

Assigning Confidence Values to Geometric Constraints

Joshua Johnston and Tracy Hammond
Department of Computer Science and Engineering
Texas A&M University
{jbjohns, hammond}@cse.tamu.edu

ABSTRACT

In this poster, we demonstrate our method for assigning confidence values to geometric constraints used to recognize hand-drawn sketches. We implement a subset of the constraints devised by Hammond and Davis for their LADDER sketch recognition language and system. We divide the constraints into two disjoint sets, those that use a distance-from-zero approach to solving confidence, and those that use a relational approach to solving confidence. A half-Gaussian probability distribution is used to assign confidence values to the former group, and a sigmoid is used to assign confidence values to the latter group. We justify our choice of these methods, explain their use, and demonstrate their effectiveness.

INTRODUCTION

In the LADDER language [1], constraints are applied to a set of primitive shapes, such as lines, arcs, and curves, that are recognized by other sketch recognition systems [2]. For example, to recognize an arrow shape, the system might require three lines (*shaft*, *head1*, and *head2*), constrained as given in Table 1.

The notation *shaft.end2* specifies an endpoint for the given line *shaft*. The constraints require the line called *shaft* to be coincident at an endpoint with the the two lines *head1* and *head2*. The two lines forming the head of the arrow are to meet at an endpoint and form an acute angle, and are also to be the same size (length). We specify that the shaft of the arrow is to be longer than the lines forming the head of the arrow. Figure 1 shows an example sketch that meets these constraints.

In the LADDER system, constraints are computed on a binary basis (Equation 9). Each constraint is assigned a hard-coded threshold, and this threshold is used to determine if a constraint holds or not. For example, the *horizontal* constraint's threshold is 15° . This means that if the angle of a line relative to the *x*-axis is $< 15^\circ$ from horizontal, it is considered to be horizontal. In our confidence-based im-

```
shape line shaft
shape line head1
shape line head2
shaft.end2 coincident head1.end1
shaft.end2 coincident head2.end1
head1 acute-meet head2
head1 same-size head2
shaft longer head1
```

Table 1. Example constraint description for a simple arrow shape illustrated in Figure 1.

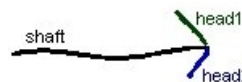


Figure 1. arrow shape defined as three lines satisfying the constraints given in Table 1.

plementation of the constraints, we use the same thresholds implemented in LADDER. Instead of using the thresholds to make a binary decision, however, we use them to assign a confidence that measures how certain the system is that the constraint holds.

With a confidence value assigned to constraints and the recognized shapes built using those constraints, we are able to add additional functionality to a sketch recognition system that might have been more difficult or impossible in a system using only binary constraints. We can provide an ordered *n*-best list of recognition interpretations, each with its own measure of confidence. We can also use the confidence values to combine the results of multiple recognition techniques using ensemble methods. Assigning confidence values to individual constraints also allows for more flexibility within the recognition algorithm itself. Rather than relying on binary decisions, an algorithm can use our constraints to allow for arbitrary levels of uncertainty in its decision and explore a larger solution-space. This might enable the algorithm to avoid local-optima and find a globally optimal solution.

METHODOLOGY

To make the discussion of constraints clear, we need to define some terminology. We treat constraints as functions

$$c = f(v), \quad (1)$$

with the input v sometimes referred to as the value for the constraint. In the case of the constraint `horizontal`, v is relative to the angle of the line in question (see below for a discussion of the normalization of v , Equation 11). The output of the function, c , is the confidence the constraint holds for the given value.

Distance-from-zero Constraints

As stated earlier, we divide constraints into two categories. The first category uses a distance-from-zero approach to solving confidence. For these types of constraints, the optimal maximal confidence is given when $v = 0$ (Equation 5). The confidence for the constraint is computed using a half-Gaussian probability distribution (Figure 2). A half-Gaussian distribution is simply a Gaussian distribution with mean $\mu = 0$, parameterized on the standard deviation σ^2 , with input v constrained to be ≥ 0 .

$$c = f_{\text{hg}}(v) \quad (2)$$

$$= \frac{2\sigma}{\pi} \exp\left(-\frac{v^2\sigma^2}{\pi}\right) \quad (3)$$

$$= 2 * N(0, \sigma^2, v). \quad (4)$$

$N(\mu, \sigma^2, x)$ is the Gaussian/Normal probability distribution with mean μ and standard deviation σ^2 , evaluated on the input x . For our constraints, we assign $\sigma^2 = 1$. See below for a more detailed explanation of our choice of σ^2 .

The constraint `horizontal` is an example of this type of distance-from-zero constraint. The value v for this constraint is relative to the absolute value of the angle of a line in relation to the x -axis (we do not use slope since the slope of a vertical line is positive infinity). The line is perfectly horizontal if its angle is $v = 0$, since

$$\arg \max_v f_{\text{hg}}(v) = 0. \quad (5)$$

As v increases (the angle moves more toward vertical), the confidence c that the line is horizontal decreases toward 0. Figure 3 shows different lines with various confidences of being horizontal.

Relational Constraints

The second category of constraints uses a relational approach. These constraints use a sigmoid function, illustrated in Figure 4, to compute confidence values.

$$c = f_{\text{sig}}(v) \quad (6)$$

$$= \frac{1}{1 + e^{-v}} \quad (7)$$

We call these constraints relational because they are used to compare two values, v_1 and v_2 , against each other. We must consider the case of negative numbers, which we do not with f_{hg} . As v_1 grows larger than v_2 , the confidence c increases toward 1. As v_1 grows smaller than v_2 , the confidence decreases toward 0. The input to the constraint function is the difference in the values

$$v = v_1 - v_2. \quad (8)$$

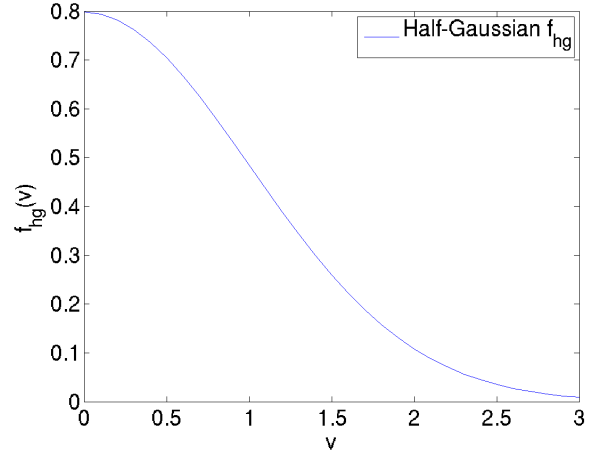


Figure 2. A half-Gaussian probability density function, defined for $v \geq 0$, which is used to assign a confidence to constraints using the distance-from-zero approach. The x -axis is the normalized value (Equation 11) and the y -axis is the confidence.

A value of $v = 0$ is ambiguous—it is not clear whether or not the constraint should hold since we cannot tell any difference between the values v_1 and v_2 .

An example of a relational constraint is `above`, which looks at the y coordinates of two points on the screen to determine if one is above the other. The y -coordinate of the first point is v_1 , and the y -coordinate of the second point is v_2 . The difference in the coordinates is $v = v_2 - v_1$.¹ If $v = 0$, meaning the two points are at the same vertical “height,” the confidence that one is above the other is 50% ($f_{\text{sig}}(0) = 0.50$). As v increases ($v > 0$), and the first point moves above the other, the confidence that the first point is above the second increases. Inversely, as v decreases ($v < 0$), and the first point moves below the second, the confidence that the first point is above the second decreases. Figure 5 shows various pairs of shapes that have different confidences of the first shape being above the second.

Normalizing v

We maintain the idea of a threshold as defined in the LADDER system’s implementations of the various constraints. In LADDER, each constraint has a threshold t used to determine if the constraint was true or false.

$$c = f_{\text{LADDER}}(v) \quad (9)$$

$$= \begin{cases} 1, & v < t \\ 0, & v \geq t \end{cases} \quad (10)$$

We use these same thresholds in our system. However, instead of using them to make a binary decision, we use them for normalization purposes. There are many types of constraints, and many different threshold values of various magnitudes. In order to maintain the consistency of confidences

¹The terms of the difference equation are flipped here (compared to Equation 8) because in screen coordinates, the y -axis is “flipped” and something that is `above` will have smaller y values, the opposite of the Cartesian coordinate system.

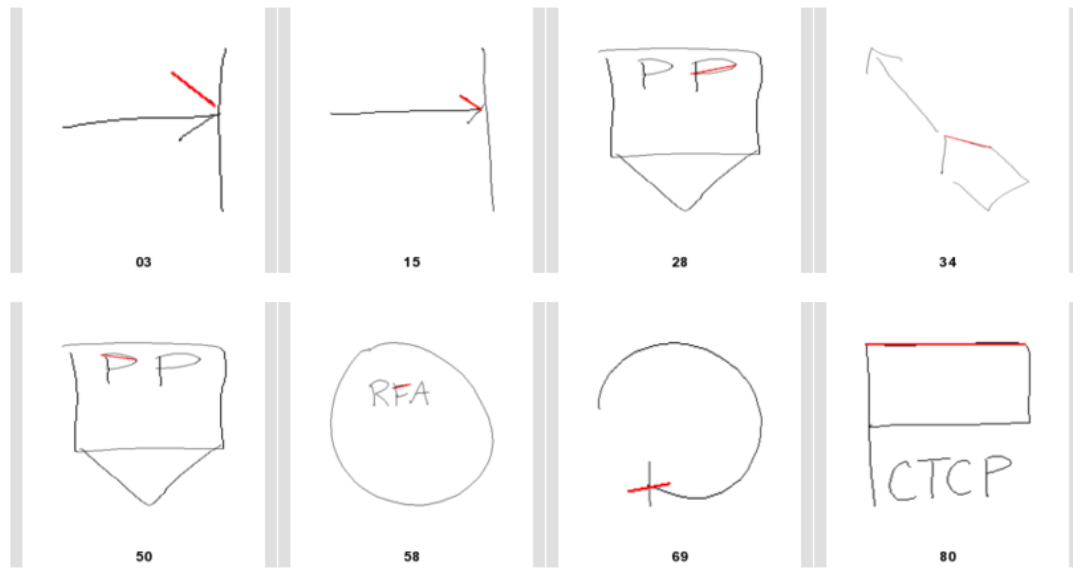


Figure 3. Images of the horizontal constraint solved on various sketches from a military Course of Action symbol domain. Read the images as “Is the red line horizontal?” The general confidence for the answer “Yes” is given below each image. The general confidence “0” means any confidence value in the range $c \in [0 \dots 10)$, and “70” means $c \in [70 \dots 80)$, etc.

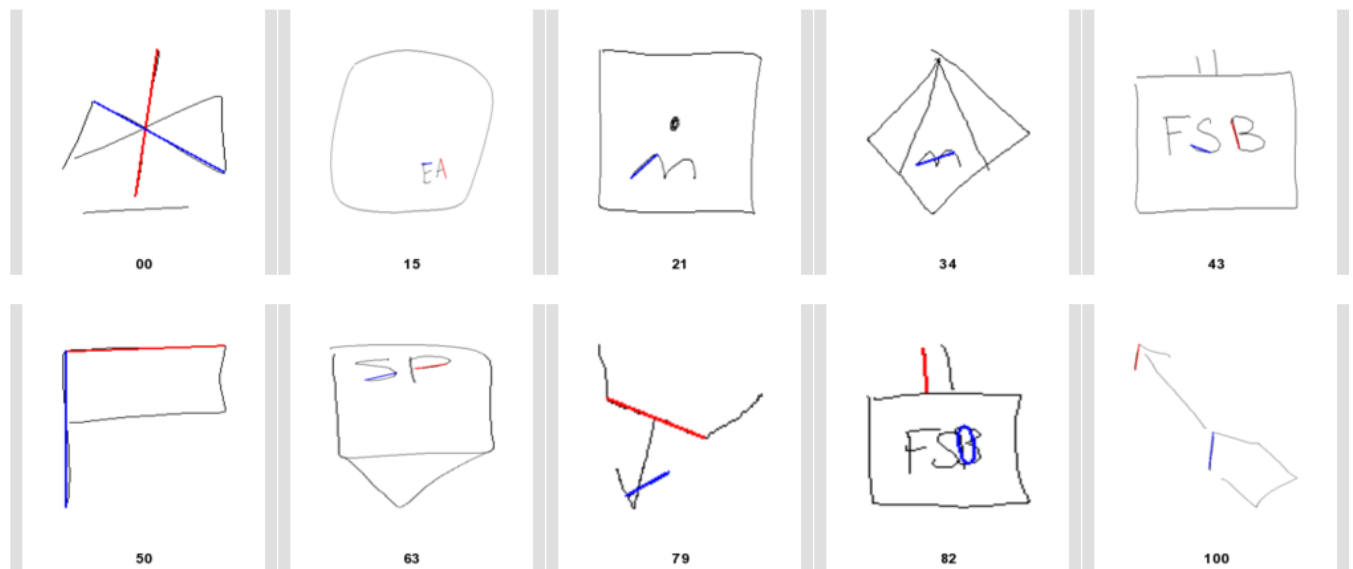


Figure 5. Images of the above constraint solved on various sketches from a military Course of Action symbol domain. Read the images as “Is the red shape above the blue shape?” The general confidence for the answer “Yes” is given below each image. The general confidence “0” means any confidence value in the range $c \in [0 \dots 10)$, and “90” means $c \in [90 \dots 100)$, etc.

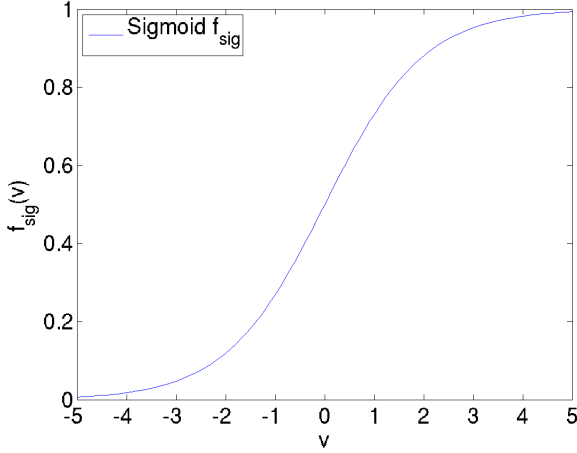


Figure 4. The sigmoid function is used to compute confidence for relational constraints. The x -axis is the normalized value (Equation 11) and the y -axis is the confidence.

returned by these different constraints, we normalize a value by the threshold for the constraint before solving for the confidence. That is,

$$v' = \frac{v}{t} \quad (11)$$

and

$$c = f(v'). \quad (12)$$

If we did not perform this type of normalization, two constraints with different thresholds using the same confidence function f would not give the same confidence values, even if their inputs were comparable. For instance, `horizontal` has a threshold of 15° , while `acute-angle` has a threshold of 45° . Even though the two constraints both use the half-Gaussian confidence function f_{hg} , they will not return the same confidence value if both are exactly one threshold away from optimal. If we set $\sigma^2 = t$ for each constraint, then

$$f_{hg}(15) = 2 * N(0, 15, 15) = 0.0323$$

(for `horizontal`) and

$$f_{hg}(45) = 2 * N(0, 45, 45) = 0.011$$

(for `acute-angle`). But, if we set $\sigma^2 = 1$ and normalize our values as in Equation 11, then all constraints can be solved as

$$f_{hg}(1) = 2 * N(0, 1, 1) = 0.484$$

for values that are at the threshold. Similarly, we normalize the values for use in the sigmoid function f_{sig} to obtain consistent confidences. By normalizing in this way, we can picture constraint confidence as a measure of the “number of thresholds away from optimal” a given value is, ensuring a consistent space to compute confidence in regardless of the magnitude of the threshold.

Relational (f_{sig})	Distance-from-Zero (f_{hg})	
above	acute-angle	obtuse-meet
below	acute-meet	parallel
contains	bisects	perpendicular
left-of	closer	positive-slope
right-of	coincident	same-height
	connected	same-size
	equal-angle	same-width
	horizontal	same-x
	intersects	same-y
	larger-size	slanted
	negative-slope	smaller-size
	obtuse-angle	vertical

Table 2. The 29 constraints implemented with our confidence system. The five relational constraints deal with positions of shapes in relation to one another. The rest of the constraints use f_{hg} to solve confidence.

DISCUSSION

We have implemented 29 constraints from the LADDER shape description language, listed in Table 2. Five of these constraints use the relational approach for solving confidence, and all deal with the positioning of shapes in relation to one another. We use f_{sig} rather than f_{hg} for these constraints because there is not a good point at which to define a “zero” to use the distance-from-zero approach. For the rest of the constraints, there is a point at which you can define a baseline zero, and f_{hg} works well for computing confidence.

FUTURE WORK

For future work, we would like to perform user studies to verify that our confidence values are assigned in a manner that aligns itself with human intuition and interpretation. We would also like to try learning appropriate confidence distributions from verification studies, as opposed to assuming a half-Gaussian or sigmoid function. Machine learning techniques could be used to learn appropriate thresholds from labeled data rather than relying on the empirical thresholds defined in LADDER.

ACKNOWLEDGEMENTS

This research is supported by the NSF IIS Creative IT Grant #0757557 Pilot: Let Your Notes Come Alive: The SkRUI Classroom Sketchbook.

REFERENCES

1. T. Hammond and R. Davis. LADDER: A language to describe drawing, display, and editing in sketch recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 461–467, 2003.
2. B. Paulson and T. Hammond. PaleoSketch: accurate primitive sketch recognition and beautification. In *Proceedings of the 13th International Conference on Intelligent User Interfaces (IUI)*, pages 1–10, New York, NY, USA, 2008. ACM.