}(r_i, t_j) + \min(X * D(i-1, j), D(i-1, j-1), X * D(i, j-1))$
  - ▣  $X > 1$
- Force l'alignement suivant la diagonale

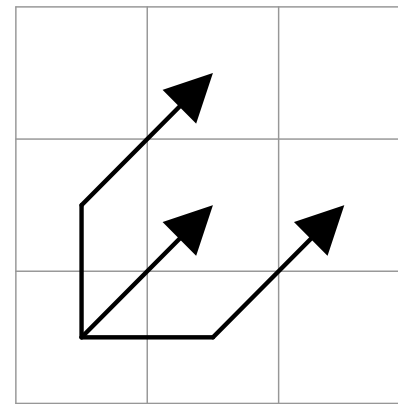
# Singularités: Step patterns

44

- $D(i,j) = \text{dist}(r_i, t_j) + \min(D(i-1, j-2), D(i-1, j-1), D(i-2, j-1))$
- Force le déplacement suivant la diagonale



**A**



**B**

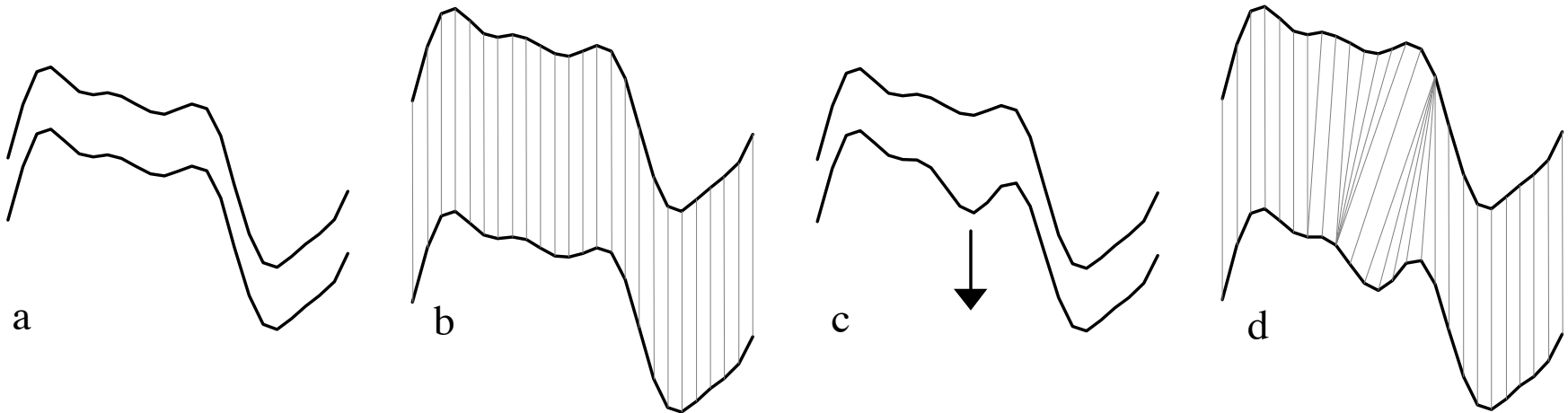
# Singularités

45

- Problème: ces contraintes peuvent empêcher d'obtenir l'alignement correct
- Comment définir les différents paramètres des techniques de levée des singularités (taille de fenêtre, coefficients de pondération, motifs)?

# DDTW: Derivative Dynamic Time Warping

46



```
@INPROCEEDINGS{Keogh01derivativedynamic,  
  author = {Eamonn J. Keogh and Michael J. Pazzani},  
  title = {Derivative Dynamic Time Warping},  
  booktitle = {In First SIAM International Conference on Data Mining (SDM'2001)},  
  year = {2001}  
}
```

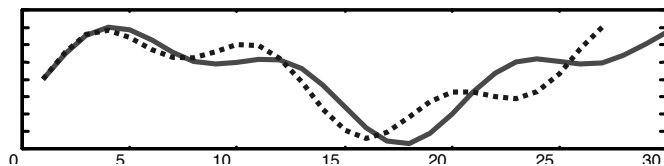
# DDTW: Derivative Dynamic Time Warping

47

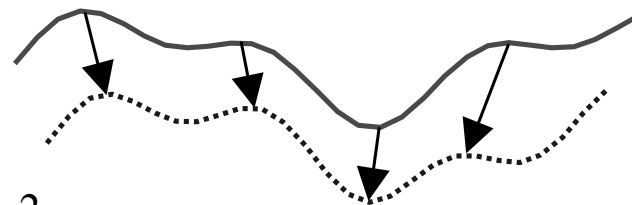
- Utilisation de la dérivée des données des séquences

$$D_x[q] = \frac{(q_i - q_{i-1}) + ((q_{i+1} - q_{i-1})/2)}{2}$$

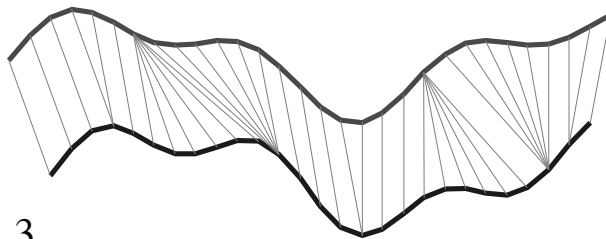
- Calcul du carré de la différence des dérivées



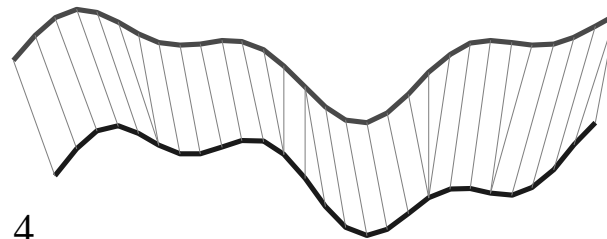
1



2



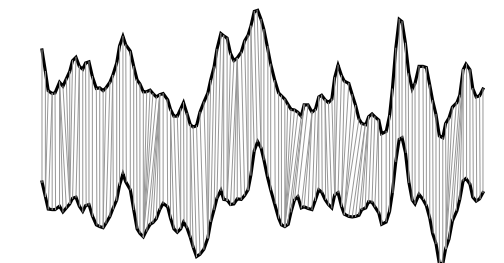
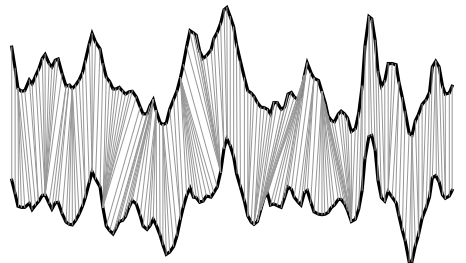
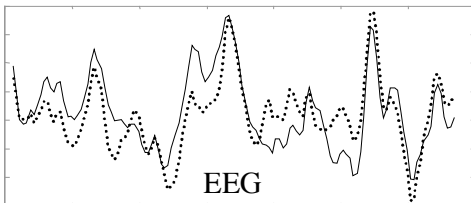
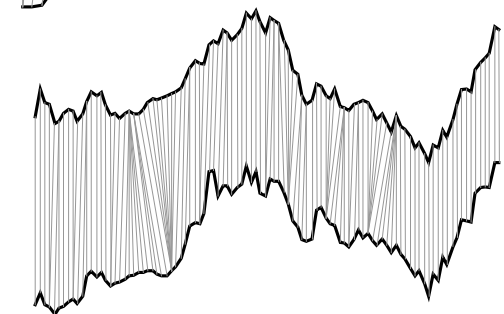
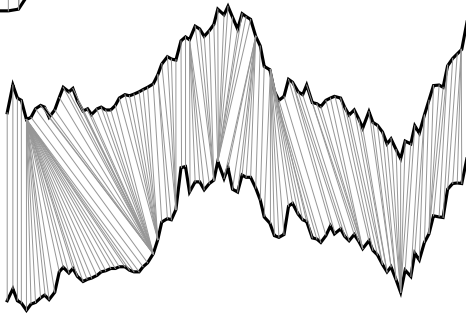
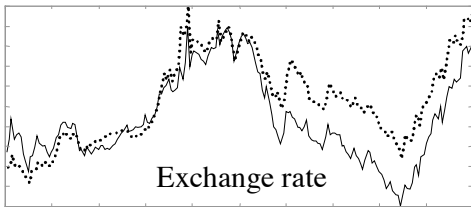
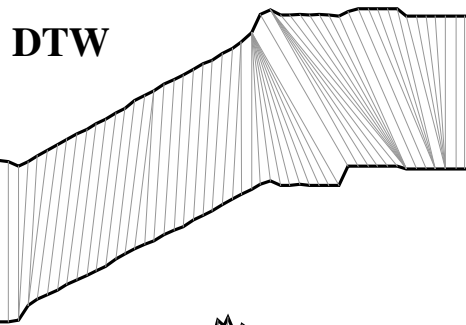
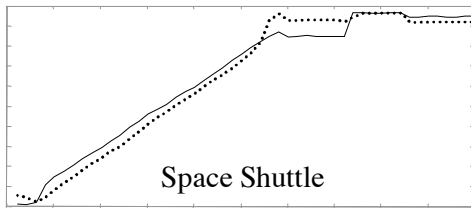
3



4

# DDTW: Derivative Dynamic Time Warping

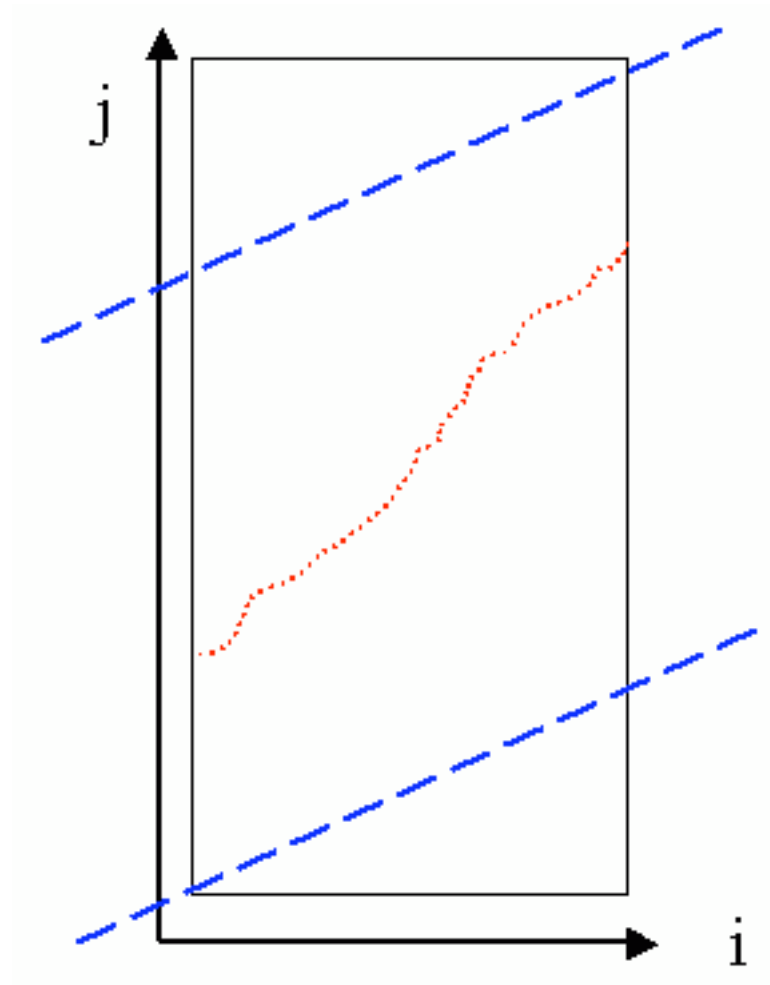
48





# Recherche de motifs

49



# Reconnaissance de gestes

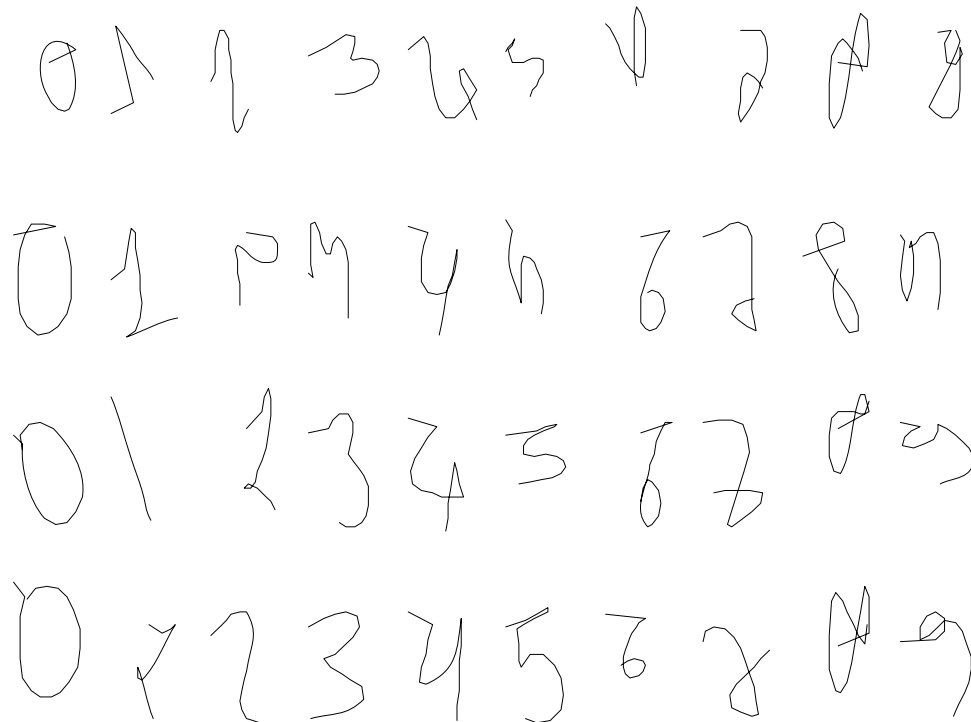
50

- Pré-traitement des gestes de référence et du geste réalisé par l'utilisateur
  - ▣ mise à l'échelle, changement origine, rotation...
- Alignement avec chacun des gestes
- Le geste de référence pour lequel la distance entre les deux séquences est la plus faible est considéré comme reconnu

# Affine invariant DTW

51

- Problème: identifier dans une séquence un objet qui subit une transformation affine
- Choix de la fonction de coût?



# Affine invariant DTW

52

## Optimization Algorithm of AI-DTW:

Initialize

The warping path  $w^{(1)} = DTW\_PATH(T, R)$ .

Iteration number  $k=1$

While not convergence

$k = k+1$ ;

Update the transformation matrix by:

$$A^{(k)} = \arg \min_A \left\{ \sum_{i=1}^n \left\| t_i - r_{w^{(k-1)}(i)} A \right\|^2 \right\} .$$

Update the warping path by:

$$w^{(k)} = DTW\_PATH(T, RA^{(k)}).$$

End While

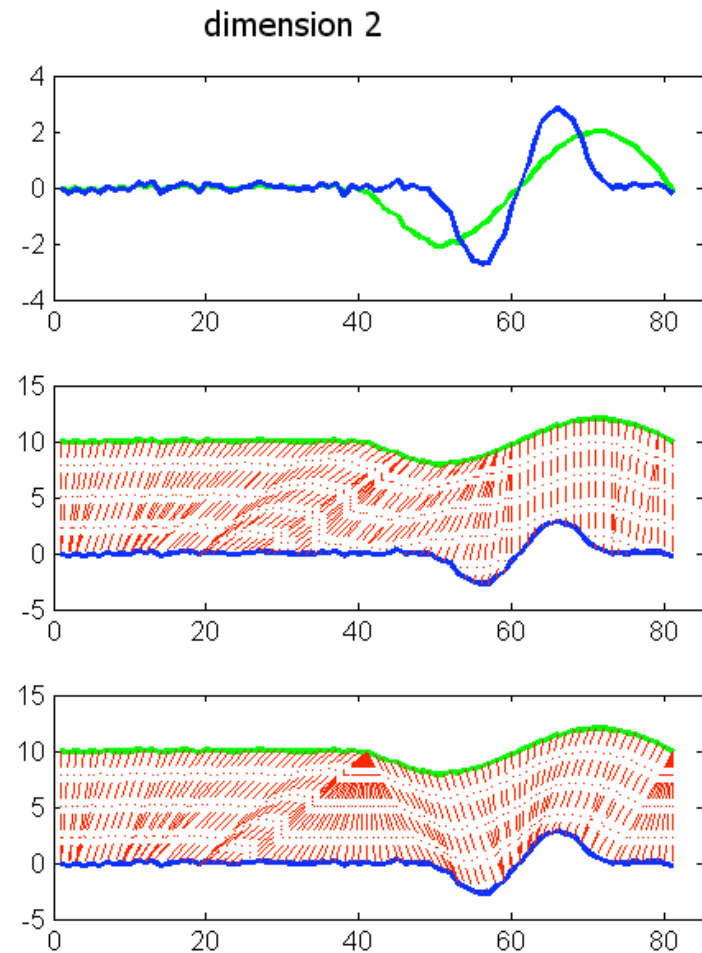
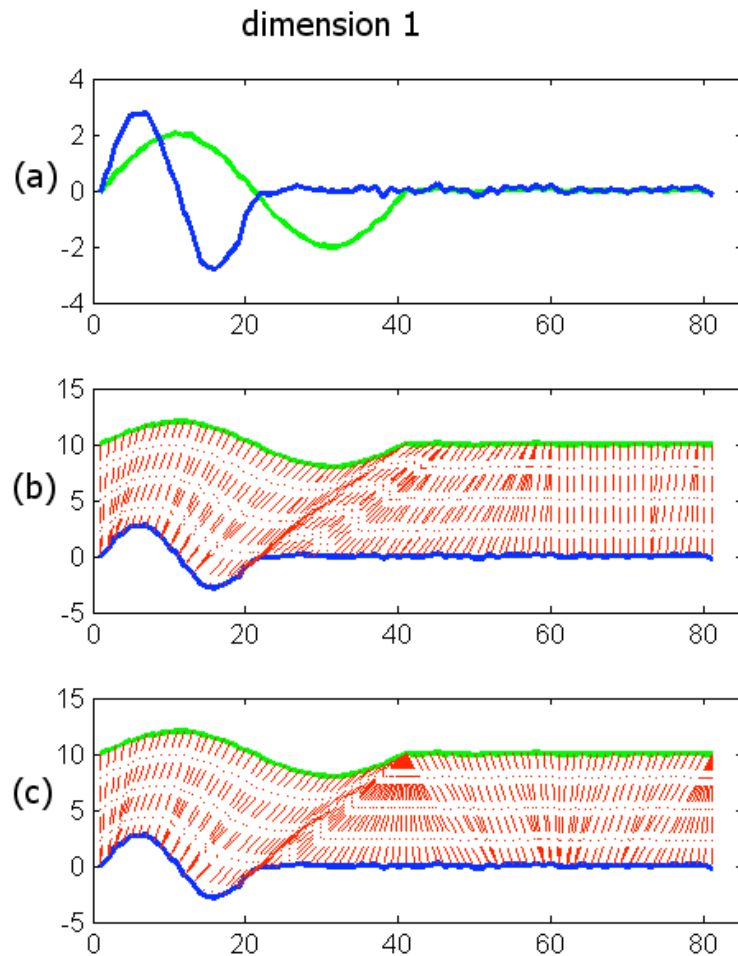
# Multi-Dimensional Dynamic Time Warping

53

- Comment appliquer DTW quand plusieurs grandeurs sont mesurées à chaque instant?
- Modification de la fonction de distance:
  - ▣ Chaque dimension est normalisée avec une moyenne de 0 et une variance de 1
  - ▣ Calcul de la somme des valeurs absolues des différences suivant chaque dimension

# Multi-Dimensional Dynamic Time Warping

54



# Mutli-Dimensional Dynamic Time Warping

55

- Possibilité d'utiliser les signaux et leur dérivée

question

# TP

56

