Proposal for Digital Prewriting Assessments:
Upgrading Data Capture and Analysis Capability

A Proposal

By

Virginia Commonwealth University
Engineering Capstone Team CS-23-321

Submitted

On

October 1, 2022

For the

Virginia Commonwealth University Departments of
Occupational Therapy and Computer Science

Proposal for Digital Prewriting Assessments: Upgrading Data Capture and Analysis Capability

1. Introduction

1.1 *Purpose*

This document proposes a general upgrade to Dr. Virginia Chu's capture and analysis of digital prewriting assessment capabilities. This upgrade is necessary to enable Dr. Chu and other clinicians to make prewriting assessments more easily without the need to move data between the capture application and the analysis application.

1.2 *Background*

Prewriting Skills are necessary fundamental skills children develop before they can write. Often these skills refer to the ability to draw lines and shapes, as simple shapes are the natural step in development before letters or words. Prewriting skills are classified as complex fine motor skills developed during early childhood – typically between the ages of 3-5 years old – that require coordinating visual motor skills with other skills from sensory and motor domains. More specifically, prewriting skills serve as precursors to the ability to write more advanced forms such as "alpha-numeric".

It has been found that the early evaluation of prewriting skills is important because they set the stage for learning the fine motor and visual motor skills needed for writing later on in life. Evaluation of these skills allows for children who struggle with prewriting tasks to be identified early. Consequently this provides the necessary data for intervention that helps address deficits in these foundational skills before academic demands become more challenging. Currently, occupational therapists assess prewriting skills with pencil and paper assessments. Assessments typically require children to look at an image of a simple shape and then recreate that image by drawing it with a pencil. An example assessment, specifically the one Dr. Chu used in her research, is shown in figure 1. These assessments are quick to administer, taking roughly 10 - 15 minutes, but require a large portion of the therapist's time and effort to score. Notably, therapists are limited in their ability to assess the subtle differences between the shown visual image and a child's drawing. Additionally, the scoring of these assessments can be highly subjective despite scoring guides that exist to alleviate disparities. Ultimately, when it comes to researching prewriting skills, because of the subjectivity in scoring, and the fact that the drawings in each assessment are scored as either right or wrong, the assessments do not capture the changes in performance across different ages and ability levels.

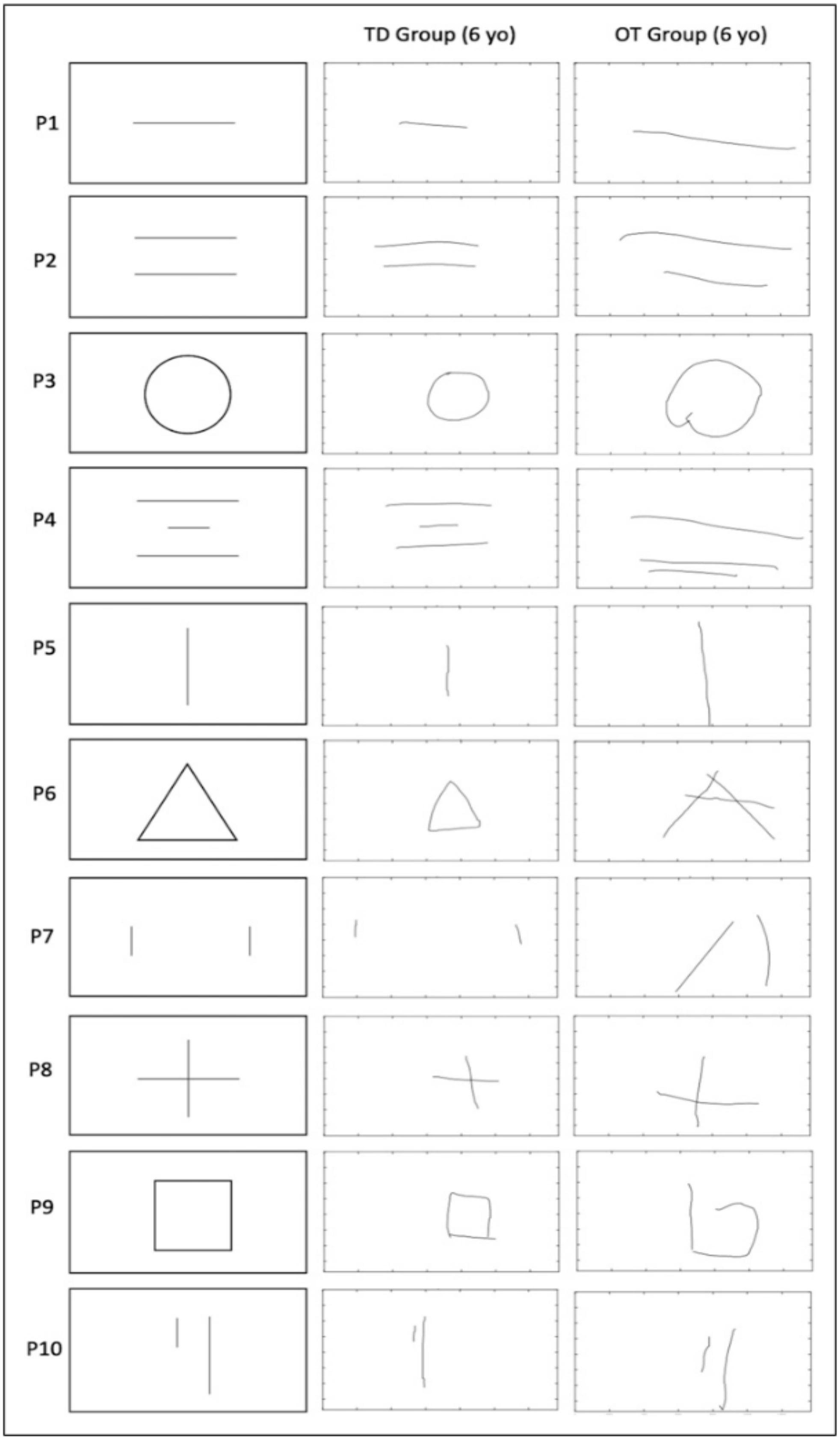

Figure 1. Example prewriting assessment with sample data

To address the problems with subjectivity and the lack of ability to meaningfully capture performance across different ages and ability levels, Dr. Chu developed a more objective system. Her system uses two separate components, one to capture the data, that is, to collect information about each drawing a child produces during an assessment, and one to analyze the data, or, score the respective assessments of each child.

The drawing data collection component of the system is a program on a computer tablet, developed using Visual C++, that allows Dr. Chu to record a child's drawing as position vs time data. *Visual C++* is a work environment for creating computer programs in the programming language of C++. *C++* is a programming language initially developed in 1985 that has a very specific format and is primarily used to create high-performance applications. Currently, a child draws with their finger on the tablet and what they draw shows up on the screen beneath their finger as they draw. When they have finished, the data that describes their drawing is saved before moving to the next image in the assessment. At the end of the assessment there is a digital folder that contains all of the data from each of the drawings in the assessment.

The drawing data analysis component of the system is on a desktop computer and uses MATLAB to analyze and score the assessments. *MATLAB* is both a programming language and a work environment and is used to create and execute programs that require numerical computations. MATLAB programs are typically created to analyze data and in this project the original MATLAB program was created to analyze the drawings children drew during the prewriting assessments and score them based on a number of criteria.

Presently to process the information collected when a child takes Dr. Chu's digital prewriting assessment, the data from each of the drawings in the assessment has to be transferred to the desktop computer, via file sharing, where the MATLAB program resides before the child's assessment can be scored. This process takes time and effort that could otherwise be devoted to other tasks. Additionally the application for capturing the data from the drawings is specific to Dr. Chu's Windows™ tablet which prevents the deployment to other touch-enabled devices. Ultimately without a single, fully-integrated application that can capture and score the assessments and be deployed on a multitude of touch enabled devices, including iPads™ and Windows™ tablets, Dr. Chu is unable to expand her current level of research in the prewriting skills of children.

1.3 *Scope*

This proposal addresses only the upgrade for Dr. Virginia Chu's capture and analysis of prewriting assessment capabilities by means of the creation of a Web Application. This proposal does not include other graphics, modeling, or analysis tasks. Nor does this proposal include the wide scale deployment or subsequent maintenance of the proposed Web Application.

2. Discussion

2.1 *Approach*

Capstone team CS-23-321 proposes to Dr. Virginia Chu that the digital prewriting assessment data capture and analysis system be upgraded by integrating the functionalities from the current two applications into a single web application with a frontend to capture data and an appropriate backend server to support the analysis of the data.

The frontend will control what the therapists and children see, the user interface, and how the system can receive data, the user inputs. The user interface will be quite simple in structure. In order to prevent children from tampering with the test or testing system, and to reduce distractions, the main user interface will consist of two components. These two components will be the only things visible on screen and are: (1) the drawing for the children to look at and duplicate; and (2) a blank canvas area for the children to draw on. The blank drawing canvas will be completed by utilizing basic HTML Canvas elements. Native to HTML and JavaScript, Canvas elements are capable of storing x and y coordinates of a drawing, as well as the timestamp associated with each pair of coordinates. This data will be stored into a JSON (JavaScript Object Notation) object, which will be sent to the backend as part of a POST request. *HTML* is a markup language that is used to build websites and web applications. HTML provides the necessary tools to structure web applications and is equivalent to the foundation and framing of a house but for web applications. *JSON* is a convenient method to send information between web applications and the server where other information is stored or a server where calculations are completed. JSON is very compact and efficient which makes it convenient for fast transfer of data.

Each component of the two components will be equally sized and each will represent half of the screen space available. The displayed drawing will be the top half of the screen and the drawing area will be the bottom half of the screen. The user inputs will allow for children to draw in the drawing area and for therapists to control the overall test being administered. The drawing component will accept touch inputs and, as input is received, will render a drawing to match. So that the therapists have control over the test, they will have the ability to use an external Bluetooth keyboard in order to perform simple functions. These functions include: (1) 'clear' to clear the drawing area; (2) 'next' to go to the next displayed drawing image and send the completed drawing data for the previous image; (3) 'score' to score the test and display results. The above user interface and user input functionality will be written in the ReactJS framework.

ReactJS is a JavaScript library that facilitates the creation of interactive user interfaces for web applications. ReactJS allows a programmer to create certain "views" which are used to control what users see on their screen. A good example of a web application that use ReactJS is Netflix. *JavaScript* is a programming language that is a core component for designing and building web applications.

The backend is where all computation for analysis will happen and all functionalities outside of the frontend will reside. The backend will be written in Python and will use the Flask framework to communicate with the frontend. *Python* is a general purpose programming language that is used to create programs, or *scripts*, that are easy to read. Python is used to create programs that require data analysis. *Flask* is a lightweight framework for integrating Python code with web applications. The backend will handle all mathematical tasks required to score a child's drawing. It is important to note that each drawing has multiple criteria that get scored and thus the total score for any given drawing is a combination of the individual scores for each criteria. We consider only the task of scoring drawings created in response to one of the assessment images. In more detail, the image the children are asked to copy is a perfect circle and is shown in the top half of the screen. The children draw in the bottom half of the screen their best attempt at copying the circle. The data of what they drew is then sent to the backend. The circle drawing scores only rely on one criteria, the “roundness”. To score these drawings, which are intended to be circles, the backend will consist of one Python script that will calculate a score to determine the drawings “roundness”, that is, how round the drawn circle is. It will return the score to the frontend and the resulting score will be displayed to the screen. The roundness score calculation is described in further detail in the following process description.

PROCESS DESCRIPTION OF THE ROUNDNESS SCORE SCRIPT

Introduction

The *roundness score script* is a Python script in which positional data from a drawing is used to generate a numerical measurement of how round that drawing is. The roundness score script involves the following steps: center estimation, polar coordinate conversion, calculation of $D_{inscribed}$, calculation of $D_{circumscribed}$, and calculation of roundness.

Discussion

Estimation of the center of the drawing

The center of the drawing is estimated by first estimating the x-coordinate of the center point and then estimating the y-coordinate of the center point. The x-coordinate of the center point is estimated by taking the average of the smallest valued x-coordinate, the minimum, and the largest valued x-coordinate, the maximum. The minimum x and maximum x are selected from the complete list of all x-coordinates for a given drawing. The y-coordinate of the center point is estimated the same way, by taking the average of the smallest valued y-coordinate, the minimum, and the largest valued y-coordinate, the maximum. The minimum y and maximum y are selected from the complete list of all y-coordinates for a given drawing. This estimated center point is then used to convert all the x and y cartesian coordinates into polar coordinates.

Polar Coordinate Conversion

The two lists of x and y coordinates are converted into polar coordinates. That is, the two lists of x and y cartesian coordinates are used to generate two lists which contain the theta and radius values for the respective equivalent polar coordinates. These polar coordinates are used to more easily determine the possible diameters of inscribed and circumscribed circles.

Calculation of the diameter of the maximum inscribed circle , $\boldsymbol{D_{inscribed}}$

$D_{inscribed}$ is a measurement of the diameter of the largest circle that can be created within the circle that was drawn by a test subject. More specifically, to calculate $D_{inscribed}$ the computer must determine the shortest straight line that can be drawn. Since the points have been converted to polar coordinates, this is easy to do. The computer can simply grab the points that have an angle differing by pi, and thus are exactly opposite each other, and check how long the line is between those points. It is interesting to note that in cartesian coordinates, it is less obvious how to find the pairs of points that form the endpoints of this shortest straight line and hence polar coordinates were used. After calculating the diameter the method next needs to calculate the diameter of the smallest circle that can be circumscribed.

Calculation of the diameter of the minimum circumscribed circle , $\boldsymbol{D_{circumscribed}}$

$D_{circumscribed}$ is a measurement of the diameter of the smallest circle that can be created surrounding the circle drawn by a test subject. To calculate $D_{circumscribed}$ the computer can do a similar process to finding the maximum inscribed circle. In this case however it is looking for the longest line that can be drawn between two points. After calculating the diameter the method next needs to calculate the ratio between the two diameters also known as the roundness.

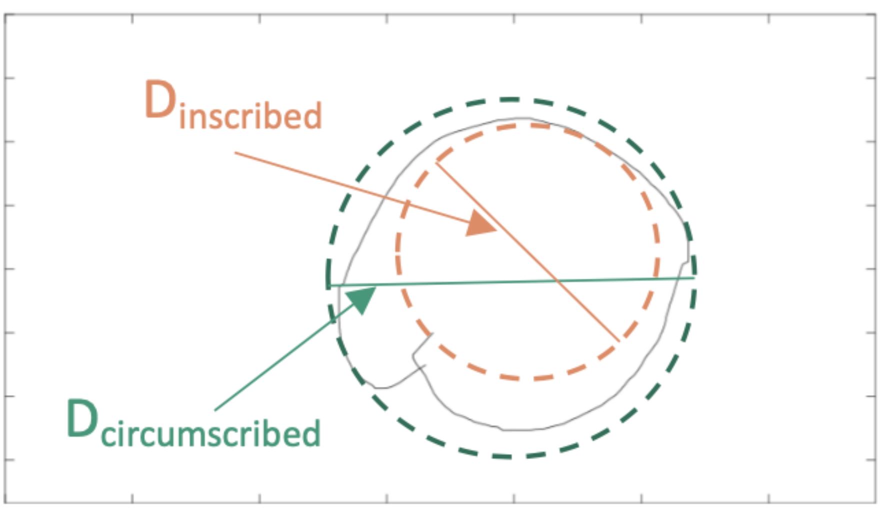

Figure 2, Visual example of roundness calculation

Calculation of roundness

Roundness is the ratio of the diameter of the largest circle that can be inscribed in the drawn circle and the diameter of the smallest circle that can be circumscribed around the drawn circle. Therefore, roundness is the ratio between $D_{inscribed}$ and $D_{circumscribed}$, that is,

$$Roundness = \frac{D_{inscribed}}{D_{circumscribed}}.$$

Two examples of roundness calculations are Figure 3 and Figure 4. Figure 3 is a baseline test of roundness on a perfect square and Figure 4 is a sample of a roughly drawn circle for comparison. The scores are 0.707 and 0.703 respectively.

Conclusion

The *roundness function* is a method in which positional data from a drawing is used to generate a numerical measurement of how round a circle was drawn. With roundness being the ratio of the diameter of the largest circle that can be inscribed and the diameter of the smallest circle that can be circumscribed. A perfect circle has a roundness score of 1, where the maximum inscribed circle is the same as the minimum circumscribed circle. It follows that in any drawing in which the drawn shape deviates from a perfect circle the roundness function would output a value less than 1.

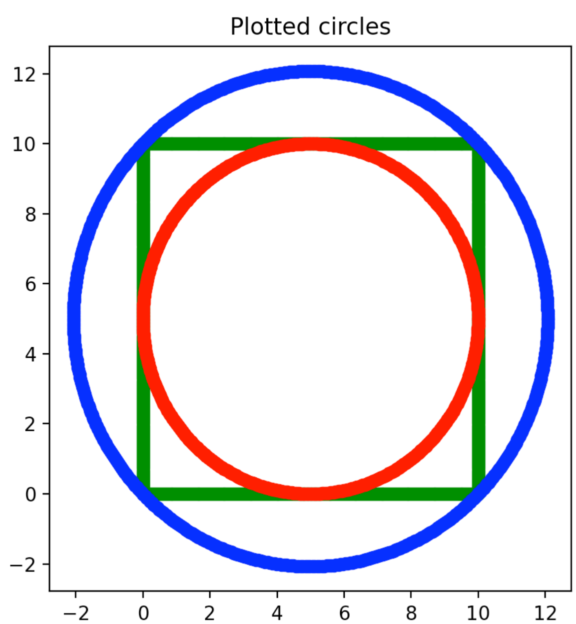

Figure 4

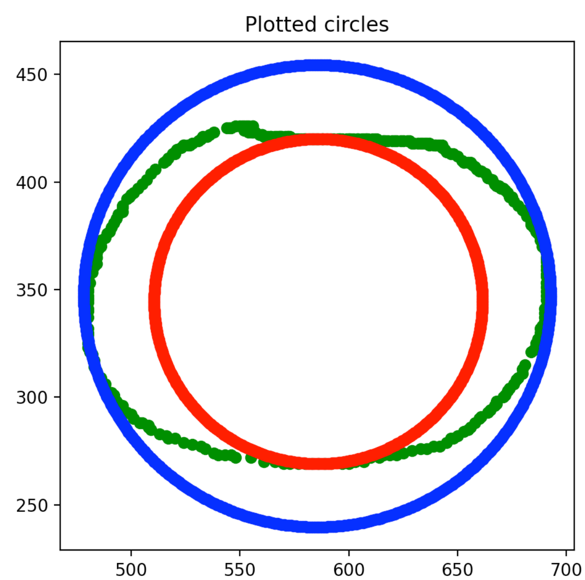

Figure 3

For the purposes of the proposed prototype, this application will run locally from each of the team laptops on a localhost server through the web browser. Specifically, a user's machine is used to access the network services that are running on the host. This means that each specific laptop will be running a version of the application that only the user of the laptop can use. For the prototype, instead of implementing a database, we will simply code the functionality to calculate the roundness score and display the assessment grade to the laptop screen. Following the protype, pending the approval of Dr. Chu, the application can then be put onto a web server so that it can be used on any device connected to the internet.

2.2 *Result*

The resulting application will be able to be used by multiple people on a wide selection of touch enabled devices so that many assessments can be administered at the same time. It is important to note, however, that the assessments will consist only of one test, the circle test, and the results of the assessment will not be stored between assessments. Instead the score is only displayed on the screen. The application will cut down processing time and increase the amount of assessments that can be administered per hour. It is conceivable that, upon completion of the proposed application, that further upgrades, including the remaining portions of the whole assessment could be implemented. Ultimately the fully completed single page application will allow Dr. Chu to scale her research to include larger sample sizes by providing a single source for digital prewriting assessment data collection and analysis.

2.3 *Statement of Work*

To achieve the goals of this proposal, the following tasks will be accomplished:

- Task 1: Develop high level architecture diagram (3 Hours)
- Task 2: Refine and refactor the high level diagram into an IPO diagram (6 Hours)
- Task 3: Construct a full "pseudocode" version of the application (6 Hours)
- Task 4: Divide functionalities among team members and begin development (1 Hour)
- Task 5: Development of application by means of the divided workload (30 Hours)
- Task 6: Testing and sponsored sample trial (20 Hours)

3. Resources

3.1 *Personnel*

The capstone team CS-23-321 is composed of a sponsor, an advisor, and four computer science students. The sponsor is Dr. Virginia Chu, Assistant Professor of Occupational Therapy and researcher at VCU. The advisor is Dr. Daniel Cranston, Associate Professor of Computer Science and researcher at VCU. The four computer science students are Noah Shields, Edward Ladia, Christopher Smith, and Charles Cutler. The sponsor is the "customer" and will determine the completion requirements for the project. The advisor and computer science students will work together to implement a solution to the described needs of the sponsor.

3.2 *Facilities and Equipment*

The team meets in VCU Engineering East Hall on the fourth floor in room 4228. This room provides adequate table space, includes a whiteboard and projector, and is available each week on our meeting date and time. Each of us uses our own personal laptops to accomplish our respective tasks for the project.

4. Costs

4.1 *Fiscal*

Our proposed solution involves the use of "open source" software which is free and significantly reduces the cost for developing the application. All in all, our proposed solution involves no costs.

4.2 *Time*

Assuming availability of the meeting space the entire upgrade can be completed in 66 hours. This estimate includes 3 Hours to develop the high level architecture diagrams, 6 hours refine and refactor the high level architecture diagrams into IPO diagrams; 6 hours to construct a full "pseudocode" version of the application; 1 hour to divide the functionalities that need to be constructed amongst the team members; 30 hours to develop the application by means of the divided workload; and 20 hours to work on testing and complete a sponsored sample trial with Dr. Chu.

5. Conclusion

5.1 *Summary*

The proposed web application will provide a cost effective and scalable solution to meeting the data capture and analysis requirements. The entire web application can be up and running moments after the final code is completed and can be scaled to include a wide array of desired shapes and scoring criteria. The web application is versatile, transcending the barrier between different operating systems by "living" on a web server. If connected to a database secured for healthcare data, the web application can be used by dozens of researchers simultaneously with no loss of data. Ultimately, the application provides crucial support to Dr. Virginia Chu in her research and is critical to the successful execution of a large scale study on the prewriting skills of children.

5.2 *Contact*

For more information regarding this proposal, contact Dr. Daniel Cranston, Ph.D., Associate Professor of Computer Science, Virginia Commonwealth University, by email. dcranston@vcu.edu.