

# Ecrivez à la souris ! - Les solutions des challengeurs.



par [Equipe DELPHI](#)

Date de publication : 4 décembre 2006

Dernière mise à jour :

Voici quelques présentations de codes des participants au défi numéro 2 : écrivez à la souris !

- I - La solution de TicTacToe
  - I-A - Principe général
  - I-B - Comparaison de 2 symboles
    - I-B-1 - Principe
    - I-B-2 - Normalisation
    - I-B-3 - Comparaison brute
  - I-C - Comparaisons de 2 formes multi-symboles
  - I-D - Interface
  - I-E - Conclusion
  - I-F - Les sources de TicTacToe
- II - La solution d'acness
- III - La solution de Claudius40
- IV - La solution de Jeannot Alpin
- V - La solution de korntex5

## I - La solution de TicTacToe

La méthode employée est "maison" et n'est pas basée sur la distance de Levensthein que je ne connaissais pas.

Quoi qu'avec un peu d'imagination, cette méthode peut se rapprocher de la distance de Levensthein.

Par ailleurs, le code était bouclé avant les indices donnés dans le fil.

Je ne parle ici, que du principe de reconnaissance en lui-même et des quelques options sans rentrer dans les détails techniques de programmation.

### I-A - Principe général

Le principe général est le suivant:

- L'utilisateur trace à main levée un symbole, correspondant à une suite de points récupérés par la souris.
- Ce symbole est comparé à chaque symbole de la bibliothèque.
- Chaque symbole de la bibliothèque est lui-même un symbole classique, c'est à dire une suite de points tracés au préalable à la souris.
- La fonction de comparaison renvoie un écart. Le symbole recherché est celui qui correspond au plus petit écart.

### I-B - Comparaison de 2 symboles



#### I-B-1 - Principe

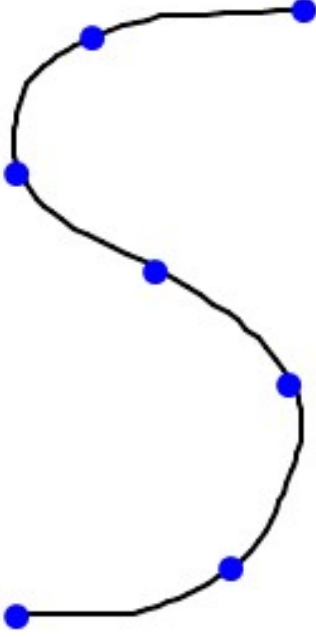
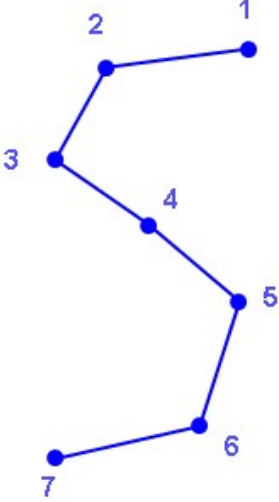
Pour comparer, il y a 2 étapes, et ceci pour les 2 symboles (celui qui vient d'être tracé, et celui de la bibliothèque):


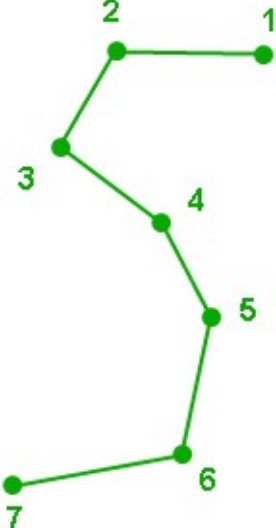
- La préparation du symbole dans une structure particulière
- La comparaison proprement dite.

Plutôt qu'un long discours, les étapes suivantes expliquent le fonctionnement graphiquement.

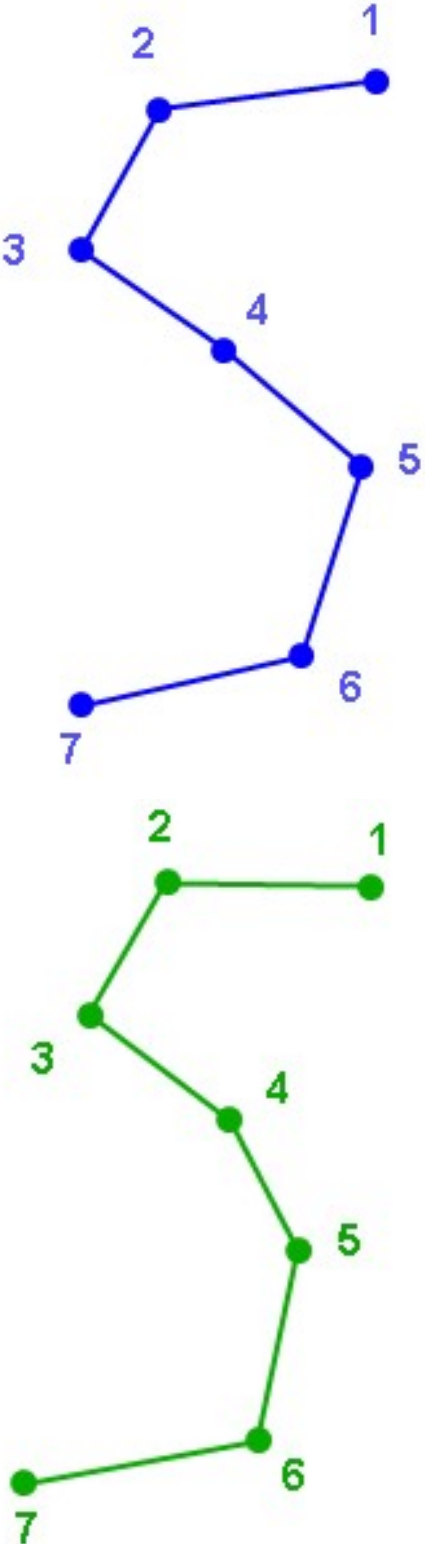
#### I-B-2 - Normalisation

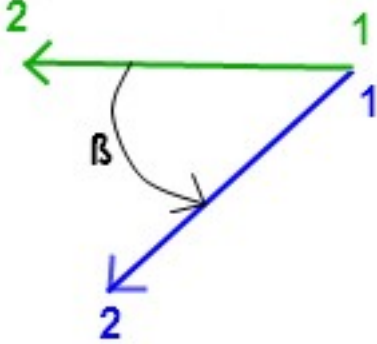
Symbole graphique	Commentaires
	<p>Voici le symbole N°1.</p> <p>Que ce symbole soit tracé à la souris ou inclus dans la bibliothèque, cela revient au même:</p> <p>la source est identique, il s'agit d'une suite de points définissant ce symbole.</p>
	<p>Les points rouges, sont les points récupérés par le "OnMouseMove".</p> <p>Selon la vitesse du tracé par l'utilisateur, leur disposition n'est pas uniforme.</p> <p>Egalement, le nombre de points d'un symbole peut varier.</p>

Symbole graphique	Commentaires
	<p>Les points bleus sont les points dits "normés".</p> <p>Le principe de positionnement de ces points "normés", est:</p> <ul style="list-style-type: none"> <li>• calculer la longueur totale du symbole: somme de tous les petits segments formés par les points rouges</li> <li>• en fonction du nombre souhaité de points à répartir (ici 6), positionner les points bleus en reparcourant la courbe et en "posant" un point tous les <math>(\text{LongueurTotal} / (6-1))</math>.</li> </ul> <p>Maintenant, Chaque symbole dispose d'un nombre de points fixe, donc d'un nombre de segments fixe.</p> <p><u>Remarque:</u> ce nombre de points fixe, est paramétrable dans la zone "précision" en bas à gauche de la fenêtre. Il est à 50 par défaut en réalité, c'est à dire que chaque symbole est systématiquement découpé en 49 segments.</p>
	<p>Le nouveau symbole "normé" est représenté par une succession de vecteurs normés: P1P2, P2P3 etc...</p> <p>Le travail terminé, <b>le symbole N°1 est normé.</b></p>

Symbole graphique	Commentaires
	Considérons un autre symbole N°2, il s'agit de faire exactement le même travail: "Normer" le symbole.
	Voici le résultat, <b>Le symbole N°2 est normé.</b>

I-B-3 - Comparaison brute

Comparaison	Commentaires
	<p>Voici les 2 symboles à comparer, le N°1 en bleu, le N°2 en vert.</p> <p>Chaque symbole est donc une succession de vecteurs orientés.</p> <p>Le nombre de ces vecteurs étant identique, il est donc possible de comparer les angles de chaque vecteur pris 2 à 2 entre chaque symbole:</p> <ul style="list-style-type: none"> <li>• Le 1er vecteur formé par P1P2 du symbole n°1 et le 1er vecteur P1P2 du symbole N°2.</li> <li>• Puis le 2eme vecteur, et ainsi de suite...</li> </ul>

Comparaison	Commentaires
	<p>Considérons le 1er couple de vecteurs formé par les 2 symboles.</p> <p>Ces 2 vecteurs forment un angle orienté, ramené en absolu.</p> <p>La fonction de comparaison renvoie simplement cet angle (en radians dans le programme), compris entre 0 et Pi.</p> <p>Dans le meilleur des cas, on s'aperçoit que si les 2 symboles sont identiques, le retour de la fonction de comparaison renverrait 0 (angle nul).</p>

**La comparaison d'un symbole entier est la somme des comparaisons entre chaque couple de vecteurs des 2 symboles.**

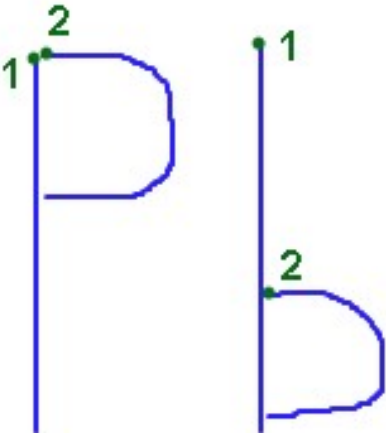
Pour les formes créées avec plusieurs symboles, la recherche s'effectue seulement entre les formes disposant du même nombre de symboles, la résultante de la comparaison est la somme des résultats de comparaison symbole à symbole.

Mais lorsqu'il y a plusieurs symboles pour une seule forme, d'autres problèmes surviennent, énoncés dans le chapitre suivant.

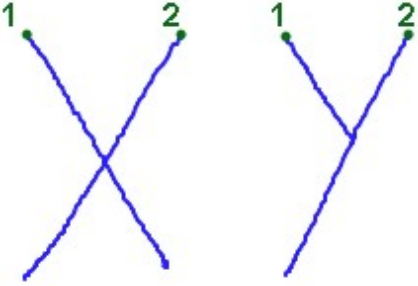
### I-C - Comparaisons de 2 formes multi-symboles

Cependant, il existe encore 2 problèmes pour les formes avec plusieurs symboles. Prenons les exemples suivants:

Chaque forme dispose de 2 symboles, le symbole 1 et le symbole 2. Le point vert symbolise le début du tracé.

Formes multi-symboles	Commentaires
	<p>On s'aperçoit dans ce cas précis, que <u>la somme des résultats de comparaison symbole à symbole</u> donnera a peu près <u>le même résultat</u>.</p> <p>Effectivement, les symboles n°2 de chaque forme, bien que placés différemment par rapport aux symboles n°1, sont identiques.</p> <p>Lors de la comparaison de 2 formes, il y a donc le cumul des comparaisons entre symboles MAIS ajouté à cela, une distance relative entre les 2 symboles de la même forme.</p> <p>Ainsi, la différence est bien faite entre ces 2 symboles, identiques avec</p>



Formes multi-symboles	Commentaires
	la comparaison symbole à symbole, mais différent en y ajoutant cette distance relative inter-symbole.
	<p>On s'aperçoit dans ce cas précis, que non seulement, la somme des comparaisons entre symbole donnera à peu près le même résultat (2 traits obliques dans le même sens), MAIS qu'en plus, la position relative des 2 symboles est identique. (si on se base sur le 1er point de chaque symbole).</p> <p>A l'instar de la distance relative citée ci-dessus, les tailles relatives entre symboles sont prises en compte</p> <p>Typiquement ici, le ratio entre le symbole 1 et 2 du 'X' sera de 1 alors que le ratio entre le symbole 1 et 2 du Y sera de 2 ( car la 2ème barre du Y est 2 fois plus longue que la 1ère barre).</p>

### Résultat final:

La comparaison de 2 formes, renvoi un indice constitué de:

- la somme des comparaisons entre chaque symbole
- + un indice de position relative inter-symboles (nul si forme mono-symbole)
- + un indice de taille relative inter-symboles (nul si forme mono-symbole)

La forme (multi-symboles ou non) tracée, est comparée à chaque forme de la bibliothèque, donnant un résultat pour chaque comparaison.

Le plus petit résultat correspond à la forme la plus proche et est sélectionnée.

## I-D - Interface

### Bibliothèque

La bibliothèque garde dans un fichier INI **la trace brute** (c'est à dire les points réels capturés dans le OnMouseMove du programme) des symboles.

Au chargement de la bibliothèque et par un soucis d'optimisation, l'étape de normalisation est effectuée une fois pour toute pour tous les symboles.

Il est possible d'établir des bibliothèques différentes. Le fichier de configuration .INI garde chaque bibliothèque dans une section différente.

### Morphing

Il a été facilement réalisable à partir du moment où tous les symboles sont "normés".

En effet, les symboles disposant du même nombre de points, pour créer l'effet de morphing, il a suffit pour chaque

couple de points de faire varier par interpolation, un 3ème point pour la forme intermédiaire.

### A noter

Il est possible d'afficher les formes gardées en bibliothèque, indispensable à mon sens pour créer sa propre bibliothèque ainsi que contrôler ses propres tracés.

Le redimensionnement de la fenêtre est pris en charge, les formes étant normées, il a été possible de les afficher de manière uniforme dans un espace Canvas de n'importe quelle dimension.

## I-E - Conclusion

Bien qu'au premier abord, ce problème paraisse compliqué, il ne l'est pas tant que cela comme le montre les explications.

Le programme peut subir moult optimisations et améliorations, comme:

- Optimiser les valeurs forfaitaires pour les comparaisons inter-symboles (définies par empirisme)
- Effectuer différentes rotations des formes, avant de comparer...
- Effectuer des comparaisons par groupe de vecteurs, au lieu de vecteurs individuels...

Quoi qu'il en soit, cela a été un défi intéressant, en témoigne le nombre de participants.

TicTacToe le 18/10/2006

## I-F - Les sources de TicTacToe

<a href="#">Téléchargez le code source de TicTacToe</a>	
<a href="#">Téléchargez l'exécutable de TicTacToe</a>	

## II - La solution d'acness

<a href="#">Téléchargez le code source d'acness</a>	
<a href="#">Téléchargez l'exécutable d'acness</a>	

### III - La solution de Claudius40

Téléchargez le code source de <a href="#">Claudius40</a>	
Téléchargez l'exécutable de <a href="#">Claudius40</a>	

## IV - La solution de Jeannot Alpin

<a href="#">Téléchargez le code source de Jeannot Alpin</a>	
<a href="#">Téléchargez l'exécutable de Jeannot Alpin</a>	

## V - La solution de korntex5

<b>Téléchargez le code source de <a href="#">korntex5</a></b>	
<b>Téléchargez l'exécutable de <a href="#">korntex5</a></b>	