Andrew Moran
3/04/15
6.835: Project 2

1. All files in Stellar submission under andrewmo@mit.edu

2. Below summarizes the results on the accuracy of the system after adjusting parameters and/or baselines.

Type	Description	Digits	Shapes	Circuits
Default	h = 24, t = 10, s = 1.0	94.25%	99.25%	83.00%
Optimal	h = 20, t = 2, s = 1.5	**98.50%**	**98.50%**	**91.50%**
Pixel Only	All intensities = 1.0	95.75%	98.75%	83.25%
No Smoothing	No Gaussian smoothing	91.50%	96.50%	78.00%

The three main parameters I played around with are the following:
- h – `h`: feature dimension
- t – `SLOPE_WINDOW`: size of window when computing tangent slopes
- s – `SIGMA`: bandwidth of Gaussian filter for smoothing

Adjusting the parameters directly affected the performance of the system. Significant improvement was made from adjusting the `SLOPE_WINDOW.` Minimizing the window forced tangent calculations to rely more heavily on regions very close to one another. `SIGMA` slightly changed results as well. In order of performance improvement, circuits increased the most, followed by a small increase with digits. Shapes actually decreased in performance (but very slightly).

The default parameters showed promising results. However, after testing through multiple values for each parameter, the optimal results came with the following: `h=24`, `SLOPE_WINDOW=2`, `SIGMA=1.5.`

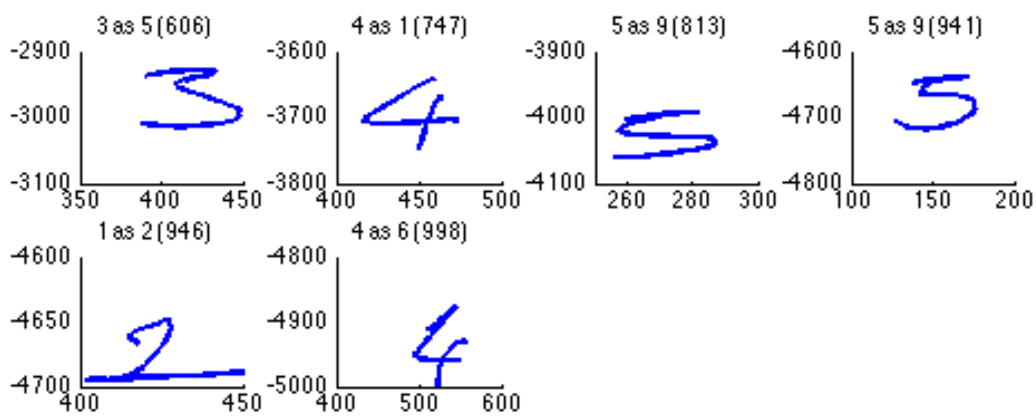
The pixel-only baseline actually made better results as compared to just applying the default parameters. Shapes showed a slight decrease in accuracy though. Perhaps utilizing angles and having a large initial `SLOPE_WINDOW` obscured details that had valuable information.

The no smoothing baseline performed the worst. This was expected, because without a blurry/fuzzy component, less forgiveness is provided. Therefore, penalties are stricter for minor misalignments given our distance metric.

3. Comparing Accuracy of the system with best parameters on all datasets. For convenience, matching a symbol to its correct are represented below.

```
Digits = digits 0-9
Shapes = [ 0=arch, 1=callout, 2=cube, 3=cylinder, 4=ellipse,
           5=heart, 6=hexagon, 7=moon, 8=parallelogram,
           9=pentagon, 10=square, 11=trapezoid, 12=triangle]
Circuits = [0=ac-source, 1=battery, 2=bjt, 3=capacitor,
            4=current-source, 5=diode, 6=ground, 7=jfet,
            8=resistor, 9=voltage-source]
```

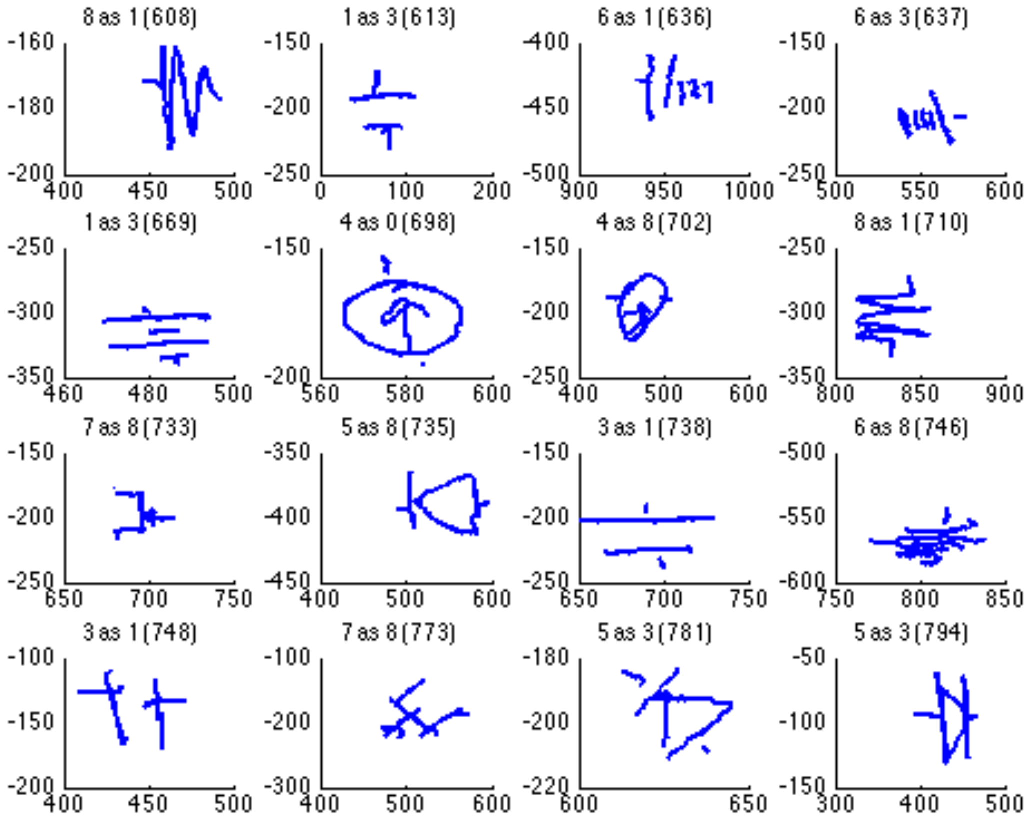
Digits




```
accuracy = 0.9850
errors =   606   747   813   941   946   998
```

Although very accurate, the system did make some mistakes worth noting. Errors 813 and 946 are understood. There are features in the symbol that can potentially match the 5 as a 9, or a 1 as a 2. Others could be linked to orientation problems or there was not enough evidence to match the symbol to the training set. In that case, the IDM distance metric could have improved performance.

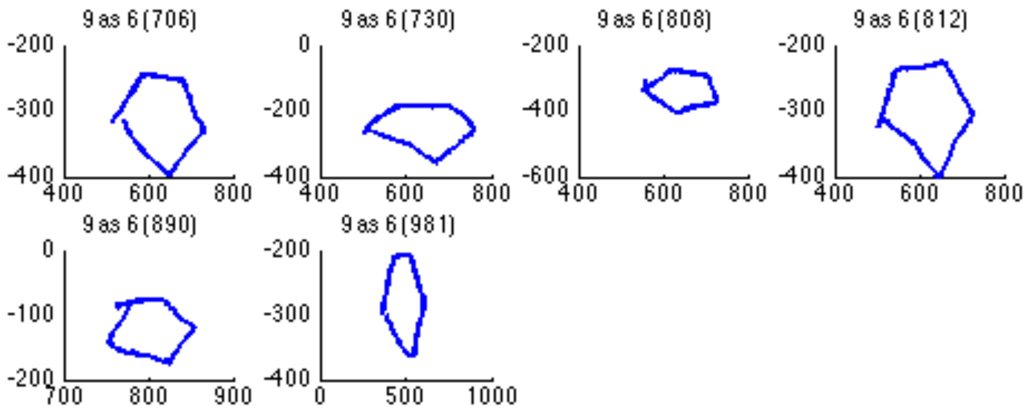
CIRCUITS




```
accuracy = 0.9150
errors =    608    613    636    637    669    698    702    710    733
            735    738    746    748    773    781    794    813    857
            858    861    867    871    889    892    893    903    938
            956    959    965    974    981    990    997
```

Many errors were found for the hand-drawn circuit elements. This is expected because there are a lot of elements that share common features such as parallel/perpendicular lines, and if drawn not precisely, could be misinterpreted. Most noteworthy are 613 and 738. Distinguishing a battery and a capacitor share similar qualities such as paralleled lines, however are only distinguishable by line length, which requires a string attention to detail. I could make the common mistake myself. Other mistakes could be associated with elements slightly disoriented so there are not aligned properly with the training set.

SHAPES




```
accuracy = 0.9850
errors =   706   730   808   812   890   981
```

Out of all the datasets, the shapes proved to be most robust with a great accuracy rating. All the errors resemble that the pentagons are being interpreted as a hexagon. As we saw with the last mini-project in stroke segmentation, rounding corners and speed of stroke play an impact on sketch understanding. Some strokes can be over-evaluated producing more corners (or overlooked missing essential corners). The orientation and skew of the shapes could have additionally misinterpreted these shapes, proving how drawing very freely can make the system miss key features.

4. There are many steps that can be considered to improve the accuracy of the system. Most notable is accounting for rotations. The digits dataset have a known rotation so therefore would not be that great of improvement. However, shapes and circuit diagrams can be in any orientation. The training set only accounts for a few known rotations, however, if this can be extended to allow more rotations to be considered, which could significantly improve the performance of the system. Mentioned earlier for the digits dataset, IDM could be very useful for applying a more robust distance metric. In addition, further playing around with Gaussian blurring and downsampling could make more improvements on all datasets.