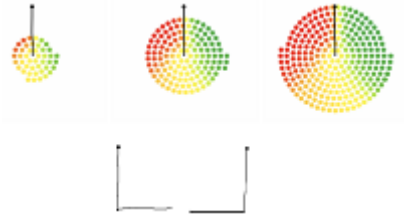
Per Ola Kristensson | Continuous gesture recognition

Click here for the continuous gesture recognizer for the Kinect.

Click here for a port of this algorithm to the programming language Lua.

Introduction

We have developed a continuous gesture recognition algorithm. It enables incremental recognition of pen strokes, touch-screen gestures and other 2D trajectories. Given a set of gesture templates the algorithm outputs a probability distribution over this template set as a function of a user's partial or complete articulation of a stroke.



 It enables interactive systems to eagerly process users' input before users have completed their input gestures. This opens up a vast array of new applications for sketching systems and touch-screen interfaces. For example, the illustration to the right uses the algorithm to visualize the probability space for a user's partial gesture being correctly recognized if the user moves linearly from the current position to an arbitrary point on the touch-screen (green means high probablity and red means low probability). In the illustration, the user is intending to gesture the left-most gesture template in the figure (starting points are indicated by solid dots) and the visualization reveals that the probability of a correct recognition result is dramatically higher if the user moves to the right. For more information and other examples of how our algorithm can be used, please see our research paper.

We have released the source code to this algorithm as open source with the hope to see it used in many interactive systems. The source code for the algorithm is self-contained within a single Java file and does not rely on any external dependencies beyond very basic Java libraries that are found on just about any reasonably modern Java platform (the dependencies are the following classes and interfaces in `java.util`: `ArrayList`, `Collections`, `Iterator` and `List`).

Interactive Demonstration

If you have a reasonably modern Java installation you can try the continuous recognition algorithm by clicking here (warning: a Java applet will load inside your browser window).

Reference

If you use this code please remember to cite our paper:

Kristensson, P.O. and Denby, L.C. Continuous recognition and visualization of pen strokes and touch-screen

gestures. In *Proceedings of the 8th Eurographics Symposium on Sketch-Based Interfaces and Modeling (SBIM 2011)*. ACM Press: 95-102. (pdf)

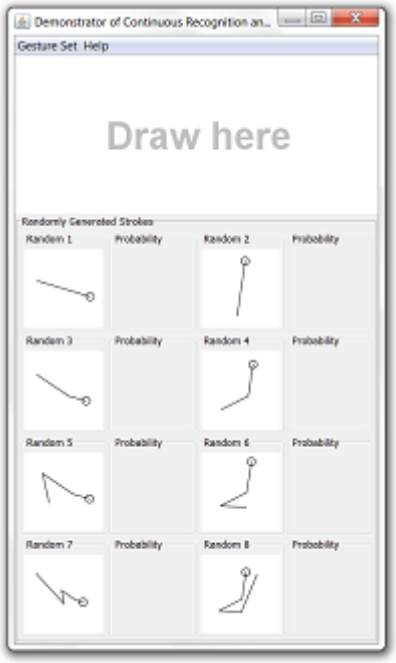
Source code

The complete source code for the continuous gesture recognition algorithm is in this self-contained Java file:

ContinuousGestureRecognizer.java

Demonstrator

To help understand how to use the continuous gesture recognition algorithm I have implemented a graphical demonstrator application that you can play around with. Put this file in the same directory as `ContinuousGestureRecognizer.java`:

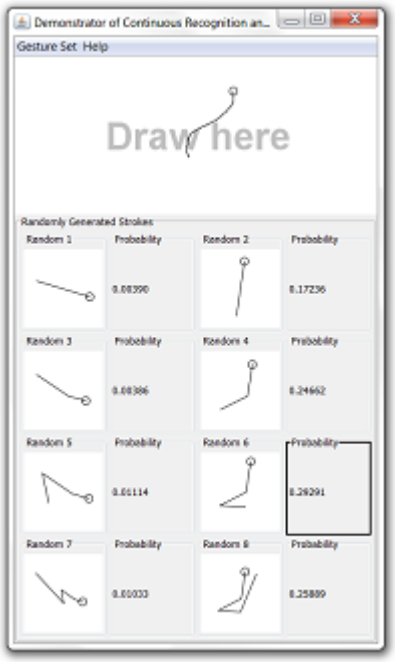


Demonstrator interface Demonstrator.java

To start the demonstrator type `java Demonstrator` in a console window. You need to have Java installed for this to work. Alternatively, if you have a reasonably modern Java installation you can run it in your browser by clicking here (a Java applet will load).

The demonstrator provides you with three template gesture sets. You can choose which template set you want to use by selecting it in the `Gesture Set` menu. The `Help` menu provides a shortcut to our research paper and access to the obligatory `About` dialog box.

The thumbnail icons in the interface show the templates. The circles denote the starting points for the templates. Next to each template the interface displays the probabilities for this template as a function of your input gesture articulation.



Partial gesture and best match To draw a gesture click and hold down the left mouse button and draw in the designated drawing area. While you draw you will notice that the probabilities for the templates below the drawing area changes.

For example, in this screenshot to the left I have partially gestured template number six in the set. To indicate that template number six is the best match the system has drawn a solid black frame around its probability value.

The demonstrator comes with three template sets. The first is a randomly generated template set that consists of two progressively extended base templates. You can generate a new random template set by reselecting this set in the `Gesture Set` menu. This set may be interesting to play around with because it highlights the importance of using an end-point bias term in the continuous gesture recognition algorithm. This bias means that the algorithm will prefer shorter complete templates that are prefixes of other longer complete templates. The second template set consists of directional strokes. It is mainly useful for testing purposes. The third template set consists of four progressively increasing test templates. To find out more about this set download our research paper and read the paragraph on page 99 that discusses figure five in the paper. The template set for figure five is this third set in the demonstrator.

License

The code is open source and licensed under the MIT license. The license is included in the source code files above. Among other things, it allows commercial and non-commercial use of the code free of charge.

Acknowledgements

This work was supported by the Engineering and Physical Sciences Research Council (grant number EP/H027408/1). L.C. Denby's internship was funded by a donation from Nokia.

Contact

Email: `kristensson @ acm.org`

Dr Per Ola Kristensson
Department of Engineering
Trumpington Street
Cambridge CB2 1PZ
United Kingdom

Last updated: September 2, 2014.