# Multicolour Sketch Recognition in a Learning Environment

Liam Don Ioannis Ivrissimtzis

*Department of Computer Science, University of Durham, United Kingdom*

## Abstract

*Virtual physics environments are becoming increasingly popular as a teaching tool for grade and high school level mechanical physics. While useful, these tools often offer a complex user interface, lacking the intuitive nature of the traditional whiteboard. Furthermore, the systems are often too advanced to be used by novices for further experimentation. In this paper we describe a physics learning environment using multicolour sketch recognition techniques on digital whiteboards. We argue that the use of coloured pens helps to resolve several ambiguities appearing in single colour sketching interfaces. The recognition system is based on a combination of Support Vector Machines and rule based methods. The system was evaluated using a constructive interaction method, with users completing a set task.*

## 1. Introduction

S*ketch recognition*, that is, computer recognition of hand drawn shapes, is an area of considerable recent research activity, driven mainly from applications. Indeed, the use of sketch recognition on popular electronic devices, such as Tablet PCs and Smart Boards, mean that advances in the area can have immediate practical implications. From this applied point of view, the general goal of the research in sketch recognition is to make drawing on a computer as close as possible to drawing on a piece of paper, or a whiteboard, by placing as few constraints on the user as possible, while still enabling robust and accurate recognition of the hand drawn shapes [1].

The field of sketch recognition has evolved out the more general and well researched field of pattern recognition. Rather than performing recognition on a bitmap input, sketch recognition methods generally analyse user strokes or gestures to produce results. In the past, this has meant that special gestures had to be learned by the user, and performed using the same stroke orientation or speed. However, the latest implementations on geometric shape recognition are now extremely robust, and have largely solved these problems as well as employing context sensitivity to allow shapes to be constructed using multiple strokes, or even multiple shapes from a single stroke. An excellent example of a robust implementation demonstrating both these features is the PaleoSketch system [14]. Researchers are also investigating the usability benefits of sketch-based interfaces, often for surprisingly specialized and complex applications [15].

In this paper, setting our target application as a teaching tool for primary level mechanical physics, we describe a sketch recognition based interface for 2-dimensional shapes. The interface is build around the *Smart Board*, which is a digital whiteboard/drawing screen using optical detection, see Figure 1.

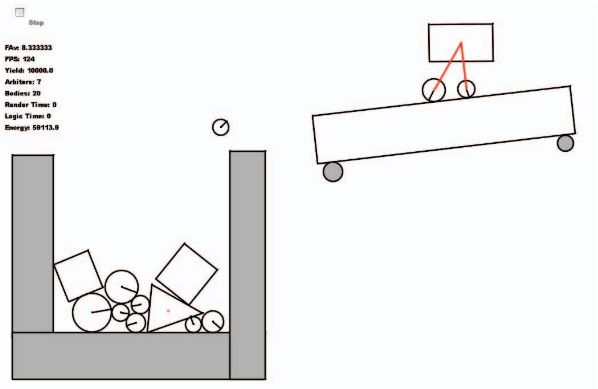
**Figure 1. The Smart Board device.**

Combining Support Vector Machines (SVM) and rule based methods, the system supports the recognition of simple 2-dimensional shapes, such as squares, circles and polygons. To increase the interactivity of the application, these shapes are beautified as they are drawn, providing the user with instant feedback as to the success of the recognition. Finally, the user can then toggle the system into a physics simulation mode, in which the sketched shapes behave realistically in a physical 2D environment, see Figure 2.

Our implementation does not seek to increase the usability of the drawing interface by emulating the context-sensitivity demonstrated by PaleoSketch. Instead, our main contribution is that we utilize the multiple coloured pens of the Smart Board device to afford users a real-world metaphor for input of different classes of

**978 -1-4244-1866-4/08/$25.00 © 2008IEEE**

objects (shapes, constraints and forces). By using multiple coloured pens, we eliminate the need for context-sensitivity, and the problems associated with it, in our recognition engine. The user can also erase recognized shapes using the Smart eraser, and even move objects around by dragging their fingers on the board.



```
FAv: 8.333333
FPS: 124
Yield: 10000.0
Arbiters: 7
Bodies: 20
Render Time: 0
Logic Time: 0
Energy: 59113.9
```

**Figure 2. Our Sketch Board running in simulation mode.**

Far from complicating the user experience, we argue that the use of a real-world "pen" analogy can aid user understanding of the input options available, perhaps to a greater extent than the "invisible" method of context-sensitivity. The implementation aims to be intuitive enough to use both in a classroom teaching environment, and at home as a personal learning tool.

## 2. Related work

The system presented in this paper is primarily informed by the ASSIST recognition system developed by Alvarado [2], which in turn was a development of RecSystem [3]. ASSIST allows the user to sketch a mechanical physical system in much the same way as our system does, but it has a greater context sensitivity, which we believe opens it up to ambiguity and therefore sometimes false recognitions. The reason for this context sensitivity is that the ASSIST system uses a single pen for drawing of all objects, which creates a far greater number of possible results for each stroke. We aim to combat this by making use of the multiple coloured pens of the Smartboard, for different stroke types (e.g. base objects, constraints and forces). While this is a user constraint, which runs in opposition to Hammond and Davis' definition of a good sketching interface [1], we expect that the users will accept and find intuitive the idea of different coloured pens providing different functionality.

*Neural networks* is an obvious choice for sketch recognition, and their success has been repeatedly proved in the more general setting of pattern recognition [4]. However, several of the most popular neural network methods for pattern recognition can not be directly applied to our problem, as they have been developed to work with images rather than strokes. Yaeger [5] has suggested a neural network approach using strokes, in his evaluation of sketch recognition on the Apple Newton device.

*Rule based methods* for sketch recognition rely on creating specific rules to characterize particular shapes. As a simple example, an equilateral triangle has its three corners at equal distance to each other. A rule based method proposed by Peng et. al. [6] takes into account the positions of endpoints of user strokes, and uses this information to determine what shape has been drawn. This method works very well, but does not take into account the timing of strokes, which can be very useful in a heuristic, particularly for broad classification into general shapes. Mackenzie [7] lists some methods based on timing of strokes, although his main focus is on agent based recognition.

The *support vector machine* is a statistical method originally proposed by Vapnik in [8]. It is a supervised learning technique with similarities to the neural networks, and can be used to solve the classification problem for two classes. Viewing the sets of data as two vectors in an *n*-dimensional space, a single SVM splits the data into two classes, computing a hyperplane that separates them. Combinations of SVMs can be used to solve the classification problem for *n* classes. For example, we can use *n* SVMs, each one trained to separate one particular class from all the other classes. For a more robust classifier, we can use one SVM for each pair of classes, that is, n·(n-1) / 2 SVMs in total. Each SVM is trained to separate the objects of two specific classes, ignoring all the other classes. Finally, the n·(n-1) / 2 SVMs are combined through a majority voting system to produce a classifier for the *n* classes. It has been shown that the pair wise coupling scheme is more robust than the one SVM per class scheme [9].

Wenyin et. al. [10] offer a quantitative comparison of SVMs, rule based methods and neural networks. SVMs is shown to be the most robust method overall, although, in their experiments, rule-based methods achieved 100% recognition rates for elliptical shapes. Neural networks perform less well, which is surprising considering their effectiveness in more general machine vision problems. Based on this work we have chosen to use a combination of SVMs and rule-based methods.

Sketch beautification is an important element of the sketching process. Because a user's sketches can be rather rough and unsuitable for design work, it is usually helpful for the computer to "tidy up" lines and connect them cleanly, see Figure 3. Except of improving the visual quality of the sketch, beautification also helps the users understand whether a shape has been recognised or not by the computer [13]. We believe that beautification should be applied after every stroke, as the immediate feedback can be very helpful in guiding the user through the sketching process. Without this feature, problems could accumulate, reducing the overall accuracy of the system.

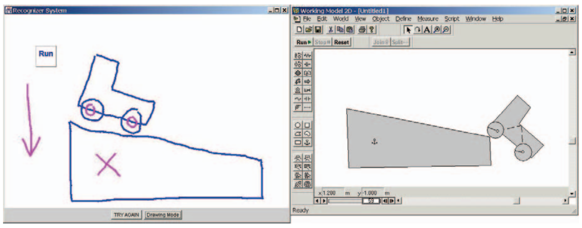


**Figure 3. Sketch beautification in the ASSIST system.**

As a fully functional sketch recognition utility can be extremely complex, we tried as far as possible to reuse existing software and libraries that are freely available in the web. One technology which we have used in our implementation is the Diva sketch system, developed at Berkeley California [11]. It features a sketching interface, with a data structure that stores user input as timed strokes, and a basic recognition framework which allows various methods of recognition to be plugged in. An excellent symbol recognition system, based on SVMs, has been proposed by Hse et al. in [12]. We found that Hse's system is excellent for broad recognition of shapes, and we make use of it as an initial broad classifier between circles, rectangles and general polygonal shapes. Following this step, we use a rule-based approach, using some stroke segmentation and geometry interpolation methods demonstrated in the ASSIST system. We combine these two distinct recognition stages to produce a robust overall recognition method.

## 3. The sketch recognition system

### 3.1 The Smart Board hardware

The Smart Board is the main hardware around which the Sketch Board system is built (see Figure 1). It consists of a screen and four pens, coloured black, red, blue and green. The pen movement is tracked on the screen by two cameras located at the corners. At any particular time only one pen can be traced on the screen.

Another limitation is that there is no active detection system for the colour of the pen in use. Instead, at the bottom of the screen there are four coloured pen-cases matching the colours of the pens. Each pen-case has an optical sensor and the colour in use is the colour of the pen-case which last had a pen removed from it. This limitation can create user confusion when a pen is not returned to its pen-case after use, or when pens are misplaced in pen-cases of different colour.

### 3.2 Architecture

The high level architecture of the Sketch Board system is described by the diagram in Figure 4.

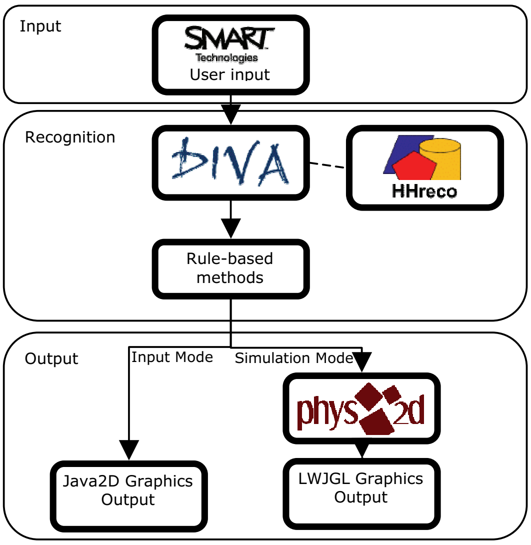


**Figure 4. Diagrammatic overview of the Sketch Board.**

The system relies on a number of external libraries and codebases, some of which are used as black-box solutions and some of which have been modified.

The sketch recognition and simulation pipeline works as follows. The Smart SDK of the Smart Board captures the user input in the form of time strokes. The Diva framework with the HHreco plugin uses SVM for a broad classification of the time strokes into few categories of objects. The main recognition system, using code from the Microsoft's version of ASSIST, uses rule based methods to complete the object recognition and beautification. Then the objects are sent to the Phys2D physics simulation engine whose output is again displayed on the Smart Board.

The main novelty of the Sketch Board is the use of multiple coloured pens, a physical eraser and the user's own fingers to indicate different modes of input. We use three modes of input, black, red and blue for shapes, constraints and forces, respectively. Additionally shapes can be removed with the eraser or moved by dragging a finger on the board. Next we describe each mode of input in detail.