Instrumental Environment for Supporting Sensorimotor Development

Boris Karapetyan
Institute for Informatics and Automation Problems of NAS RA
Yerevan, Armenia
boris.karapetyan@iiap.sci.am

***Abstract*—This paper presents a system of digital tools and methods designed to support the sensorimotor development of children through handwriting training, using the Armenian alphabet as a case study. The approach integrates a structured model that links sensory, motor, and cognitive development. As a foundational stage of a broader framework titled "Models and Programs for Sensorimotor Development of Intelligence," this system aims to facilitate the formation of fine motor skills, support speech and literacy acquisition, and enable practical, adaptive learning environments. The technological implementation includes XML-based letter representations, dynamic SVG animations, and printable PDF worksheets, providing a versatile platform for individualized and scalable education.**

***Keywords*—Handwriting training, fine motor skills, SVG-animation, XML modeling, XSLT transformation.**

I. Introduction

Sensorimotor development is a crucial stage in a child's ontogenesis, forming the foundation for the development of higher mental functions such as attention, memory, speech, thinking, and self-regulation [1]. The sensitive period for the formation of motor and sensory skills occurs before the age of 7, and it is during this time that targeted intervention is most effective [2, 5]. The development of fine motor skills contributes to the formation of speech and writing, as well as successful adaptation to academic activities [3].

Creating a model makes it possible to construct a logical-structural scheme of interaction between sensory, motor, and cognitive components of development. Implementing a program based on the model enables a transition from theoretical representations to practical correctional work with the ability to monitor developmental progress.

The choice of handwriting as the starting stage is based on several reasons [3, 4]:

- Children learn not only neat and beautiful hwriting, but also perseverance and patience
- While writing, the child learns to evaluate the overall width of words and spacing between them. They begin to understand what interletter and interline spacing are, and they discover the structure of letters.
- Repetitive writing of letters in workbooks ensures solid memorization and significantly faster acquisition of reading skills.
- Handwriting helps speech development. When writing a text, the person often subvocally pronounces the words, memorizes turns of phrase, and expands their vocabulary.
- Penmanship develops fine motor skills. While forming letters, coordination improves, visual estimation is refined, and the muscles of the fingers, hand, arm, and forearm are strengthened. Writing letters in copybooks activates rarely used areas of the brain, which helps you learn letters faster and learn to read.

II. Development Environment for Handwriting Training Stands

For lessons, we offer a demonstration of pen movement while writing a letter and worksheets in which letters are shown with decreasing detail: first with dense dotted lines, then increasingly faint and sparse. The sequence for teaching letters can be selected based on frequency of use, which allows inclusion of full words in early lessons.

We chose the SVG format to visualize hand movement. First, SVG is a standard XML format, allowing the use of a rich set of tools for generating, editing, and displaying images. All popular browsers support SVG, which enables fast previewing of changes. SVG's animation capabilities are sufficient for our goals and easily allow adjustment of animation speed, combining different image fragments, changing their sizes, and positioning them.

The format for workbooks is PDF. The generation of worksheets and animations is performed using XSLT. We developed a unified XML file for all letters and stylesheets for generating both SVG and PDF. For SVG transformation, any XSLT processor can be used—we use SAXON. PDF worksheets can be created using an XSL-FO formatter such as XEP (RenderX), AntennaHouse, or the free FOP. The styles are parameterized to set the list of letters, their size, animation speed, and the level of letter detail in the worksheets.

A. XML File Format

All alphabet letters are stored in a single XML file as a sequence of <*letter*> elements, with sub-elements <*caps*> and <*smll*> for uppercase and lowercase letters, respectively. Each <*letter*> element has an *id* attribute with an IPA value (International Phonetic Alphabet). The main component of the SVG image is the element <*path*> with *style* and *d* attributes.

The *style* attribute specifies line thickness, color, transparency, etc. The *d* attribute describes the line as a sequence of Bézier curves (cubic or quadratic).

If a letter is drawn as a sequence of separate strokes, <*path*> elements are defined in that same sequence. The

style attribute for <*path*> is defined in the stylesheet during SVG and PDF generation.

The path value is specified in <*caps*> and <*smll*> as sub-elements <*d-all*> (letter as one line) and one or more <*d*> elements. Multiple <*d*> elements are needed when a letter consists of multiple strokes (See Fig. 1).

```
<letter id="[IPA]">
  <caps>
    <d-all dw="...">M 36.14071,117...</d-all>
    <d start="ts" end="te">M 36.14071,C ...</d>
    <d start="ts" end="te">M ...</d>
    ...
    <d>M ...</d>
  </caps>
  <smll>
    <d-all dw="...">M 36.14071,117.69 C ...</d-all>
    <d start="ts" end="te">M 36.14071,117...</d>
    <d start="ts" end="te">M ...</d>
    ...
    <d>M ...</d>
  </smll>
</letter>
```

Fig. 1. Structure of the <letter> element

B. *Filling the XML File*

We propose using the Inkscape program (available at inkscape.org) to build the visual figure.

- Create a "ruler" layer: four straight lines define the dimensions for uppercase and lowercase letters—baseline, ascender/descender lines, and capital letter height.
- On a new layer, load the original letter shape (e.g., from Google Fonts). Scale it to the desired size.
- Draw a polyline over the shape along the bends. For letters with multiple parts, draw multiple polylines.
- Convert the polylines into Bézier curves. These curves can then be fine-tuned to match the original letter shape.
- Use Inkscape's XML editor to extract the d attribute values for use in the XML file.

C. *SVG Animation Parameters*

Currently, we use two animation types:

1. Movement of an object along the path using.
2. Stroke coloring via the stroke-dashoffset attribute.

All elements for a single letter are grouped within a container <*g*>, horizontally positioned using the transform attribute.

A pen icon moves along each path defined in the XML <*letter*> element. The *dur* attribute sets the duration in seconds, calculated from the *start* and *end* values in XML-*d* elements. Additional <*animate*> elements are used to show/hide the pen during the animation.

For color animation, stroke-dashoffset is used – i.e., a dotted line style with a special "5000;0"-value, where the first value determines the length of the dash, the second – the length of the space between the dashes. A large dash value leads to a gradual filling of the path. Buttons with IDs (e.g., PenAnim and StrokeAnim) trigger each animation. An example of such animation is shown in Fig. 2.

```
<g id="1" transform="translate(X-space)">
    <path style="lines parameters" d="path
description"/>
  <use href="#pen" x="X-start" y="Y-start"
stroke="color" stroke-width="0.1">
     <animate begin="PenAnim.begin"
    attributeName="stroke-opacity" from="0" to="0"/>
    <animate begin="PenAnim.begin" dur="10s"
    attributeName="stroke-opacity" from="1" to="1"/>
    <animateMotion dur="8s" begin="PenAnim.begin"
    repeatCount="freeze" path="pen path"/>
  </use>
  <path d="letter line" style="stroke:color;stroke-
width:2;stroke-dasharray:5000">
    <animate begin="StrokeAnim.begin+0.3s"
  attributeName="stroke-dashoffset" dur="25s"
  values="5000;0"/>
    <animate begin="StrokeAnim.begin" dur="3s"
  attributeName="stroke-opacity" from="0" to="0"/>
    <animate begin=" StrokeAnim.begin + 3s"
  attributeName="stroke-dashoffset" dur="25s"
  values="5000;0" repeatCount="freeze"/>
  </path>
</d>
```

Fig. 2. Animation example

III. Development Directions

A test implementation can be viewed at karapret.am. Animation and worksheet samples are shown in Figs. 3 and 4.

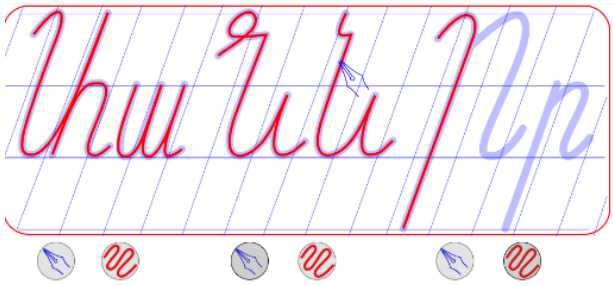

Fig. 3. Animation Example

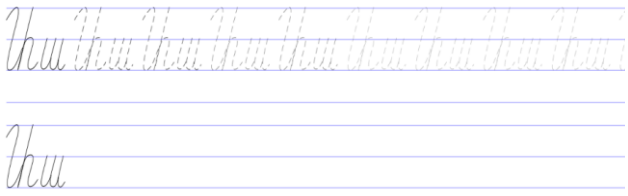

Fig.4. PDF Fragment

We aim to adapt this approach for graphics tablets without requiring an internet connection. Displaying a PDF worksheet as a background on-screen eliminates the need for printed workbooks.

To evaluate the learning process, we plan to add features for measuring task execution speed, tracking changes, and suggesting repetitions.

The choice of letters also defines user age groups. Any linear image, from simple shapes to complex mazes, can be used instead. Further collaboration with specialists will expand the application area.

We have prepared XML and SVG files for English, Russian, Georgian, and even Arabic, and we seek opportunities for collaborative development.

REFERENCES

[1] A.R. Luria, *Elements of Information Theory*, Moscow State University Press, Moscow, 1973.

[2] J. Piaget, *The Origins of Intelligence in Children.* International University Press, New York, 1952.

[3] P.Ya. Galperin, *Methods of Teaching and Mental Development of the Child*. Moscow State University Press, Moscow, 1985.

[4] M. Longcamp, M.T. Zerbato-Poudou and J.L. Velay, "The influence of writing practice on letter recognition in preschool children: A comparison between handwriting and typing", *Developmental Neuropsychology*, vol. 29, no. 2, pp. 261--276, 2006.

[5] J. Flavell, *The Developmental Psychology of Jean Piaget*, D. Van Nostrand Comp. Inc, Princeton, 1962.