Modern SignWriting

MSW v1.0.0-beta.1

compact and tractable
self-acting mathematical machines
for SignWriting

Stephen E Slevinski Jr
slevin@signpuddle.net

Abstract: Given that SignWriting is a 2-dimensional script, here are the specifications for an encoding model based on self-acting mathematical machines. This encoding model makes explicit those features which can be effectively and efficiently processed. Formal languages and regular expressions leverage these machines to solve fundamental problems.

Several generations of symbol sets, character encodings, and string representations have evolved with a natural writing script. Simple finite state machines explicitly illustrate several glass-box models of input, processing, and output.

Table of Contents

1.Stability....5
  1.A.Semantic Versioning....5
  1.B.Formal Words....5
  1.C.Normal Strings....5
  1.D.Terminology....6
    1.D.1.Grapheme, Symbol, and Glyph....6
    1.D.2.Cluster, Sign, and SignBox....6
    1.D.3.Viewpoints, Planes, and Perspectives....7
2.Historical Foundation....7
  2.A.Before SignWriting....7
  2.B.Handwritten SignWriting....7
  2.C.Computerized SignWriting....8
  2.D.Input Model....9
3.International Corpus....9
  3.A.Historical....9
  3.B.Modern....9
4.Mathematical Model....10
  4.A.Abstract Structure....10
  4.B.Proto Encoding....10
  4.C.Regular Expressions....11
    4.C.1.Regular Expression Basics....11
    4.C.2.Proto Encoding Patterns....11
  4.D.Variability....11
    4.D.1.Viewpoint Variability....12
    4.D.2.Symbol Variability....12
    4.D.3.Center Variability....12
    4.D.4.Order Variability....12
    4.D.5.Position Variability....12
5.Symbol Set....13
  5.A.Symbol Types (3)....13
    5.A.1.Writing Symbols....13
    5.A.2.Detailed Location Symbols....13
    5.A.3.Punctuation Symbols....13
  5.B.Categories (7)....14
  5.C.SymbolGroups (30)....15
  5.D.BaseSymbols (652)....16
  5.E.Symbols (37,811)....16
6.Symbol Encoding....17
  6.A.16-bit Symbol Code....18
  6.B.ASCII Symbol Key....18
  6.C.Encoding Data....18
    6.C.1.ID to Key....18
    6.C.2.Valid Fills and Rotations....19

6.C.3.BaseSymbol Variations....19
6.C.4.Symbol Sizes....19
6.D.Transformations....19
6.D.1.Reflection Transformation....20
6.D.2.Rotation Transformation....20
6.D.3.Flop Transformation....20
6.D.4.Variation Transformation....21
7.Script Layout....21
7.A.Symbol....21
7.B.SignBox....21
7.B.1.SignBox Space....22
7.B.2.Bounding Box....22
7.B.3.Center....22
7.C.Vertical Layout....23
7.D.Horizontal Layout....23
8.Text Encoding....23
8.A.Preprocessed Maximum Coordinate....24
8.B.Repertoire and Coded Character Set....24
8.C.Tokens and Patterns....25
8.D.Lite Markup....26
8.D.1.Structural Markers....26
8.D.2.Symbol Keys....26
8.D.3.Coordinates....26
9.Regular Searching Form....27
9.A.Lite Markup....27
9.A.1.Pattern Examples....27
9.A.2.Regular Expressions....28
9.B.Query String....28
9.B.1.Pattern Examples....28
9.B.2.Regular Expressions....32
9.C.Filter....32
9.D.Displacement....32
9.D.1.Displacement Grid....33
9.D.2.Displacement Example....33
10.Variant Display Forms....33
10.A.Raw....33
10.A.1.Pattern Examples....34
10.A.2.Regular Expressions....34
10.B.Expanded....34
10.B.1.Pattern Examples....35
10.B.2.Regular Expressions....35
10.C.Layout....36
10.C.1.Pattern Examples....36
10.C.2.Regular Expressions....37
10.D.Panel....37
10.D.1.Pattern Example....37
10.D.2.Regular Expressions....39
10.D.3.Transformational Parameters....39

11.Sorting Terminology....40
11.A.ISWA Sort Order....40
11.B.Explicit Prefix....40
11.C.Sorting Unordered SignBoxes....40
12.Spelling Normalization....41
12.A.Unordered and Inexact....41
12.B.Exact Spelling....41
12.C.Reflected Statistics....41
12.D.Symbol Subsets....42

1. Stability

SignWriting is the universal script for writing any sign language. Sign languages are human languages based on visual gestures. Sign languages have proven to have many of the uses and benefits of other voiced and written languages, as well as other uses and benefits unique to sign language alone.

Modern SignWriting is a faithful encoding of SignWriting that is stable by design. It is based on the glass-box idea where all necessary information is available to understand input, processing, and output. We can faithfully predict the results of any processing without having to use a computer. The algorithms have been optimized for flexibility, accuracy, simplicity, and speed.

1.A. Semantic Versioning

Modern SignWriting follows the rules for Semantic Versioning v2 (semver.org) and uses a version string to communicate compatibility using three numbers: major, minor, patch. The version string for this document is v1.0.0. It is the first major version. Any string written according to this specification will always be understood by a v1 implementation. The minor version number is second in the version string. It represents an improvement that is compatible for all versions with the same major version and a lower minor version number. This document is the foundation for major version l and will be compatible with all improvements and fixes for major version 1.

1.B. Formal Words

A formal word requires a sequential string of characters. Each encoding in this document is based on the idea of a formal word as an abstract structure. For SignWriting plain text, there are 3 types of words: signbox, term, or punctuation.

The signbox is the 2-dimensional representation of a sign that uses symbols from the ISWA 2010. This 2-dimensional order is represented as a list of symbols with 2 numbers per symbol for coordinate positioning. The 1-dimensional order of the string for a signbox is sequential and will resolve issues of overlap. Beyond that, the sequential order of a signbox should be considered meaningful to the user and should not be modified.

A term is a signbox with an added prefix as a sequential list of symbols. The majority of signs do not use the term prefix. Collation is possible with terms using a binary string comparison. A list of plain signboxes can be sorted using approximate string matching for terms and prepending the term prefix from the exact or closest match found for the signbox.

A punctuation is an isolated symbol used to structure sentences.

1.C. Normal Strings

A normal string is predictable and common. For SignWriting, normal strings do not exist outside of a community of writers. Different editors will often produce different strings for the same sign. Normalization is only possible based on an agreement within a community. Normal strings are only possible when written against a dictionary, otherwise the inherent variability of the mathematical model will produce strings that are approximately equal or equivalent, but rarely the same.

1.D. Terminology

1.D.1. Grapheme, Symbol, and Glyph

For SignWriting, a grapheme is a written mark that corresponds to a part of a meaningful visual gesture. Graphemes are the smallest semantic units capable of causing a contrast in meaning. Many graphemes of SignWriting are visually iconic. When written by hand, the variety and style of graphemes is potentially unlimited. Graphemes are somewhat abstract and do not have a concrete visual form. There are 3 types of graphemes: writing, detailed location, and punctuation.

The writing graphemes of SignWriting represent a visual conception: either hands, movement, dynamics, timing, head, face, trunk, or limb. The body concept is a combination of trunk and limb. The writing graphemes are used in 2 dimensional clusters to create one or more morphemes: the smallest semantically meaningful units in a language. The graphemes do not change size or shape when combined in a visual pattern. The graphemes can overlap and obscure graphemes underneath.

Detailed location graphemes are used individually or sequentially. They represent isolated analysis that is written outside the 2 dimensional cluster.

Punctuation graphemes are used when writing sentences. They are used individually, between 2 dimensional clusters.

For SignWriting, a symbol is equivalent to a grapheme, part of a grapheme, or several graphemes combined. Each symbol has a normative name, a definitive visual representation, and an international agreed upon meaning. The symbols of SignWriting are restricted in size and general shape more so than for other scripts because of 2 dimensional placement and the relations between the symbols. The symbols of SignWriting do not morph or rotate, but can often be changed from one symbol to another through several possible transformations. The specific size and shape of each symbol is designed to balance and complement the other symbols.

The symbols of the ISWA 2010 are extensive and specifically organized for written sign language and sign gestures. The symbol set does not include the specific symbols of DanceWriting or the general symbols of MovementWriting.

Glyphs are concrete representations of the individual symbols. A glyph is a graphic image that provides the appearance or form for a symbol. Most often, the term glyph will corresponds to an image in a font. The glyphs of SignWriting are restricted in size and shape with their corresponding symbols.

The glyphs have two aspects: positive and negative space. All glyphs have positive space: the outline or line of the glyph. Some glyphs have negative space, referred to as the fill of the glyph. When one glyph overlaps another glyph, the negative space of the top glyph will neutralize the positive space of the glyph underneath so that the line of the glyph underneath is partially obscured by the overlap.

1.D.2. Cluster, Sign, and SignBox

For SignWriting, a cluster is a 2 dimensional arrangement of graphemes, glyphs, or symbols. A 2 dimensional cluster of graphemes can be called a graphogram, which should not be confused with a "grapheme cluster" - a technical Unicode term. Likewise, a 2 dimensional cluster of glyphs can be called a glyphogram. A cluster can represents a sign of a sign language or a visual performance of a sign gesture. Many clusters are visually iconic.

A sign is an over loaded term. It can refer to either the visual gesture produced by a real person or the written 2 dimensional representation of that sign.

A signbox is a 2 dimensional mathematical construct with a visual image of variable height and width. The size of a signbox is defined by the cluster of symbols used within the signbox. The border of each signbox is a tight bounding box as the smallest possible rectangle that encloses the set of symbols. The center of a signbox is explicitly defined, either by algorithm or user choice.

1.D.3. Viewpoints, Planes, and Perspectives

Writing based on vision uses two viewpoints: receptive and expressive. The receptive viewpoint is based on the idea of receiving an image. For the receptive viewpoint, the right hand of a signer will be written on the left side of the image. When SignWriting is used for transcription, the receptive view is most often used. The related writing systems of DanceWriting and MovementWriting normally use the receptive viewpoint.

The expressive viewpoint is based on the idea of expressing a concept. For the expressive viewpoint, the right hand of a signer will be written on the right side of the image. When SignWriting is used for authorship, the expressive view is most often used.

The are two main writing planes: the front wall (Frontal Plane) and the floor (Transverse Plane). The choice of writing plane determines the shape of the symbols, such as the fill pattern for the hands or the tail for the movement arrows.

There are two perspectives: front and top. The front perspective is a straight on view of the signer. The top perspective is a top-down view of the signer. Usually, a sign will be written from a single perspective.

2. Historical Foundation

2.A. Before SignWriting

Valerie Sutton invented the DanceWriting notation in 1966.

2.B. Handwritten SignWriting

Valerie Sutton invented the SignWriting script in 1974. The SignWriting script was written exclusively by hand for 10 years. The script has evolved, spread around the world, and continues to be written on paper and chalkboard.

When written by hand, lines are drawn to form each grapheme. Different styles draw different types of lines: either for personal taste, speed, or quality. The main types of handwriting are formal, cursive, and shorthand. Formal handwriting, equivalent to block printing, includes defined lines for all graphemes, specific palm facings for hand shapes, and detailed arrow heads and tails. Cursive handwriting is more fluid and less detailed. Handwriting for personal use can omit palm facings, generalize arrows, and other liberties of personal consumption. Shorthand is a further reduction of detail, written for speed. Shorthand is a memory aid to a written record and should be rewritten soon after the notes were taken.

Understanding the ratios of size and shape for the graphemes improves hand writing.

2.C. Computerized SignWriting

When written with computers, the grapheme set must be limited and organized. A symbol is a reference to a grapheme rather than the grapheme itself. Each symbol has a unique identification.

In 1984, the first SignWriting prototype was created for the Apple IIe and Apple IIc. The application supported only a small subset of the SignWriting script.

In 1986, SignWriter was conceived by Richard Gleaves. Development started on the Apple IIe and Apple IIc. He pioneered the keyboard design for the SignWriting script. The resulting symbolset was limited due to the 128KB memory limit.

Richard Gleaves worked closely with Valerie Sutton and continued the development of SignWriter DOS to expanded the symbolset and improve the editor, resulting in the SSS-95 and a robust typing model.

The SSS-99 was created for SignWriter Java. The revamped symbolset was created without the limitations imposed upon the SSS-95.

The SSS-2002 reorganized the structure of the symbolset imposing a multi level hierarchy with the modern symbol ID. The SSS-2002 was the first symbolset used in the SignBank 2002 application by Todd Duell.

The SSS-2004 was created after reaching widespread international use. The SSS-2004 was the first symbolset used in the SignPuddle application by Steve Slevinski. This symbolset was expanded to include international MovementWriting concepts and became known as the International MovementWriting Alphabet.

The International SignWriting Alphabet 2008 was a major refactoring of the IMWA concept by eliminating the general MovementWriting symbols and focusing on the SignWriting script. The ISWA 2008 was the first symbolset released under the Open Font License.

The International SignWriting Alphabet 2010 was a further refinement of the symbolset to incorporate additional current best practices as deep in the standard as possible. The ISWA 2010 is the result of over 35 years of an inventor working with writers from around the world, and over 25 years of an inventor working with computers and programmers. The design balances complexity, efficiency, and usability. The ISWA 2010 defines 7 categories, 30 groups, and 652 bases. Understanding the basic concepts of the ISWA 2010 allows easy access to the graphemes without memorizing the minute details.

Since major standardization efforts started in 2008, there have been several disruptive but necessary changes in the data formats. With the formal specifications of Modern SignWriting, this type of disruption is no longer allowed. Signs for version 1 of Modern SignWriting will not change and will always be compatible with applications that support MSW v1.

The formal specifications for Modern SignWriting are a continuation of the past; based on long standing historical usage.

2.D. Input Model

In 1986, Richard Gleaves pioneered the keyboard model for SignWriting. With SignWriter Dos, it was possible to type complete SignWriting sentences. New symbols required 2 key presses: first for the SymbolGroup and then for the BaseSymbol. Many selected symbol could be transformed by rotation, mirror, flop, or variation using additional keystrokes. Within a sign, the cursor is based on a selected symbol and 8 rotational directions. A new symbol is added to a sign based on the selected symbol position and the placement of the cursor around that symbol. This model is extremely powerful, but has a learning curve. The design is still valid for the modern symbol set, but it is not currently implemented in software.

In 2004, Stephen E Slevinski Jr created the first drag & drop editor for SignWriting. This model incorporated symbol hierarchy for symbol selection, but did not use the keyboard or the circular cursor. This model has proven to be effective and easy to learn.

3. International Corpus

From education to research, from entertainment to religion, SignWriting is useful because people are writing.

3.A. Historical

Since it's beginnings in 1974, SignWriting has always been a practical script that has been used by real people writing real text. What started in one country has spread around the world.

Dating from 1986 till today, there is an extensive set of documents for SignWriter Dos (and other editors) from multiple countries around the world.

3.B. Modern

Since 2004, an international corpus has been building. It has undergone several symbol set updates. It has grown from comma delimited data, to XML, and finally to formal words. Each conversion has enabled the international corpus to grow and evolve. Along the way, data from previous historical formats has been reentered into the corpus.

In 2012, the international corpus has hundreds of thousands of signs from dozens of different languages.

The quality of the corpus is inconsistent. There are many spelling mistakes and older forms of writing. This body of work isn't the end, but the start of something more. As more people learn to write and develop their own style, the quality of the corpus will naturally improve.

4. Mathematical Model

If you use information correctly, the model will behave correctly. Mathematics is not subjective, but deterministic. Access to the subsystem internals make it easier to understand. This provides knowledge independent of experience.

Through trial and evaluation, the model has been successively refactored to reduce the complexity and the computation cost of the implementations. Several generations of empirical data has been created by applying the model to real world situations. The consequences of the model has been observable by the senses. Problem areas and shortcomings have been resolved to create a robust solution.

4.A. Abstract Structure

The abstract structure of the formal strings is governed by a set of laws. The grammar of the formal strings are based on syntactic rules that define their internal structure.

4.B. Proto Encoding

$C \subset L(G)$
* C = the corpus of international text
* $\subset$ = the subset of
* L = the modern SignWriting language
* G = the formal grammar of modern SignWriting

$G = (N,\Sigma,P,S)$
* N = the set of non-terminal tokens
* Σ = the set of terminal tokens
* P = the set of production rules
* S = the start token

$N = \{S,T,A,B\}$
* S = start token
* T = term
* A = ordered prefix
* B = signbox

$\Sigma = \{a,b,l,m,r,w,s,p,n\}$
* a = prefix marker
* b = signbox marker
* l = left lane marker
* m = middle lane marker
* r = right lane marker
* w = writing symbol
* s = sequential symbol
* p = punctuation symbol
* n = number token

Word production rules
* P1: $S \rightarrow T$
* P2: $S \rightarrow B$
* P3: $S \rightarrow p$
* P4: $T \rightarrow AB$
* P5: $A \rightarrow Aw$
* P6: $A \rightarrow As$
* P7: $A \rightarrow aw$
* P8: $A \rightarrow as$
* P9: $B \rightarrow Bwnn$
* P10: $B \rightarrow b$
* P11: $B \rightarrow l$
* P12: $B \rightarrow m$
* P13: $B \rightarrow r$

Sentence production rules
* P14: $S \rightarrow ST$
* P15: $S \rightarrow SB$
* P16: $S \rightarrow Sp$

4.C. Regular Expressions

A regular expression is used to examine text and identify strings that match a stated pattern. A regular expression is written in a concise and flexible formal language. It contains literals and metacharacters. A literal is any character that we literally want to find in the string. A metacharacter is a special character with a unique meaning that is not literally in the string, but represents a type of search pattern.

4.C.1. Regular Expression Basics

Characters	Description	Example
*	Match a literal 0 or more times	ABC* matches AB, ABC, ABCC, ...
+	Match a literal 1 or more times	ABC+ matches ABC, ABCC, ABCCC, ...
?	Match a literal 0 or 1 times	ABC? matches AB or ABC
{#}	Match a literal "#" times	AB{2} matches ABB
[ ]	Match any single literal from a list	[ABC] matches A, B, or C
[-]	Match any single literal in a range	[A-C] matches A, B, or C
( )	Creates a group for matching	A(BC)+ matches ABC, ABCBC, ABCBCBC, ...
(|)	Matches one of several alternatives	(AB|BC|CD) will match AB, BC, or CD

4.C.2. Proto Encoding Patterns

Pattern	Description
wnn	A single writing symbol positioned with 2 numbers
(wnn)*	Zero or more writing symbols, each positioned with 2 numbers
b(wnn)*	A signbox used for horizontal writing
[lmr](wnn)*	A signbox in a lane used for vertical writing
a[ws]+	A prefix with one or more writing or sequential symbols
(a[ws]+)?	An optional prefix with one or more writing or sequential symbols
p	A punctuation
(((a[ws]+)?b(wnn)*)|p)+	A sign text for horizontal writing as a string of signboxes (with optional prefixes) and punctuation
(((a[ws]+)?[lmr](wnn)*)|p)+	A sign text for vertical writing as a string of signboxes in lanes (with optional prefixes) and punctuation

4.D. Variability

The symbols and their meaning are based on an international agreement that does not allow for variability. Everyone should understand the individual symbols to mean the same thing regardless of language or writing style.

Beyond the individual symbols, there are two primary spelling rules for writing a sign as a 2 dimension cluster of symbols: 1) write the position of contact and 2) every sign has a center.

Beyond the individual symbols, there exists a vast amount of variability in how people write: horizontal or vertical, minimalist or exacting detail, few heads or lots of heads, stick figures or detached hands.

Even within the same style, there exists five types of variability for Modern SignWriting.

4.D.1. Viewpoint Variability

Some signs can be written from the front perspective (straight-on view of the signer) or from the top perspective (top-down view of the signer). Although these written signs represent the same sign, they will not use the same string.

4.D.2. Symbol Variability

Symbol choice is sometimes subjective. A sign can be written with more, less, or different details. The choice of detail will affect the choice of symbol and hence the string.

4.D.3. Center Variability

Every 2 dimensional cluster of symbols has a center by definition. This center is defined as the origin of the signbox space (0,0). The center of a signbox is important for layout.

All signs that do not contain a head or trunk symbol will center the same regardless of writing direction. Horizontal writing will center on the head symbols only. Vertical writing will center on both the head and the trunk symbols.

A hybrid centering technique can solve both writing directions at the same time. The vertical center of a signbox should be based on the heads alone. The horizontal center of a signbox should be based on heads and trunks.

A custom center allows the user to specifically set the center of a sign and override any predetermined value.

4.D.4. Order Variability

The order of symbols in a signbox string is only meaningful for issues of overlap when one symbol is positioned on top of another and the negative space of the top symbol obscures part of the positive space of a lower symbol. Otherwise, the order of the symbols is irrelevant to the visual representation of a string. The relationship is surjective with several strings mapping to a single visual image. Order variability can be exploited to resolve issues of ambiguity. Two different signs with the same visual appearance but different meanings can have different spellings based on order variability.

4.D.5. Position Variability

Within the formal strings, the precise position of each symbol is user defined. Unless written against an accepted dictionary, a user has the ability to fine tune the position of each symbol. For a single symbol, there is no positional variability. Two symbols will have dozens of approximate relations.

More than two, and the number of approximate relations increases exponentially.

5. Symbol Set

The ISWA 2010 is a mathematical alphabet. It is the end result of 8 years of standardization efforts with writers from around the world. Serious standardization efforts were started in 2004 with the International MovementWriting Alphabet. These efforts refocused with the ISWA 2008. These efforts were realized for the symbol set with the ISWA 2010.

Valerie Sutton created and named each symbol of the ISWA 2010 with a definitive image and a normative ID. Steve Slevinski reflected Valerie's design though visual and mathematical models.

The normative IDs structure the symbols as a hierarchy with 6-degree of features. A symbol ID is a sequence of six formatted numbers of increasing detail. The first dashed number defines the Category (11). The first two dashed numbers define the SymbolGroup (11-22). The first four dashed numbers define a BaseSymbol (11-22-333-44). The fifth number represents the fill (55). The sixth number represents the rotation (66). A symbol ID is a combination of BaseSymbol ID with a fill value and a rotation value. A symbol ID has the format "nn-nn-nnn-nn-nn-nn", where each "n" is a digit from 0 to 9.

The ISWA 2010 is fully documented and available under the Open Font License with a file based font, either PNG or SVG, either individual files or XML.

5.A. Symbol Types (3)

5.A.1. Writing Symbols

The writing symbols of SignWriting represent a visual conception: either hands, movement, dynamics, timing, head, face, trunk, or limb. The body concept is a combination of trunk and limb.
The writing symbols are used in 2 dimensional clusters to create one or more morphemes: the smallest semantically meaningful units in a language. The symbols do not change size or shape when combined in a visual pattern. The symbols can overlap and obscure symbols underneath.

5.A.2. Detailed Location Symbols

Detailed location symbols are used individually or sequentially. They represent isolated analysis that is written outside the 2 dimensional cluster.

5.A.3. Punctuation Symbols

Punctuation symbols are used when writing sentences. They are used individually, between clusters.

5.B. Categories (7)

Category 1: Hands

Handshapes from over 40 Sign Languages are placed in 10 groups based on the numbers 1-10 in American Sign Language.

Category 2: Movement

Contact symbols, small finger movements, straight arrows, curved arrows and circles are placed into 10 groups based on planes: The Front Wall Plane includes movement that is “parallel to the front wall” and the Floor Plane includes movement that is “parallel to the floor”.

Category 3: Dynamics

Dynamics Symbols are used mostly with Movement Symbols and Punctuation Symbols, to give the "feeling" or "tempo" to movement. They also provide emphasis on a movement or expression, and combined with Puncuation Symbols become the equivalent to Exclamation Points. The Tension Symbol, combined with Contact Symbols, provides the feeling of 'pressure", and combined with facial expressions can place emphasis or added feeling to an expression. Timing symbols are used to show alternating or simultaneous movement.

Category 4: Head & Faces

Starting with the head and then from the top of the face and moving down. Group 22 includes head movement and views of the head. Groups 23-26 include detailed facial expressions and movement of parts of the face and neck.

Category 5: Body

Torso movement, shoulders, hips, and the limbs are used in Sign Languages as a part of grammar, especially when describing conversations between people, called Role Shifting, or making spatial comparisons between items on the left and items on the right. This category is important when writing sign language storytelling and poetry. All sign languages have some signs that point below the hips, or touch the torso, or hunch the shoulders, or touch the arms and wrists.

Category 6: Detailed Location

Detailed Location symbols are used in the SignSpelling Sequence and not in the Spatial SignSpelling. May be useful for sorting large dictionaries, refining animation, simplifying translation between scripts and notation systems, and for detailed analysis of location sometimes needed in linguistic research.

Category 7: Punctuation

Punctuation Symbols are used when writing complete sentences or documents in SignWriting. The Punctuation Symbols do not look like the symbols for punctuation in English, but they do have similar meanings. SignWriting punctuation symbols include a period, comma, colon, semicolon, exclamation point and so forth.

5.C. SymbolGroups (30)

There are 30 SymbolGroups. The first 2 dashed numbers in the symbol ID identify the group. The 30 groups can be divided into 3 sets of 10. The first ten are hands, category 1. The second ten are movements, category 2. The third ten are categories 3 thru 7. In order, 1 group for the Dynamics & Timing category, 1 for Head, 4 for Face, 1 for Trunk, 1 for Limb, 1 for Detailed Location, and 1 for Punctuation.

First Set	Second Set	Third Set
01-01 Index	02-01 Contact	03-01 Dynamics & Timing
01-02 Index Middle	02-02 Finger Movement	04-01 Head
01-03 Index Middle Thumb	02-03 Straight Wall Plane	04-02 Brow Eyes Eyegaze
01-04 Four Fingers	02-04 Straight Diagonal Plane	04-03 Cheeks Ears Nose Breath
01-05 Five Fingers	02-05 Straight Floor Plane	04-04 Mouth Lips
01-06 Baby Finger	02-06 Curves Parallel Wall Plane	04-05 Tongue Teeth Chin Neck
01-07 Ring Finger	02-07 Curves Hit Wall Plane	05-01 Trunk
01-08 Middle Finger	02-08 Curves Hit Floor Plane	05-02 Limbs
01-09 Index Thumb	02-09 Curves Parallel Floor Plane	06-01 Detailed Location
01-10 Thumb	02-10 Circles	07-01 Punctuation

5.D. BaseSymbols (652)

There are 652 BaseSymbols. The first 4 dashed numbers of a symbol ID identify the base. The 652 bases are divided between the 30 groups. For each group, there are less than 60 bases. The bases are often displayed in columns of 10.The details of the individual BaseSymbols is beyond the scope of this document. Please refer to the ISWA 2010 HTML Reference or any of Valerie Sutton's ISWA 2010 documents.

5.E. Symbols (37,811)

The symbols are variably sized; they do not share a common height or width. The symbols are static; they do not rotate or morph. Each symbol has a unique name as the symbol ID.

The symbol ID is a six part number formatted for zero padding with a mask of "xx-xx-xxx-xx-xx-xx-xx". The six individual numbers of the symbol ID listed by order are: category, group, base, variation, fill, rotation.

The first number by itself identifies the Category, together with the second number identifies the SymbolGroup ID.

The third number identifies the BaseSymbol place, usually with an "01" for the fourth number: variation. When more than one BaseSymbols share the first three numbers, the fourth number will order these BaseSymbols with variation numbers starting from 1.

The fifth and six numbers uniquely define an individual symbol. They are used to place symbols on a 6 by 16 palette. Each BaseSymbol has a unique symbol palette that lists the symbols on a 6 by 16 grid with valid columns and valid rows. Any cell on a valid column and a valid row is valid and must identify a symbol. Any cell on an invalid column or an invalid row does not represent a symbol and is invalid.

The fill modifier can best be understood through the palm facing of the hand graphemes. The palm facing is based on planes. The SignWriting script uses two planes: the Front Wall (Frontal Plane) and the Floor (Transverse Plane). There are 6 palm facings. The first three palm facings are parallel with the Front Wall. The second three palm facings are parallel with the Floor. The reader can view the signer from different viewpoints (expressive or receptive) and can view the hands from different perspectives (front or top), but no matter what the viewpoint or perspective, the first three Fills represent the palm facing parallel to the Front Wall and the second three Fills represent the palm facing parallel to the Floor.

Fill	Indicator	Meaning
01	grapheme with white palm	reader sees palm of hand parallel Front Wall
02	grapheme with half black palm	reader sees side of hand parallel Front Wall
03	grapheme with black palm	reader sees back of hand parallel Front Wall
04	grapheme with white palm and broken line	reader sees palm of hand parallel Floor
05	grapheme with half black palm and broken line	reader sees side of hand parallel Floor
06	grapheme with black palm and broken line	reader sees palm of hand parallel Floor

The fill modifier is redefined for the movement arrows of category 2.

Fill	Indicator	Meaning
01	a grapheme with a black arrow head	movement of the right hand
02	a grapheme with a white arrow head	movement of the left hand
03	a grapheme with a thin, unconnected arrow head	spatial overlapping of movement arrows for the left and right hands when they move as a unit
04	Irregular arrow stems	building blocks for complex movement

The rest of the other bases use a fill modifier for grouping and visual organization that is meaningful only for a particular base symbol or small set.

The rotation modifier can best be understood through the hand symbols. The first 8 rotations progress 45 degrees counter clockwise. The last 8 rotations are a mirror of the first 8 and progress 45 degrees clockwise. Zero (0) degrees is understood to point to the top of the grapheme.

Rotation	Direction	Degrees from top
01	Counter Clockwise	0
02	Counter Clockwise	45
03	Counter Clockwise	90
04	Counter Clockwise	135
05	Counter Clockwise	180
06	Counter Clockwise	225
07	Counter Clockwise	270
08	Counter Clockwise	315
09	Clockwise	0
10	Clockwise	45
11	Clockwise	90
12	Clockwise	135
13	Clockwise	180
14	Clockwise	225
15	Clockwise	270
16	Clockwise	315

6. Symbol Encoding

Every symbol of the ISWA 2010 has a normative name as the symbol ID, which is 18 characters long. These names are overly verbose and are not used directly for encoding.

6.A. 16-bit Symbol Code

A 16-bit code is an integer between 0 and 65,535. This type of value is perfect for a primary key for database lookup or other integer index. Through a simple formula, each symbol has a unique 16-bit codepoint in the x-ISWA-2010 coded character set.

There are 652 BaseSymbols, numbered from 1 to 652. Each BaseSymbol can be visualized on a grid of 6 columns and 16 rows: for the 6 fills and 16 rotations. Each symbol can be identified by 3 values of BaseSymbol, column and row.

If we start on the first BaseSymbol grid, give the first symbol a code value of 1, incremented down the first column, continue to the next column, and continue through the remaining BaseSymbols; we end up with numbers from 1 to 62952.

Given any symbol with:

BaseSymbol number = n
Fill = f
Rotation = r

code = (n-1)*96 + (f-1)*16 + r

6.B. ASCII Symbol Key

The symbol key is a formatted string, 6 ASCII characters long, that identifies a symbol. The symbol key makes explicit features of the symbol otherwise only implicit in the symbol code.

The first character of the symbol key is always "S". This aids in parsing and human readability. The last 5 characters represent the symbol encoding in hexadecimal: base 16.

Base 10 is called decimal and is the most widely used numbering system. This uses 10 digits from 0 through 9. Hexadecimal numbers use base 16. This uses 16 digits, from 0 through 9 and from “a” through “f”.

Given any symbol with:

BaseSymbol number = n
Fill = f
Rotation = r
dechex = function that converts a decimal number to a hexadecimal number.

key = S . dechex(n+255) . dechex(f-1) . dechex(r-1)

6.C. Encoding Data

Most operations of layout and searching can be completed without requiring additional information outside of the sign text strings. There are 4 special cases that require access to outside information.

6.C.1. ID to Key

The symbol ID does not have an automatic conversion to symbol key without additional information. Every one of the 652 BaseSymbols has a unique mapping from BaseSymbol ID to BaseSymbol number. The data file includes 652 entries of the format “xx-xx-xxx-xx equals BS #”

6.C.2. Valid Fills and Rotations

Each BaseSymbol has 2 integer values that define validity for fills and rotations. The Binary Fills value is a 6-bit number from 1 to 63. This value defines which fills are valid for the BaseSymbol. The Binary Rotations value is a 16-bit number from 1 to 65,535. This value defines which rotations are valid for the BaseSymbol. The data file includes 652 entries that map each BaseSymbol to 2 different values: one for the Binary Fills value and one for the Binary Rotations value.

Given any symbol defined as a Symbol Key:

> *Symbol Key = “S” . H1 . H2 . H3 . H4 . H5*
> *Fill Power = 2 ^ hexdec(H4)*
> *Rotation Power = 2 ^ hexdec(H5)*
> *Binary Fills = look up value for BaseSymbol*
> *Binary Rotations = look up value for BaseSymbol*

The symbol key represents a valid symbol if both:

> **(Fill Power BITWISE AND Binary Fills) > 0**
> **(Rotation Power BITWISE AND Binary Rotations) > 0**

6.C.3. BaseSymbol Variations

Most symbol IDs use a variation number of 1. This is the fourth number of the symbol ID. A few symbols utilize the variation number to create a relationship between different BaseSymbols: such as arrows of different length. When BaseSymbols are related through the variation number, the first 3 numbers of the symbol ID will be the same, and the variation number will start at 1 for the first BaseSymbol and increase by 1 for each subsequent related BaseSymbol. The data file includes 652 entries that list the number of variations for each BaseSymbol. BaseSymbols that are related by variation will have the same number of variations. The symbol ID must be referenced to determine the variation number for any symbol.

6.C.4. Symbol Sizes

Every symbol of the ISWA 2010 has a specific width and height. These values are needed for string preprocessing that includes centering or sizing. These values can be determined several different ways.

Image Analysis

The PNG or SVG image can be analyzed for each symbol to determine the width and height.

ID or Key Lookup

Using the 16-bit symbol code or symbol key, the width and height can be accessed from a database or other data storage.

6.D. Transformations

Many symbols can be transformed by rotation, mirror, flop, or variation.

Given any symbol defined as a Symbol Key:

```
    Symbol Key = "S" . H1 . H2 . H3 . H4 . H5
    Fill = hexdec(H4) + 1
    Rotation = hexdec(H5) + 1;
```

New symbol keys should always be checked for validity. If the symbol key is not valid, repeat the transformation.

6.D.1. Reflection Transformation

```
If Binary Rotations > 255 there are more than 8 rotations:
  If Rotation > 8
    Rotation = Rotation - 8
  Else if Rotation <= 8
    Rotation = Rotation + 8

If Binary Rotations < 256 there are 8 or less rotations
  Rotation 1 mirrors 1
  Rotation 2 mirrors 8
  Rotation 3 mirrors 7
  Rotation 4 mirrors 6
  Rotation 5 mirrors 5

Symbol Key = "S" . H1 . H2 . H3 . dechex(Rotation - 1) . H5
```

6.D.2. Rotation Transformation

```
Rotations 1 thru 8 loop
Rotations 9 thru 16 loop

If Rotation < 9
  Clockwise
    Rotation = Rotation - 1
  Counter Clockwise
    Rotation = Rotation + 1

If Rotation > 8
  Clockwise
    Rotation = Rotation + 1
  Counter Clockwise
    Rotation = Rotation - 1

Symbol Key = "S" . H1 . H2 . H3 . dechex(Rotation - 1) . H5
```

6.D.3. Flop Transformation

```
Fills 1 thru 6 loop
Fill = Fill + 1

Symbol Key = "S" . H1 . H2 . H3 . H4 . dechex(Fill - 1)
```

6.D.4. Variation Transformation

Each BaseSymbol has 2 numbers that deal with variations: the first is the specific variation number in the Symbol ID and the second is the general number of variations found in a data file.

Given any symbol:

Variation number = symbol specific value from the symbol ID
Number of variations = BaseSymbol value from a data file

The variation transformation is available if the number of variations is greater than 1:

```
If (variation number < number of variations)
   Variation number = variation number + 1
Else if (variation number = number of variations)
   Variation number = 1
```

7. Script Layout

Sign languages are fundamentally different than spoken language, just as vision is fundamentally different that sound. SignWriting is an unusual script because the writer chooses symbols from a vast symbol set and the writer chooses the 2-dimensional position for each symbol.

The layout of the SignWriting script is based on clusters of symbols and individual punctuation. Each cluster represents a sign of a sign language or a visual performance of a sign gesture. Punctuation divides the clusters into sentences. Most commonly, the clusters are arranged vertically, from top to bottom aligned on their centers. Other times, the signs are arranged horizontally from left to right.

7.A. Symbol

Each symbol has a specific shape and size. The size of each symbol is static with an explicit width and height for each. The shape of each symbol is meaningful but informal.

7.B. SignBox

A SignBox is a 2 dimensional arrangement of writing symbols. The 2 dimensional nature of the SignBox does not have a normative string representation. Consider the following arrangement.

```
       |
       |  B
     C |
-------+------
       |
       |   A
```

The previous arrangement is not ABC, ACB, BAC, BCA, CAB, or CBA. There is no definitive first, second, or third element. For an accurate representation, the following arrangement must specify the 2 dimensional position of each element: C (-2,-1), B (3,-2), A (5,2). The sequential order of the elements in not meaningful, except in cases of overlap.

7.B.1. SignBox Space

SignBox space is 2 dimensional with an X and Y axis. Each axis has a range from -250 to 249. The center of SignBox space is the coordinate (0,0).

```
           Y Axis
             | -250
             |
             |
   X Axis    |
     --------+--------
     -250    |     +249
             |
             |
             | +249
```

7.B.2. Bounding Box

A tight bounding box is defined as the smallest possible rectangle that encloses a set of symbols. The size of a sign is defined by a tight bounding box. Each bounding box has 3 points of interest: a minimum, a maximum, and a center. The minimum coordinate is explicitly defined by the position of the symbols. Since symbols are positioned by the top-left coordinate, the minimum coordinate is a combination of the minimum X coordinate and the minimum Y coordinate. These values can come from 2 different symbols.

The maximum coordinate is the bottom-right of the bounding box. The maximum coordinate is not explicitly defined by the position of the symbols. The maximum coordinate requires the respective width and height of each symbol. For each symbol, the width and height are added to the placement coordinate. The maximum X and Y from this new list is the maximum coordinate for the bounding box.

The maximum coordinate can be preprocessed, processable, or undefined. If the value is preprocessed, the maximum coordinate is explicitly defined for the SignBox. If the value is processable, the symbol list includes both the symbol placement and the symbol size. If the value is undefined, the size of the symbols must be looked up from some outside data source.

The center coordinate of a bounding box is some point within the bounding box, but no necessarily the midpoint. If a sign is centered, the center is the coordinate (0,0). This means that a centered sign will have a negative minimum coordinate and a positive maximum coordinate.

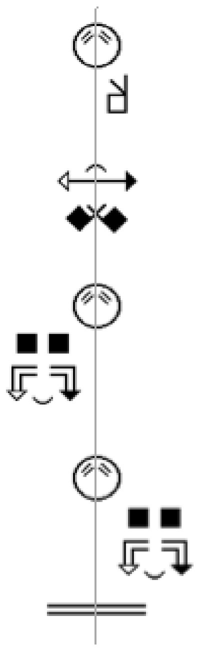
7.B.3. Center

The center of a sign is important for layout. The center of a sign is dependent upon the types of symbols used in the SignBox. If a SignBox does not contain any head or trunk symbols, then the center of a SignBox is the midpoint of the tight bounding box around all of the symbols.

If a SignBox contains head symbols, then the vertical center is the midpoint of the tight bounding box around the head symbols. If a SignBox contains head or trunk symbols, then the horizontal center is the midpoint of the tight bounding box around the head and trunk symbols.

A custom center would allow a user to specifically set the center of a sign and override any predetermined value.

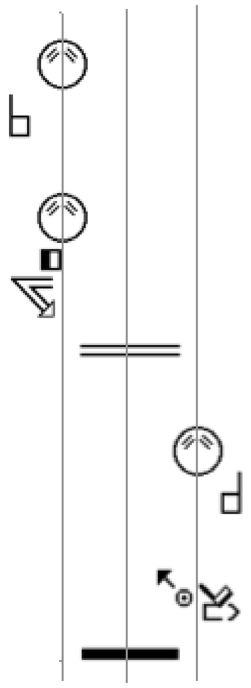
7.C. Vertical Layout



When written vertically, SignWriting can use 3 different lanes: left, middle, and right. The middle lane is the default lane and punctuation is always used in the middle lane. No matter the lane, the center of a sign is aligned with the center of the lane.

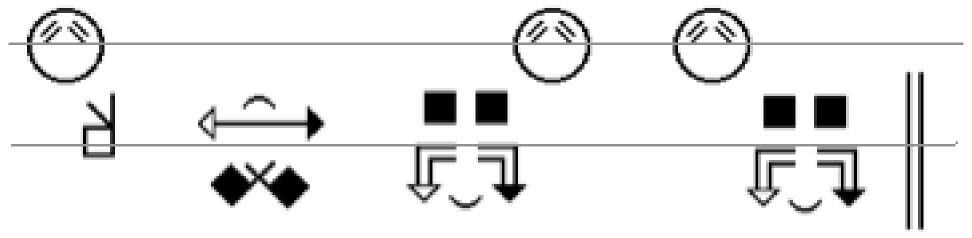
For body weight shifts to one side or the other, the center of the cluster is aligned with a fixed horizontal offset from the middle lane into either the left or right lane.

The left and right lanes are used to represent body weight shifts and are represented by a horizontal offset from the middle lane. Body weight shifts are important to the grammar of sign languages, used for two different grammatical aspects: 1) role shifting during sign language storytelling, and 2) spatial comparisons of two items under discussion. One "role" or "item" is placed on the right side of the body (right lane), and the other on the left side of the body (left lane), and the weight shifts back and forth between the two, with the narrator in the middle (middle lane).



7.D. Horizontal Layout

When written horizontally, SignWriting uses 2 lanes: a middle lane and an upper head lane. Signs with head symbols are centered in the upper head lane. All other signs and punctuation are centered in the middle lane.



8. Text Encoding

A text encoding is machine readable. It should make explicit features of the text that can be automatically analyzed. Text encoding exists solely to support the various processes that act upon text and should therefore make it possible to resolve hard problems in a principled and efficient manner.

8.A. Preprocessed Maximum Coordinate

The maximum coordinate is the bottom-right coordinate of the signbox. While the minimum coordinate can be determined from the symbol positioning, the maximum coordinate requires the addition of the respective width and height for each symbol. At the time of data entry, the maximum coordinate can be preprocessed and explicitly defined for the SignBox. This preprocessed value makes it possible to perform script layout without accessing an outside data source. When present, the maximum coordinate is added to the signbox definition directly after the signbox or lane marker.

8.B. Repertoire and Coded Character Set

The repertoire of the Modern SignWriting encoding consistes of 5 markers, 6 columns, 16 rows, 652 grid pages, and 500 numbers. These abstract characters can be mapped to codepoints in the 12-bit coded character set of x-BSW-12 or to codepoints in the Unicode private use area.

The BSW string has been designed for an easy conversion to Unicode. Each x-BSW-12 codepoint can be shifted by the hex value of FD700 to determine the Unicode PUA codepoint.

Name	Token	BSW Codepoint(s)	Unicode PUA
Sequence Marker	A	B+100	U+FD800
SignBox Marker	B	B+101	U+FD801
Left Lane Marker	L	B+102	U+FD802
Middle Lane Marker	M	B+103	U+FD803
Right Lane Marker	R	B+104	U+FD804
Columns 1 thru 6 (fills)	i	B+110 – B+115	U+FD810 – U+FD815
Rows 1 thru 16 (rotations)	o	B+120 – B+12F	U+FD820 – U+FD82F
SignWriting Grid Pages (base symbols)	w, s, or P	B+130 – B+3BB	U+FD830 – U+FDABB
Negative Numbers: -250 thru -1	n	B+706 – B+7FF	U+FDE06 – U+FDEFF
Positive Numbers: 0 thru 249	n	B+800 – B+8F9	U+FDF00 – U+FDFF9

Token	Type	Section	Symbol Key	BSW Codepoint(s)	Unicode PUA
w	Writing		S100 - S37e	B+130 - B+3ae	U+FD830 - U+FDAAE
w	Writing	Hands	S100 - S204	B+130 - B+234	U+FD830 - U+F934
w	Writing	Movement	S205 - S2f6	B+235 - B+326	U+FD935 - U+FDA26
w	Writing	Dynamics	S2f7 - S2fa	B+327 - B+32a	U+FDA27 - U+FDA2A
w	Writing	Timing	S2fb - S2fe	B+32b - B+32e	U+FDA2B - U+FDA2E
w	Writing	Head	S2ff - S309	B+32f - B+339	U+FDA2F - U+FDA39
w	Writing	Face	S30a - S36c	B+33a - B+39c	U+FDA3A - U+FDA9C
w	Writing	Trunk	S36d - S375	B+39d - B+3a5	U+FDA9D - U+FDAA5

w	Writing	Limb	S376 - S37e	B+3a6 - B+3ae	U+FDAA6 - U+FDAAE
s	Detailed Location		S37f - S386	B+3af - B+3b6	U+FDAAF - U+FDAB6
P	Punctuation		S387 - S38b	B+3b7 - B+3bb	U+FDAB7 - U+FDABB

8.C. Tokens and Patterns

There are 11 tokens used with Mondern SignWriting. They can be grouped in 4 layers: the 5 structural makers (A, B, L, M, R), the 3 ranges of base symbols (w, s, P), the 2 modifiers (i, o), and the numbers (n).

A string of Modern SignWriting characters (either BSW or Unicode PUA) can be visualized as tokens rather than characters. A tokenized view replaces each character with 1 of the 11 token values. The use of tokens clarifies structures and simplifies regular expressions. See section 4.C.1 on page 11 for Regular Expression Basics.

Pattern	Description
wio	a writing symbol as 3 characters of writing base, fill modifier and rotation modifier
nn	coordinate with X and Y values as 2 numbers
wionn	a spatial symbol as 5 characters with 3 characters of a writing symbol and 2 characters for coordinates for top left placement
(wionn)*	zero or more spatial symbols
Bnn(wionn)*	a signbox with a preprocessed maximum coordinate and a list of spatial symbols used for horizontal writing
[LMR]	a lane marker: either left, middle or right.
[LMR]nn(wionn)*	a signbox in either the left, middle, or right lane with a preprocessed maximum coordinate and a list of spatial symbols used for vertical writing
[ws]	a writing base symbol or a detailed location base symbol
[ws]io	a writing symbol or a detailed location symbol
([ws]io)+	one or more writing symbols and/or detailed location symbols
(A([ws]io)+)?	an optional prefix as a prefix marker followed by one or more writing symbols and/or detailed location symbols
Pio	a punctuation symbol as a punctuation base symbol with a fill modifier and a rotation modifier
(((A([ws]io)+)?Bnn(wionn)*)|Pio)+	a sign text for horizontal writing as a string of signboxes (with optional prefixes) and punctuation
(((A([ws]io)+)?[LMR]nn(wionn)*)|	a sign text for vertical writing as a string of signboxes in

Pio)+	lanes (with optional prefixes) and punctuation

8.D. Lite Markup

Instead of binary character data or full XML, it has proven to be beneficial to use a human readable lite markup of ASCII words separated by spaces. Each word represents either a signbox or a punctuation. The lite markup has the advantage of a small size without requiring special Unicode or XML functions. Simple regular expressions can quickly and efficiently process the lite markup.

8.D.1. Structural Markers

In the lite markup, the structural markers use the token values instead of BSW or Unicode PUA.

Lite markup	Description
A	Sequence Marker
B	SignBox Marker
L	Left Lane Marker
M	Middle Lane Marker
R	Right Lane Marker

8.D.2. Symbol Keys

In the lite markup, symbols are referenced by symbol keys: the letter 'S' followed by 5 hexadecimal values.

8.D.3. Coordinates

In the lite markup, there are 2 types of coordinates: regular fixed-width coordinates and irregular variable-width coordinates. Both types of coordinates contain 2 numbers separated by the letter 'x'.

Regular Coordinates

In the lite markup, regular coordinates are always 7 ASCII characters long: 3 digits followed by the letter 'x' followed by 3 more digits. The numbers range from 250 to 749, with 500 being the center point as zero. So for regular coordinates, the string “250” is equal to the number value of -250 and “749” is equal to the number value of 249.

The loose definition of regular coordinates matches numbers with 3 digits without specifying the number range. It has a regular expression of /[0-9]{3}x[0-9]{3}/.

The strict definition of regular coordinates only matches numbers in the range from 250 to 749. It has a more verbose regular expression of /(2[5-9][0-9]|[3-6][0-9]{2}|7[0-4][0-9])x(249|2[5-9][0-9]|[3-6][0-9]{2}|7[0-4][0-9])/.

Irregular Coordinates

In the lite markup, irregular coordinates are variable width. The numbers can be positive or negative. For negative numbers, the '-' minus sign is replaced with the letter 'n'. The two numbers in the coordinate are separated by the letter 'x'. The center coordinate of (0,0) is represented by the string '0x0'. The coordinate (-250,-250) is represented by the string 'n250xn250'.

Although signs have a coordinate number limit of -250 to 249, irregular coordinates are unbounded when used for display with compounds of multiple signs and punctuation.

9. Regular Searching Form

The regular searching form consists of two parts: the lite markup for the text and a query string for searching the text. The query string is a concise representation for a much larger and detailed regular expression.

9.A. Lite Markup

The text of the regular searching form uses the lite markup with the token values for structural markers (A, B, L, M, R), symbol keys, and regular coordinates. Spaces separate words for signs and punctuation.

9.A.1. Pattern Examples

Sign

A sign is a combination of a lane maker (BLMR), followed by the maximum coordinate, followed by zero or more symbol keys with placement coordinates.

Example: M518x529S14c20481x471S27106503x489

Text	**Description**
M	middle lane
528x529	maximum coordinate (28,29)
S14c20	symbol key
481x471	placement coordinate (-19,-29)
S27106	symbol key
503x489	placement coordinate (3,-11)

Punctuation

A punctuation is a combination of a symbol key followed by a placement coordinate. The center is assumed to be the coordinate (0,0). The maximum coordinate is the additive inverse of the placement coordinate.

Example: S38800464x496

Text	**Description**

S38800	symbol key
464x496	placement coordinate (-36,-4)
assumed value of 536x504	maximum coordinate (36,4)

9.A.2. Regular Expressions

Structure	**Regular Expression**
Symbol key	S[123][0-9a-f]{2}[0-5][0-9a-f]
Coordinate	[0-9]{3}x[0-9]{3}
Signbox	[BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*
Term	(A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)[BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*
Punctuation	S38[7-9ab][0-5][0-9a-f][0-9]{3}x[0-9]{3}
Text	((A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)?[BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*|S38[7-9ab][0-5][0-9a-f][0-9]{3}x[0-9]{3})((A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)?[BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*| S38[7-9ab][0-5][0-9a-f][0-9]{3}x[0-9]{3})*

9.B. Query String

The query string is a concise representation for a much larger and detailed regular expression. The query string is a combination of several types of searches of symbols and ranges.

9.B.1. Pattern Examples

SignBox Search

The query string to search for all signboxes is simply the letter “Q”. This will return the regular expression for a signbox. This example has a ratio of 1 letter of query string to 75 letters of regular expression.

Query String: Q
Regular Expression: [BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	**Description**
Q	Query string for all signboxes

Term Search

The query string to search for all terms is simply the letters “QT”. This will return the regular expression for a term. This example has a ratio of 1 letter of query string to 55.5 letters of regular expression.

Query String: QT
Regular Expression: (A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)[BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	Description
Q	Query string for signboxes
T	Query string for terms

Exact Symbol Search

The query string to search for a specific symbol uses the exact symbol key. This will return the regular expression for a term. This example has a ratio of 1 letter of query string to 21 letters of regular expression.

Query String: QS10000
Regular Expression: [BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*S10000[0-9]{3}x[0-9]{3}(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	Description
Q	Query string for signboxes
S100	Symbol key prefix for base
0	Specific fill value
0	Specific rotation value

Generic Symbol Search

The query string to search for a generic symbol replaces either one or both of the fill or rotation aspect of the symbol key with 'u' for unspecified. This will match all symbol keys for the specific symbol base without regard to fill or rotation. This example has a ratio of 1 letter of query string to 23 letters of regular expression.

Query String: QS100uu
Regular Expression: [BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*S100[0-5][0-9a-f][0-9]{3}x[0-9]{3}(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	Description
Q	Query string for signboxes

S100	Symbol key prefix for base
u	Unspecified fill value
u	Unspecified rotation value

Symbol search with approximate location

It is possible to search for a symbol around an approximate location. The position is specified with a regular coordinate string. The default variance for coordinates is plus or minus 20. This example has a ratio of 1 letter of query string to 14 letters of regular expression.

Query String: QS14c20481x471
Regular Expression: [BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*S14c20((46[1-9])|(4[7-9][0-9])|(50[01]))x((45[1-9])|(4[6-8][0-9])|(49[01]))(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	Description
Q	Query string for signboxes
S14c20	Exact symbol key
481x471	Approximate symbol location at coordinate (-19,-29)

Symbol search with approximate location and custom variance

It is possible to search for a symbol around an approximate location with a custom variance. The custom variance in this example is 10, so the coordinate numbers can vary by plus or minus 10. This example has a ratio of 1 letter of query string to 12 letters of regular expression.

Query String: QS14cuu481x471V10
Regular Expression: [BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*S14c[0-5][0-9a-f]((47[1-9])|(48[0-9])|(49[01]))x((46[1-9])|(47[0-9])|(48[01]))(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	Description
Q	Query string for signboxes
S14cuu	Generic symbol key without specific fill or rotation
481x471	Approximate symbol location at coordinate (-19,-29)
V10	Custom variance of 10

Range Search

It is possible to search for a range of symbol bases. The query string does not specify a fill or rotation value. It has a start based and an end base. This will match all symbol keys that are inside the range. This example has a ratio of 1 letter of query string to 21 letters of regular expression.

Query String: QR2fft36c
Regular Expression: [BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*S((2ff)|(3[0-5][0-9a-f])|(36[0-9a-c]))[0-5][0-9a-f][0-9]{3}x[0-9]{3}(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	Description
Q	Query string for signboxes
R	Marker for start of range
2ff	Initial base value for search
t	Separation character for range
36c	Final base value for search

Range Search with approximate location

It is possible to search for a range of symbol bases. The query string does not specify a fill or rotation value. It has a start based and an end base. This will match all symbol keys that are inside the range. This example has a ratio of 1 letter of query string to 14 letters of regular expression.

Query String: QR2fft36c480x480
Regular Expression: [BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*S((2ff)|(3[0-5][0-9a-f])|(36[0-9a-c]))[0-5][0-9a-f]((4[6-9][0-9])|(500))x((4[6-9][0-9])|(500))(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	Description
Q	Query string for signboxes
R	Marker for start of range
2ff	Initial base value for search
t	Separation character for range
36c	Final base value for search
480x480	Approximate location of symbol from specified range

Term search with range and exact symbol

It is possible to use multiple criteria for the search query. This example searches for terms that include a symbol from a range and an exact symbol. This will require 2 regular expressions that are performed is sequence. This example has a ratio of 1 letter of query string to 23 letters of regular expression.

Query String: QTR2fft36cS10000
Regular Expression 1: (A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)[BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*S10000[0-9]{3}x[0-9]{3}(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Regular Expression 2: (A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)[BLMR]([0-9]{3}x[0-9]{3})(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*S((2ff)|(3[0-5][0-9a-f])|(36[0-9a-c]))[0-5][0-9a-f][0-9]{3}x[0-9]{3}(S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

Text	Description
Q	Query string for signboxes
T	Query string for terms
R2fft36c	Range search for 2ff to 36c
S10000	Search for exact symbol S10000

9.B.2. Regular Expressions

Query	Regular Expression
SignBox	Q
Term	QT
Range with optional location	R[123][0-9a-f]{2}t[123][0-9a-f]{2}([0-9]{3}x[0-9]{3})?
Symbol with option location	S[123][0-9a-f]{2}[0-5u][0-9a-fu]([0-9]{3}x[0-9]{3})?
Variance	(V[0-9]+)
Full query string	QT?(R[123][0-9a-f]{2}t[123][0-9a-f]{2}([0-9]{3}x[0-9]{3})?)*(S[123][0-9a-f]{2}[0-5u][0-9a-fu]([0-9]{3}x[0-9]{3})?)*(V[0-9]+)?

9.C. Filter

Query strings will return 1 or more regular expressions. The regular expressions should be executed against the input in series.

The output from the first regular expression should return a list of words that matched the first regular expression. The next regular expression should be executed against the returned list of words. This pattern should be followed until all the regular expressions have been executed.

The final list of words is the result of the search. This list of words represents signboxes or terms that match each of the regular expressions. This is the filter and repeat pattern.

9.D. Displacement

When searching with approximate location, the symbol's position is based on the center of the signbox rather than the relation between the symbols. If we want to find all examples of a sign, we will want to account for signs that are not in the same location with regards to the center of the signbox. This requires accounting for displacement.

Displacement can be cause by the use of head symbols (which changes the signbox center) or caused by multiple signs in the same signbox (which changes the signs relation to the signbox center).

9.D.1. Displacement Grid

To search for displacement, we'll need to use 8 additional query strings. We can either add or subtract the double of the variance to the X and/or Y values for each of the coordinates. The default variance is 20, so the displacement is adjusted by +/- 40

X-40, Y-40	Y-40	X+40, Y-40
X-40	Original Query String	X+40
X-40, Y+40	Y+40	X+40, Y+40

9.D.2. Displacement Example

Consider this query string: QS14c20481x471S27106503x489

QS14c20441x431S27106463x449	QS14c20481x431S27106503x449	QS14c20521x431S27106543x449
QS14c20441x471S27106463x489	QS14c20481x471S27106503x489	QS14c20521x471S27106543x489
QS14c20441x511S27106463x529	QS14c20481x511S27106503x529	QS14c20521x511S27106543x529

10. Variant Display Forms

The variant display forms consists of several formats. Through simple processes, one format can be transformed into another.

Each format uses a lite markup with the token values for structural markers (A, B, L, M, R), symbol keys, and irregular coordinates. Spaces separate words for signs and punctuation.

Structure	**Regular Expression**
Symbol key	S[123][0-9a-f]{2}[0-5][0-9a-f]
Coordinate	n?[0-9]+xn?[0-9]+

10.A. Raw

The raw display format string contains the minimal amount of data required to represent text. It defines signs and punctuations. The signboxes are neither centered or sized. A signbox can occur anywhere in the signbox space and the center is not assumed to be the coordinate (0,0). The maximum coordinate for a signbox is unstated. Likewise, the punctuation does not contain any placement information. Layout is impossible without access to an outside datasource.

10.A.1. Pattern Examples

Sign

A sign is a combination of a lane maker (BLMR), followed by zero or more symbol keys with placement coordinates.

Example: MS14c20n19xn29S271063xn11

Text	Description
M	middle lane
S14c20	symbol key
n19xn29	placement coordinate (-19,-29)
S27106	symbol key
3xn11	placement coordinate (3,-11)

Punctuation

A punctuation is represented with a single symbol key.

Example: S38800

Text	Description
S38800	symbol key

10.A.2. Regular Expressions

Structure	Regular Expression
Signbox	[BLMR](S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*
Term prefix	A(S[123][0-9a-f]{2}[0-5][0-9a-f])+
Punctuation	S38[7-9ab][0-5][0-9a-f]
Text	((A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)?[BLMR](S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*|S38[7-9ab][0-5][0-9a-f])((A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)?[BLMR](S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*| S38[7-9ab][0-5][0-9a-f])*

10.B. Expanded

The expanded display format string contains sizing information (width and height) for every symbol outside of the term prefix. The maximum coordinate for a signbox can be calculated by adding the symbol width and height to the symbol placement coordinate.

For any symbol key in the signbox or for punctuation, the width and height is accessed from an outside data source. The size information is written as an irregular coordinate and appended to the symbol key through a simple search and replace.

10.B.1. Pattern Examples

Sign

A sign is a combination of a lane maker (BLMR), followed by zero or more symbol keys with sizing information followed by placement coordinates.

Example: MS14c2023x31xn19xn29S2710615x40x3xn11

Text	Description
M	middle lane
S14c20	symbol key
23x31	width of 23 and height of 31 for symbol S14c20
x	divider between size and placement
n19xn29	placement coordinate (-19,-29)
S27106	symbol key
15x40	width of 15 and height of 40 for symbol S27106
x	divider between size and placement
3xn11	placement coordinate (3,-11)

Punctuation

A punctuation is represented with a symbol key and a size coordinate

Example: S3880072x8

Text	Description
S38800	symbol key
72x8	width of 72 and height of 8 for symbol S38800

10.B.2. Regular Expressions

Structure	Regular Expression
Signbox	[BLMR](S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]+x[0-9]+xn?[0-9]+xn?[0-9]+)*
Term prefix	A(S[123][0-9a-f]{2}[0-5][0-9a-f])+

Punctuation	S38[7-9ab][0-5][0-9a-f][0-9]+x[0-9]+
Text	((A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)?[BLMR](S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]+x[0-9]+xn?[0-9]+xn?[0-9]+)*| S38[7-9ab][0-5][0-9a-f][0-9]+x[0-9]+)((A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)?[BLMR](S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]+x[0-9]+xn?[0-9]+xn?[0-9]+)*| S38[7-9ab][0-5][0-9a-f][0-9]+x[0-9]+)*

10.C. Layout

The layout display format string contains the maximum coordinate as a preprocessed value for signboxes and it contains the placement coordinate for punctuation. It is equivalent to the lite markup for the regular searching form, but with irregular coordinates.

10.C.1. Pattern Examples

Sign

A sign is a combination of a lane maker (BLMR), followed by the maximum coordinate, followed by zero or more symbol keys with placement coordinates.

Example: M18x29S14c20n19xn29S271063xn11

Text	Description
M	middle lane
18x29	maximum coordinate (28,29)
S14c20	symbol key
n19xn29	placement coordinate (-19,-29)
S27106	symbol key
3xn11	placement coordinate (3,-11)

Punctuation

A punctuation is a combination of a symbol key followed by a placement coordinate. The center is assumed to be the coordinate (0,0). The maximum coordinate is the additive inverse of the placement coordinate.

Example: S38800n36xn4

Text	Description
S38800	symbol key
n36xn4	placement coordinate (-36,-4)
assumed value of 36x4	maximum coordinate (36,4)

10.C.2. Regular Expressions

Structure	Regular Expression
Signbox	[BLMR]([0-9]+x[0-9]+)(S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*
Term prefix	A(S[123][0-9a-f]{2}[0-5][0-9a-f])+
Punctuation	S38[7-9ab][0-5][0-9a-f]n?[0-9]+xn?[0-9]+
Text	((A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)?[BLMR]([0-9]+x[0-9]+)(S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*| S38[7-9ab][0-5][0-9a-f]n?[0-9]+xn?[0-9]+)((A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)?[BLMR]([0-9]+x[0-9]+)(S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*| S38[7-9ab][0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*

10.D. Panel

A panel display format string combines multiple signs and punctuations into a unit as a defined height column or defined width row. Each signbox contains an offset coordinate that is used to position the symbols inside of the signbox. The offset is added to the placement coordinate to determine the position of each symbol on the panel.

10.D.1. Pattern Example

Panel Prefix

Each panel begins with a panel display marker “D” followed by a sizing coordinate. The top-left of the panel is taken to be the coordinate (0,0) such that the sizing coordinate can be understood as the width and height of the panel as well as the maximum coordinate.

Example: D102x200

Text	Description
D	Panel display marker
102x200	width (102) and height (200) of panel

SignBox with offset

Each panel can contain several signboxes. Each signbox has it's own offset coordinate. The offset coordinate is used to determine the position of the signbox's symbols within the panel.

Example: _M51x49S14c20n19xn29S271063xn11

Text	Description
_M	signbox marker
51x49	coordinate offset for signbox (51,49)
S14c20	symbol key
n19xn29	placement coordinate (-19,-29) offset to coordinate (32,20)
S27106	symbol key
3xn11	placement coordinate (3,-11) offset to coordinate (54,38)

Full Panel

A full panel includes the panel prefix with several signboxes with offsets.

Example:
D102x200_M51x49S14c20n19xn29S271063xn11_M51x139S1870an11x15S18701n18xn10S205008xn4S2e7340xn32_B51x196S38800n36xn4

Text	Description
D	Panel display marker
102x200	width (102) and height (200) of panel
_M	signbox marker
51x49	coordinate offset for signbox (51,49)
S14c20	symbol key
n19xn29	placement coordinate (-19,-29) offset to coordinate (32,20)
S27106	symbol key
3xn11	placement coordinate (3,-11) offset to coordinate (54,38)
_M	signbox marker
51x139	coordinate offset for signbox (51,139)
S1870a	symbol key
n11x15	placement coordinate (-11,15) offset to coordinate (40,154)
S18701	symbol key
n18xn10	placement coordinate (-18,-10) offset to coordinate (33,129)
S20500	symbol key
8xn4	placement coordinate (8,-4) offset to coordinate (59,135)
S2e734	symbol key
0xn32	placement coordinate (0,-32) offset to coordinate (51,107)

_B	signbox marker
51x196	coordinate offset for signbox (51,196)
S38800	symbol key
n36xn4	placement coordinate (-36,-4) offset to coordinate (15,192)

10.D.2. Regular Expressions

Structure	Regular Expression
Signbox with offset	_[BLMR]([0-9]+x[0-9]+)(S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*
Panel	D[0-9]+x[0-9]+(_[BLMR]([0-9]+x[0-9]+)(S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*)*
Panels	D[0-9]+x[0-9]+(_[BLMR]([0-9]+x[0-9]+)(S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*)*(D[0-9]+x[0-9]+(_[BLMR]([0-9]+x[0-9]+)(S[123][0-9a-f]{2}[0-5][0-9a-f]n?[0-9]+xn?[0-9]+)*)*)*

10.D.3. Transformational Parameters

Transformation parameters are used to transform a layout display string into a panel display string. When written vertically in columns, the length represents the vertical length that all columns share in common. When written horizontally in rows, the length represents the horizontal length that all rows share in common.

Name	Value	Description
length	number of pixels	The chunk size of columns or rows
width	number of pixels	The width of the column or row
padding	number of pixels	Distance from closest symbol to width edge.
form	col or row	Form of display panel
style	fix or flex	The style of the width is either fixed by the width or flexible. For a flexible style, the padding is used as the distance from the edge of the column or row and the edge of the closest symbol. If a width is given, a flexible style will use the width as the minimum column or row width.
signTop	number of pixels	Padding before a sign.
signBottom	number of pixels	Padding after a sign.
puncTop	number of pixels	Padding before a punctuation.
puncBottom	number of pixels	Padding after a punctuation.

offset	number of pixels	The horizontal offset from the center of the middle lane to either the left or the right.
top	number of pixels.	The distance from the start of the column or row and the edge of the first symbol.
justify	option number: 1, 2, 3	Justify 1 pulls punctuation to the end of a column or row by moving signs closer together. Justify 2 pushes sign apart to evenly cover a column or row. Justify 3 will both pull punctuation and push signs.

11. Sorting Terminology

Terms are signboxes with an optional prefix as an ordered list of symbols. The term prefix is ordered in 1 dimension, whereas a signbox is ordered in 2 dimensions. The term prefix is distinct from the signbox, but will often use the same symbols. Neither structure can be automatically derived from the other.

11.A. ISWA Sort Order

The symbols of the ISWA have been ordered in a meaningful and international manner. This ordering is evident in the symbol IDs and the symbol codes. This ordering has been maintained in the various encoding forms and allows for sorting using a binary string comparison.

Symbol orders other than the default ISWA order are not directly supported. This may be an issue if a language group chooses an alternative order for their specific hand shape subset. It is advised to maintain the same hand shape order as the ISWA for all subsets. Otherwise, sorting will require the addition of an extra layer of processing such as a custom Unicode sorting table.

11.B. Explicit Prefix

A term prefix is a list of writing symbols and/or detailed location symbols. A valid term prefix must contain at least one symbol and can not contain punctuation. The term prefix is used to define a temporal order.

There are several theories on the best way to structure a term prefix. The most productive is based on the SignSpelling Sequence theory of Valerie Sutton. A SignSpelling Sequence is structured as a series of starting handshapes followed by optional movements, transitional handshapes, movement, and end handshapes. Only symbols from category 1 (hands) and category 2 (movement) should be used in this first section. The last section of the sequence should contain symbols of dynamics & timing, head & face, or body: categories 3, 4, and 5.

Detailed location symbols from category 6 can be used in a term prefix, but are rarely (if ever) needed for a sequence in general writing.

11.C. Sorting Unordered SignBoxes

Signboxes are ordered in 2 dimensions and as such do not have a normative 1 dimensional order. It is difficult to correctly sort a list of signboxes that do not have term prefixes.

A possible solution is to search a data source for terms that are an approximate match for each signbox without a term prefix. The term prefix from the search results can be prepened to the signbox without a term prefix so that sorting is possible and meaningful.

Given a signbox as: M518x529S14c20481x471S27106503x489

A query string can be constructed by replacing the lane marker and maximum coordinate with the query string for terms of "QT". A first attempt could search for an exact match by adding a variance of 0 by appending "V0" to the end of the query string.

Resulting query string: QTS14c20481x471S27106503x489V0

12. Spelling Normalization

Normalization is the process of making something normal with regards to a standard that has been accepted by a community.

SignWriting has two issues with spelling normalization: the visual appearance as 2-dimensional order and the formal string as 1-dimensional order. With SignWriting it is possible to have several strings have the same exact 2-dimensional visual appearance. For complete normalization, a spelling standard must specify both the visual appearance and the formal string.

12.A. Unordered and Inexact

Proper spelling with SignWriting is primarily concerned with the 2-dimensional visual appearance of a written sign. As the writer can start with any symbol and a writer can precisely position each symbol, spelling is considered unordered and inexact.

Artificial rules can be created that limit the variability, but none are universal or inherent in the SignWriting script.

12.B. Exact Spelling

Outside of a two writers using the same editor with restricted rules, it is unlikely that two writers will produce the exact same spelling for any sign. The more complicated the sign, the less chance of producing an exact match.

Exact spellings are only possible if two writers normalize to the same standard.

12.C. Reflected Statistics

There are two ways to achieve this standard of normal spelling: either a dictate from an accepted authority or an analysis of a large and coherent body of writing.

When considering any spelling, there are three statistics which can be helpful: strength, association, and displacement.

The strength of a spelling is evident in the number of time the exact spelling has been used in a body of writing. A query search with a variance of "0" can be used to determine the strength of a spelling.

The association of a spelling is evident in the number of approximate spelling matches. A standard query search can be used to determine the association of a spelling.

The displacement of a spelling can be realized by using the displacement search method. A spelling displacement will report the number of times a spelling was used with a different relation to the center of the signbox. The displacement usually deals with combination signs, when 2 signs are written together.

For spelling normalization, the reflected statistics can provide a guide when choosing one spelling over another.

12.D. Symbol Subsets

The ISWA is a huge set of symbols. There is no language that will use every symbol. As with reflected spelling statistics, a body of writing can be analyzed for the symbols that have been used. Reflected symbol statistics can provide a guide to the norms within a community. If the writer is offered a symbol subset rather than the entire ISWA, the symbol subset can become self reinforcing and aid in spelling normalization.