STEM tech review

Accessibility of solutions for interacting with content in science, technology, engineering, and mathematics

MYNDIGHETEN FÖR TILLGÄNGLIGA MEDIER

2025 – Swedish Agency for Accessible Media

Diarienummer: MTM 2024/213

STEM tech review: accessibility of solutions for interacting with content in science, technology, engineering, and mathematics

Authors:
Tim Arborealis Lötberg, STEM specialist, MTM, editor and project manager
Lisa Hartun Bennedsen, Special pedagogical consultant, Nota
Evelina Frischenfeldt Bååth, Special teacher with specialisation in mathematical development, graduate from Gothenburg University
Anders Eklund, Project manager, SPSM
Marthe Gjelstad, STEM specialist, NB
Lars Henrik Johansen, Senior adviser, Statped
J. M. Kvile, Math teacher, Statped
Sami Määttä, Planner, Celia
Björn Westling, Braille specialist, MTM

MYNDIGHETEN FÖR
TILLGÄNGLIGA MEDIER

Introduction and a short summary

This report is the result of a joint effort by the Nordic agencies for accessibility within the fields of media, literature, and education.

The agencies have an ongoing collaboration in development, following a vision of a complete digital environment in which STEM content is compatible and fully accessible. The STEM tech review project is one example of this collaboration. The agencies are aiming to develop the production and adaptation of teaching materials for all our target groups, including users of all ages, studying STEM on all levels. Target groups include persons with blindness, low vision, dyslexia, neurological disabilities, and motoric disabilities.

The report is presenting:

- Information on technical solutions that are working for all target groups, with an emphasis on persons with visual impairment or blindness.
- Areas in need of further study and research.
- Suggestions for development projects.

Among interesting results are the needs for the agencies to monitor and to contribute to evolving technologies and standards. The needs are pointed out for flexible solutions for writing math accessibly, where the user can choose in which way to write, and to make digital STEM content interactive.

Ten areas of interest are covered: Screen readers, Reading systems, Writing math, Calculators and graph programs, Multiline tactile displays, Graphs and diagrams, Programming interfaces, Learning platforms, Conversion tools, and Large Language Models.

Another outcome of the project is up-to-date user recommendations on hardware and software which make STEM content accessible.

The project has had a short timeline but has been very successful due to excellent competence and willingness to contribute by all partners.

Björn Westling, project owner

Table of contents

1. Background and scope

By Tim Arborealis Lötberg

1.1 Background

Science, Technology, Engineering and Mathematics (STEM) often relies on complex content such as mathematics, chemical formulae, graphs, diagrams and computer code. This can pose significant challenges to people with reading disabilities when engaging with the field. These challenges are prevalent throughout school (McCabe, 2023) and linger throughout higher education, demotivating many from pursuing careers in STEM.

New technologies are emerging which open up for greater accessibility in STEM. Additionally, an important incentive for publishers to produce accessible STEM books is the European Accessibility Act (European Comission, 2025), which includes e-books in its scope. Using current standards and applying principles of universal design, STEM books could be born accessible to a much greater degree. This, in turn, would lessen the need for retrospective adaptations and provide users with the literature they need on time.

While reading is an important part, it is not enough for books to be accessible for full participation in STEM to be possible. Users need to be able to author mathematic expressions, solve equations, program, create graphs and diagrams and communicate STEM with their peers to be able to work independently.

In the Nordic countries, users often face an additional barrier: even though tools for accessibility in STEM may be available, they are not always localised to the Nordic languages. The Nordic agencies for accessible media and education consist of MTM and SPSM (Sweden), NB and Statped (Norway), Celia (Finland), Nota (Denmark) and HBS (Iceland). Since our users face the same challenges, we collaborate on STEM accessibility.

STEM accessibility is a field undergoing rapid development, and to best serve our users' needs we need to keep up to date with the latest technology. Therefore, we performed this tech review during spring 2025, with the intent of getting an overview of what solutions exist, how accessible they are, and how compatible they might be with one another.

The review is divided into ten parts:

- Screen readers
- Reading systems
- Writing math
- Calculators and graph programs
- Multiline tactile displays
- Graphs and diagrams
- Programming interfaces
- Learning platforms
- Conversion tools

- Large Language Models

STEM technology is a huge field, and with limited time and resources we have had to limit the scope accordingly. For some areas we have had the possibility to go into details with user testing, while for others we have mainly identified gaps in our knowledge which need filling. Our findings have resulted in several suggestions for further studies as well as for development projects. In addition, we have compiled a set of recommendations for users on what solutions are available right now.

It is our hope that this report will be of interest not only to accessibility agencies, but also to producers and facilitators of learning materials, as well as STEM tech developers in the industry. Together we can work towards greater inclusion in STEM, so that everyone is free to choose career based on interest rather than limitations.

1.2 Glossary

AT – Assistive technology, i.e. hardware and software which help persons with disabilities function and participate independently. Includes e.g. screen readers and refreshable braille displays.

BRF – Braille Ready File, an ASCII-based file format for pre-formatted braille. Used widely in the USA and many countries across the world.

BVI – Blind or visually impaired.

CAS – Computer algebra system, i.e. mathematical software with the ability to manipulate mathematical expressions in a way similar to the traditional manual computations of mathematicians and scientists.

EAA – European accessibility act (European Comission, 2025).

IDE – Integrated Development Environment. A software application that provides comprehensive facilities for software development. An IDE normally consists of at least a source-code editor, build automation tools, and a debugger.

LaTeX – A software system for typesetting documents, including mathematical notation (LaTeX, 2025). It is commonly used for STEM publications.

LLM – a language model trained with self-supervised machine learning on a vast amount of text, designed for natural language processing tasks, especially language generation.

LMS – Learning management system. A software application for the administration, documentation, tracking, reporting, automation, and delivery of educational courses, training programs, materials or learning and development programs. A platform where the school makes the learning content and controls the system to some extent.

MathCAT – Math capable assistive technology (GitHub, 2025). A plugin tool for screen readers that renders speech and braille from MathML.

MathJax – software which displays MathML markup for a reading system or the web (MathJax, 2025). It has many settings for producing images of math, MathML code, invisible math expressions and invisible text for math for screen readers. These are hardcoded into the reader and not controlled by the user. MathJax is also a converter tool. This means that the author can write mathematics in LaTeX or MathML, and it will be rendered in the chosen output format.

MathML – Mathematical markup language (W3C, 2025) for describing mathematical notation and capturing both its structure and content. It is the standard recommended by W3C for processing mathematics on the web, just as HTML is for text.

Multiline tactile display – a device containing a display composed of pins raised through holes in a flat surface. It can be used for tactile graphics, multiline braille or both.

PEF – Portable Embosser Format, a Unicode-based file format for pre-formatted braille. Used primarily in Scandinavia.

Reading system – Software for reading e-books.

Screen reader – A form of assistive technology that renders text content into speech or braille output.

Sonification – The use of non-speech sound in an intentional, systematic way to represent information.

STEM – Science, technology, engineering and mathematics.

TTS – Text-To-Speech, technology for converting written text into spoken words using speech synthesis.

User – A person with a reading disability who has access to our agencies' products and services.

UX – User eXperience, how a user interacts with and experiences a product, system or service.

VLE – Virtual Learning Environment, where students study a digital-based curriculum taught by instructors that lecture online via video or audio. More directly aimed at learners than LMS, and the school does typically not make the content.

WCAG 2 – The Web Content Accessibility Guidelines. The WCAG 2.0 consists of 12 guidelines that include testable criteria for accessibility at A, AA, and AAA levels (W3C, 2025).

1.3 Acknowledgements

This review has truly been a team effort, and we are grateful for valuable input and support along the way from our reference group and steering committee at MTM, as well as from external referents. Special thanks go out to Eva Bengtsson, Darrin Evans, Jimmy Freiding, Alfreð Halldórsson, Andrew Flatres, Marie Ivarsson, Pontus Jeppsson, Stephen Simei Kimogol, Julia Lindberg, Martin Persson, Stacy Scott and Peder Wollinger.

2. Screen readers

By Anders Eklund

2.1 Introduction

A screen reader is a type of assistive technology software that enables people who are blind or visually impaired to use computers, smartphones, and other digital devices. It works by reading aloud the text displayed on the screen using synthesized speech and by converting it into braille, if the user has a refreshable braille display.

Screen readers also give the user a description of the semantic structure of web pages, documents, and applications, including text, buttons, menus, images (if they have alternative text), and form fields. Users navigate through the content using keyboard commands or gestures (on mobile devices), allowing them to interact with digital content independently.

2.2 Screen Readers and Mathematics

For a screen reader to be able to read out mathematical expressions, the expressions need to be coded in a format that the screen reader software can process. The recommended format for accessible maths is MathML, which also is the web standard for mathematics. It takes a lot more than being able to read out numbers and symbols correctly. The structure of a mathematical expression needs to be correctly explained to the user, using appropriate terminology. This requires a stringent markup of high quality and software that can parse and correctly interpret mathematical expressions. Not all screen readers have this. Some screen readers rely on third-party additions and some also rely on intermediate technology, such as MathJax, to handle MathML content.

2.2.1 MathJax

MathJax (MathJax, 2025) is an open-source JavaScript display engine designed to produce high-quality mathematical typesetting in web browsers. It allows authors to include mathematical content on web pages using standard markup languages such as LaTeX, MathML, and AsciiMath, which MathJax then renders with precision and accessibility.

MathJax works consistently across all modern browsers without need for additional plugins. It also has some accessibility features that can provide semantic information to screen readers.

2.2.2 MathCAT

MathCAT (GitHub, 2025) is an open-source software that converts MathML expressions into speech strings with embedded speech engine commands and into strings for output on refreshable braille displays. It also allows a user to navigate a mathematical expression and zoom in and out of the various layers in the expression.

The software was initially built to function as an addon to the open-source Windows screen reader NVDA. However, the software can also be built into other applications.

The software is designed to make translation into other languages or braille standards possible. At this point, available languages for speech are English, Spanish, Swedish, Finnish, Vietnamese, Indonesian and Taiwanese Chinese. Available braille options are Nemeth, UEB, LaTeX, ASCIIMath, Swedish and Vietnamese.

2.3 Screen Reader Testing

Testing was performed between March and May 2025 for the screen readers that I had access to. Other screen reader options were investigated through online research. Note that the tests concern screen readers, not text-to-speech tools.

Testing was done by loading an HTML file with MathML expressions into a browser (Chrome for Windows and Safari for iOS) and start the screen reader. A refreshable braille display was also connected. Another HTML file, which uses MathJax to render the MathML, was then opened and the output between the two files were compared.

2.3.1 JAWS

OS: Windows
Native MathML: Yes. JAWS still has its own MathML support, which works in English. MathCAT is in the "Upcoming functions" section and can be activated. It seems to work in English only and I can't access any settings for MathCAT. Hopefully this will be resolved soon.
Using MathJax: MathJax doesn't seem to make any difference.
Braille output: The current functionality has Nemeth output only.
Languages: Currently only English.

2.3.2 NVDA

OS: Windows
Native MathML: Yes. With MathCAT addon NVDA has support for native rendering of MathML.
Using MathJax: It works with MathJax but is not dependent on MathJax to get it to work.
Braille output: Yes. Nemeth, UEB, Swedish braille, LaTeX and ASCIIMath.
Languages: A selection of languages at this point, including Swedish and Finnish.

2.3.3 VoiceOver

OS: iOS
Native MathML: Yes. It seems that VoiceOver has some built-in support for

MathML, but the quality depends on the reading system. Safari seems to be the best option, but there are some issues. It is not as good as MathCAT. For instance, VoiceOver doesn't handle tabular math expressions as well as MathCAT does.
Using MathJax: Seems to be able to give a more reliable experience but needs to be tested.
Braille output: Not very good. The only support that exists is for UEB with English as main language. But it is still not complete.
Languages: English and Swedish were tested.

OS: MacOS
Native MathML: I do not have access to a Mac computer, but online research seems to suggest that MathML support is still unreliable, but slightly better than on the mobile OS. Still, Safari is the best choice of reading system.
Using MathJax: Seems to be able to give a more reliable experience but needs to be tested.
Braille output: Not very good. The only support that exists is for UEB with English as main language. But it is still not complete.
Languages: Not tested but should be the same as for the iOS version.

2.3.4 TalkBack

OS: Android
Native MathML: No. It can read numbers and symbols but has no semantic interpretation.
Using MathJax: No difference with MathJax.
Braille output: No useful output.
Languages: Irrelevant, since there is no MathML support.

2.3.5 Narrator

OS: Windows
Native MathML: No. Narrator has very limited possibilities and no actual MathML support.
Using MathJax: No difference with MathJax.
Braille output: No useful output.
Languages: Irrelevant, since there is no MathML support.

2.3.6 Orca

OS: Linux
Native MathML: Not tested. Online research suggests that it can read numbers and symbols but has no semantic interpretation. I have not found out if there are any third party addons that enable MathML support.
Using MathJax: Not tested.
Braille output: Not tested.
Languages: Not tested.

2.3.7 Dolphin Supernova and ScreenReader

OS: Windows

These products were not tested as I had no access to them. Also, online research suggests that Dolphin has integrated MathCAT into their reading application, EasyReader, and not in their screen reader. This suggests that using the Dolphin eco system still would enable reading maths, but only using their reading system, not globally.

2.4 Conclusions

The tests indicates that Windows and iOS/MacOS are the best OS choices for reading maths, as NVDA, JAWS and VoiceOver all have support for MathML. For braille readers the Windows options are the most reliable. For languages other than English, VoiceOver might still be an option, although not for braille output. For non-English braille output MathCAT seems to be the only actual option, but it may require translating it first, if the desired language option is not yet available. MathCAT also has the option of displaying LaTeX or ASCIIMath on a braille display, which are commonly used solutions for students who are braille readers.

If you aren’t depending on a screen reader for navigation and semantic overview of the screen content, but only need to have the content read aloud, a TTS software, such as ReadSpeaker could be an option. ReadSpeaker (ReadSpeaker, n.d.) has MathML support that is dependent on MathJax. This works in English, but there may be other language options.

When it comes to future development in screen reader support for mathematics, a lot depends on the big corporations, such as Microsoft, Apple, Google etc. But securing a future for a tool, such as MathCAT, is also important, as well as advocating for interested parties to make translations into more languages.

3. Reading systems' support for MathML

By Sami Määttä

3.1 Background

As a part of the STEM tech review, it was investigated how the MathML markup is rendered in reading systems. Many (if not all) reading systems are built upon different browser engines, so some conclusions can be drawn from which a specific reading system has been built from.

The caveat is that from the moment a browser engine is adapted for a reading system, they start to diverge, and the developers of the reading systems need to update the underlying engine to match the latest development of the browsers. Users might expect similar accessibility and usability in reading systems as they do on the web in a browser.

Based on this review, it can be said that the reading systems seem to lag behind the most widely used browser engines Chromium [Google Chrome, Windows Edge] (Chromium, n.d.), Gecko [Mozilla Firefox] (Gecko — Firefox Source Docs documentation, n.d.), Webkit [Safari] (WebKit, n.d.)) and conclusions can't be drawn from just the knowledge of the engine. For many reading systems the engine is also not known.

3.2 Research question

How do different reading systems support the visual rendering of MathML markup?

3.2.1 Why focus only on visual rendering?

Most digital book (e-books, talking books) formats are made of XHTML or HTML files. The MathML markup present in the files is exposed to the user in the book even if it is not rendered visually. This means, for example, a user using MathCAT with a screen reader, would still be able to access the mathematics of the MathML markup (GitHub, 2025). Blind users using screen readers would have an acceptable experience using a reading system without the visual rendering of MathML markup. However, the e-book won't be accessible for users with other print disabilities. Some examples of scenarios, where visual rendering of MathML markup is needed:

- A user requiring zooming of math content. This user might have a visual impairment but not be blind or use a screen reader. If MathML was invisible or poorly rendered, it would be useless to the user.
- A user with dyslexia, or challenges in perception, when they are reading an e-book with a synthetic voice. They would hear the mathematical equation, but not see the equation at the same time,

which would cause confusion. Same goes for a sighted person using a screen reader.

Anybody else with or without disabilities would also have a subpar experience with a mathematics e-book that doesn't visually render the MathML markup correctly.

3.3 Research design

3.3.1 Chosen test e-book

An e-book with mathematics was prepared to use in testing. The e-book can be downloaded though The e-book can be downloaded from Google Drive through this link (Google Drive, 2025). It is based on the Accessibility Test Mathematics v.1.1.1 (epubtest.org, 2025).

The test e-book is a full-text e-book and it has no audio. It is written in English, but it makes no difference to the results of the testing.

3.3.1.1 Modifications to the test e-book

The scope of tests in Epubtest.org was not sufficient for the purposes of this review (epubtest.org: Test visual-550 in Visual Adjustments (2.0.0), n.d.). That is why more MathML elements, which are known to potentially cause issues in visual rendering, were added. The problems can be noticed when the typesetting and placement of elements inside the math elements behave oddly.

Especially the elements that stretch math expressions vertically or have specific typesetting can cause issues with visual rendering. These can be for example the index in a root element, a strike-through in an enclosed element or having the enclosed parentheses stretch vertically around a matrix.

Such added math elements include:

- roots (<msqrt>, <mroot>)
- large operators
- fractions (<mfrac>)
- enclosures (<menclose>, which is not supported in MathML Core, but is supported in MathCAT) (MathML Core, n.d.)
- table-like expressions (<mtable>) such as a piece-wise function, equation solving and matrices.

3.3.2 Criteria for defining support for visual rendering of MathML

There were four labels a reading system could receive that described the support for visual rendering of MathML markup. The following labels have a clear hierarchy:

1. Native support for MathML
2. Partial support for MathML
3. No support for MathML.

Outside of this hierarchy is the use of MathJax, which displays MathML markup for the reading system (or the web) (MathJax, 2025). It can be thought of as a separate program that handles the MathML markup for the reading system.

MathJax has many settings for producing images of math, MathML code, invisible math expressions and invisible text for math for screen readers. Customization possibilities for MathJax are great, but this is not controlled by the user of a reading system. The settings are determined by the developers. This can result in clashes with the assistive technology being used.

For example, MathJax can output invisible text for describing the mathematics, but the language of the text might be different from what the user wants.

MathJax is also a converter tool. This means that the author can write mathematics in LaTeX or MathML, and it will be rendered in the chosen output format.

3.3.2.1 Native support for MathML

If the support is on par with current browser support, then the reading system is labelled to have native MathML support. This includes support for

- Basic visual rendering: simple one-line expressions, roots and subscripts and superscripts.
- Visual rendering of "2D mathematics" such as fractions.
- Visual rendering of multiline mathematics such as <mtable>, which is used for expressions like piece-wise functions, matrices and equation solving.
- Support for enclosures is not required. (Chromium-based browsers don't render enclosures.)

Current browser versions support the MathML Core specification quite well. There are visual differences between browsers, but they are not major. For example between Mozilla Firefox (Gecko) and Google Chrome (Chromium) the spacing between elements differ: Firefox’s rendering is more compact, and Chrome’s has more space between characters. Chrome does not support the strikethrough element (enclosure), but Firefox does.

3.3.2.2 Partial support for MathML

If **only one** criterion in "native MathML" is **not met.**

This means that the reading system has support for multiple math elements, but the support is missing for one element. This way the reading system can be useful, but not ideal.

3.3.2.3 No support for MathML

If **more than one** criterion in "native MathML" is **not met.**

This means that a reading system can have support for individual math elements but still be categorized as having no support for MathML.

3.3.2.4 Support for MathML by MathJax

If it can be ascertained that MathJax has been used. This decision is based on personal experience or knowledge of the reading system.

MathJax also provides the support for the enclose ("strikethrough") element and because it works in all platforms that can run JavaScript, it is a good solution for MathML support.

3.3.3 Tested reading systems

The reading systems that were tested are listed below with the test results. The reading systems were based on a list on epubtest.org and suggestions from other reviewers of the STEM tech review team.

The tests include reading systems on the desktop, mobile and web (web readers).

The tests were done in different language versions of the reading systems, but it makes no difference to the results of the testing.

3.3.4 Limitations

No testing has been done on MacOS or Linux operating systems, because there was no access to them.

The mobile operating system used for testing was mainly Android. Some reading systems on mobile were tested with iOS when it was possible.

There were some reading systems that couldn't be installed on a workstation, so they are not in the testing pool. They have not been specified.

No e-ink reading devices were tested.

3.4 Results

Below are the test results for different reading systems. Also included is information on version of the tested reading system (when available), the supported platform(s) for reading systems, operating system that the reading system is available for, and lastly some notes about the test and support.

A question mark (?) has been added to some reading systems' operating system(s) and platform(s) if there was no knowledge of which are supported.

3.4.1 Thorium Reader

(Thorium Reader, n.d.)

Tested version: 3.0

Available on platform(s): Desktop

Available on operating system(s): Windows, Linux, MacOS

Testing result (label): MathJax (only tested on Windows).

Notes: Thorium Reader is based on Chromium, so it could be possible to have native MathML support when EDRLab (the developers) decide to upgrade the version of Chromium they use. Now it requires MathJax.

EDRLab is also developing a web reader, which relies on the browser's engine to render MathML. They are also developing a mobile app, but it is unclear what the level of support in general for MathML would be.

3.4.2 Calibre e-reader

(Calibre - E-book management, n.d.)

Tested version: 7.22

Available on platform(s): Desktop

Available on operating system(s): Windows, Linux, MacOS

Testing result (label): MathJax (only tested on Windows).

Notes: Calibre can be used on mobile to manage books, but no mobile reading app exists.

3.4.3 Adobe Digital Editions

(Adobe Digital Editions, n.d.)

Tested version: 4.5.12.112

Available on platform(s): Desktop

Available on operating system(s): Windows

Testing result (label): MathJax

Notes: There are issues in rendering roots' indices in the correct position.

3.4.4 Dolphin EasyReader

(EasyReader App | Dolphin Computer Access, n.d.)

Available on platform(s): Desktop, mobile.

Results are listed below for each platform.

3.4.4.1 Desktop

Tested version: 11.0.5 build 622

Available on operating system(s): Windows, Linux, MacOS

Testing result (label): Native MathML (only tested on Windows).

Notes: No support for `<mspace>`, which can be used to create space between math elements.

3.4.4.2 Mobile

Tested version: 11.05 build 825

Available on operating system(s): Android

Testing result (label): Not tested

Notes: Crashes on launch, so no testing could be done.

3.4.5 Colibrio Reader

(Colibrio Reader - Colibrio Reader, n.d.)

Available on platform(s): Web reader, mobile app.

Results are listed below for each platform.

3.4.6 Web reader

Tested version: 250204-1455

Testing result (label): Partial support

Notes: There is no spacing between elements in matrices. The web reader doesn't (only) use the browser's engine to render MathML, since then the expected support should be on the same level as the browser's. It is difficult to know why this is the case without contacting the developer.

3.4.7 Mobile app

Tested version: 1542

Available on operating system(s): Android

Testing result (label): Partial support

Notes: Simple roots and fractions display well. If the root's index is a two-digit number, then the positioning of the number causes rendering issues where the different parts of root notation overlap. No support for stretched parentheses in multiline math.

The mobile app uses mobile Chromium as its base. There were some bugs in the rendering of MathML that are known to the Chromium team based on Colibrio's developer's comments.

3.4.8 Pratsam Reader

(Pratsam Reader App – Pratsam, n.d.)

Tested version: 5.11 (292)

Available on platform(s): Mobile

Available on operating system(s): Android, iOS

Testing result (label):

- Partial support (Android)
- Native MathML or MathJax (iOS)

Notes: Pratsam Reader uses Colibrio for its EPUB reader. Some bugs are due to mobile Chromium, which is used for mobile Colibrio's engine. iOS's mobile app has better support for MathML than the Android version. It can't ascertained which method is used to render MathML in iOS without contacting the developer.

3.4.9 Lithium: EPUB reader

(Lithium: EPUB Reader – Google Play, n.d.)

Tested version: 0.24.6.1

Available on platform(s): Mobile

Available on operating system(s): Android

Testing result (label): No support

Notes: The app has support for fractions, but not any of the other elements: roots, exponents or matrices. This reader might be based on Readium.

3.4.10 Cantook by Aldiko

(Cantook by Aldiko – Google Play, n.d.)

Tested version: 1.12.2

Available on platform(s): Mobile

Available on operating system(s): iOS, Android

Testing result (label): No support (only tested on iOS).

Notes: Fractions were rendered, but otherwise there was no support.

3.4.11 Kobo Books

(Kobo | Rakuten Kobo, n.d.)

Tested version: No version number recorded.

Available on platform(s): Mobile

Available on operating system(s): iOS, Android

Testing result (label): No support (only tested on iOS).

3.4.12 Legimus

(Legimus, n.d.)

Tested version: Not tested

Available on platform(s): Web reader, mobile

Testing result (label): Not tested

Notes: Web reader and mobile app, which is provided by MTM to their users. New web reader and mobile app are being developed. Both are based on Readium, which is used in Thorium Reader as well. As was mentioned in the Thorium Reader subsection, the web reader will rely on the browser's support for MathML. The mobile app's future support for MathML is unclear.

3.4.13 RedShelf

(RedShelf, n.d.)

Tested version: Not tested

Available on platform(s): Mobile, ?

Available on operating system(s): Android, ?

Testing result (label): Not tested

Notes: Account couldn't be created on mobile.

3.4.14 Nota Bibliotek 2.0

(Nota Bibliotek 2.0 app | Nota bibliotek, n.d.)

Tested version: Not tested

Available on platform(s): Mobile

Available on operating system(s): Android, iOS

Testing result (label): Not tested

Notes: The Nota app is used in Denmark, and it is provided by Nota to their users. It is based on Readium, which is the same system that Thorium Reader uses. As was mentioned in the Thorium Reader subsection, the mobile app's future support for MathML is unclear.

3.4.15 Book Share

(Bookshare, n.d.)

Tested version: Not tested

Available on platform(s): Mobile, ?

Available on operating system(s): Android, iOS

Testing result (label): Not tested

Notes: No e-books could be manually added to the app on Android. iOS was not tested.

3.4.16 Play Books

(Google Play Books, n.d.)

Tested version: No version number could be found.

Available on platform(s): Mobile

Available on operating system(s): Android

Testing result (label): Not tested

Notes: The test e-book couldn't be transferred to the app.

3.4.17 Clusive

(Clusive, n.d.)

Tested version: No version number could be found.

Available on platform(s): Web reader, ?

Testing result (label): No support or not tested

Notes: The test e-book could be uploaded to the web reader, but it doesn't display files with MathML in them and throws an error. It is unclear why this happens, so it has been labelled with no support as well as not tested.

3.5 Conclusions

The results on support of visual rendering of MathML are presented in the table below.

Table 3-1: Results on support of visual rendering of MathML.

Support for TathML	Number of reading systems
MathJax	3
Native support for MathML	2
Partial support for MathML	3
No support	4

Not tested (6 reading systems): Legimus, RedShelf, Bookshare, Nota app, Play Books and Clusive.

3.5.1 Best support when using MathJax

The best support for visual rendering for MathML markup comes from using MathJax. Mathjax is used in

- Thorium Reader
- Calibre e-reader
- Adobe Digital Editions.

MathJax still produces the highest quality of mathematics rendering in reading systems. Combined with the lack of support for native MathML, MathJax seems to be the best choice for rendering mathematics.

It is hard to find out which settings each reading system uses without contacting the developer directly.

For some reason Adobe Digital Editions still had challenges in displaying root elements indices in the correct position and place. This should be handled by MathJax, so it could be that the reading system doesn't use it or that there is something else going on with the rendering.

3.5.2 Native MathML support is rare

Only two of the tested reading systems seemed to have (near) support for native rendering of MathML markup:

- Dolphin EasyReader on desktop
- Pratsam Reader mobile app for iOS.

It is not clear if Pratsam Reader mobile app for iOS uses MathJax without contacting the developer.

Even if the reading system qualified for “native support for MathML”, it still could be spotty in places and not the highest quality compared to MathJax or the Gecko-based browser engines.

No reading system that was tested can be recommended for native visual rendering of MathML markup. It became clear that even when there seems to be native support for MathML, there might be some elements that are not rendered correctly.

More testing should be done with EasyReader (desktop) to see which math elements it supports and how this might change in the future.

3.5.3 Partial support

Partial support for visual rendering of MathML markup was in

- Colibrio Reader web reader
- Colibrio Reader mobile Android
- Pratsam Reader on mobile Android.

Reading systems with partial support can’t be recommended for use, since the user can never know which parts of MathML are supported.

3.5.4 No support is the most common in mobile apps

No support for visual rendering of MathML markup was in

- Apple Books on mobile iOS
- Lithium: EPUB reader on mobile Android
- Cantook by Aldiko on mobile iOS
- Kobo Books on mobile iOS.

“No support” might still mean some support, but it is very unreliable. At most it can mean that simple operations such as plus, minus, times and divided by are supported. However, many times anything that requires other symbols, such as the root or fraction line, is not displayed at all.

No patterns emerged from different tests as to which elements are supported in these mobile apps. This might suggest that many mobile apps use different engines or different versions of the same engines.

3.6 Discussion

3.6.1 Next steps

1. MathJax provides the best visually rendered experience, but it might hinder the experience of using assistive technologies, because MathJax has many settings that have been set by the developers of reading systems. Users can’t change these settings on their own. MathJax is used in Thorium Reader, Calibre e-reader and Adobe Digital Editions.

Next steps: Encourage reading system developers to implement native MathML support and make MathJax customization controller by the user or the settings very lightweight.

2. Mobile apps have poor support for MathML.

   Next steps: Reading system developers should be encouraged to implement native MathML support for mobile apps. EDRLab is developing a mobile app based on Readium and they should be encouraged to implement MathML support on par with Chromium-based browsers, since this is what their app will most likely be based on. Colibrio should be encouraged to improve the mobile app's MathML support.

3. Native MathML support should be the end goal of all reading systems.

   Next steps: The agencies should continue to encourage usage of MathML in web and e-book environments. Native MathML works best with assistive technologies now and in the future, but it is less versatile in usage than MathJax.

3.6.2 In-depth discussion

In short it can be said that the visual rendering of MathML markup is not uniform across different reading systems.

The best support for MathML markup comes from using MathJax, since it works across platforms if it is implemented. It is a problem that the user doesn't know how MathJax has been set up and the user can't change these settings by themselves.

There is some native support for MathML markup in desktop and web apps, but even this doesn't mean that the reading experience would be the smoothest. Since web readers are used in a browser, it was assumed that the MathML support would always be native support. This was proven incorrect by Colibrio's web reader. This means that using a web reader doesn't guarantee a browser's level of support for MathML markup.

The mobile reading systems seem to fare the worst when it comes to MathML markup. In general, the mobile apps either have partial support or no support at all, and they should be improved in the future. Until then, the desktop apps should be recommended for use.

3.7 Further research

More testing needs to be done to find out what exactly is the scale for visual rendering MathML support in reading systems.

The test e-book should have more rigorous and systematic representation of different math elements, since during testing it was found out that some elements, that were should have been supported, were not (such as the <mspace> in desktop Dolphin EasyReader).

Maybe in the future it could also be considered how the visual rendering of MathML conforms to the relevant WCAG 2.2 criteria (W3C, 2024).

With more resources, different reading systems could be tested if they would require licenses or subscriptions. The need for testing in different operating systems is apparent.

Epubtest.org continues to test for mathematics in EPUB books, but the tests are very limited (epubtest.org: Test visual-550 in Visual Adjustments (2.0.0), n.d.). A notable difference to this review is that epubtest.org also tests zooming of mathematical equations.

3.7.1 Note on the review

All the tests that were planned have not been done. Notable misses on the tests: Legimus and Nota’s apps.

4. Writing Math

By Lisa Hartun Bennedsen

4.1 Introduction

The following explores how blind and visually impaired students write and access mathematics in both the Danish educational system and broader international contexts.

The research seeks to answer the following questions: *How do BVI students in Denmark and abroad write mathematics in practice? What assistive technologies, workflows, and support systems are currently enabling or obstructing their participation in mathematics education?*

In Denmark, students often rely on familiar tools from primary school such as Word with WordMat, and technologies like GeoGebra, TI-Nspire, or Maple - with varying degrees of accessibility.

Internationally, students adopt a range of approaches depending on local infrastructure, institutional expectations, and available training, often combining LaTeX, ASCII math, screen readers, braille displays, and audio-based tools like Desmos or EquatIO.

For the purposes of this study, a **screen reader** is defined as software that reads aloud digital content or sends it to a braille display (e.g., JAWS, NVDA, VoiceOver), while a **reading system** refers more broadly to any combination of software, hardware, and human or procedural support that enables blind students to interpret and produce mathematical content.

The study highlights the lack of universally accessible tools and emphasizes the importance of flexibility, institutional awareness, and student-driven adaptation in overcoming structural barriers in math education.

4.2 Blind Students in Denmark

This report is based on insights gathered through ongoing, dialogic engagement with 14 participants. These conversations have taken place over time and in a variety of informal and semi-structured contexts, including follow-up discussions and reflective exchanges. While not conducted as formal interviews or surveys, these dialogues have enabled a rich and iterative understanding of the participants' experiences. All participants are referred to as "students", even if not currently enrolled in an educational program.

Of the 14 participants, 2 blind participants are enrolled in higher education institutions, focusing exclusively on STEM subjects (fields of study are not specified to ensure greater anonymity). 6 have attended upper secondary education. 6 have been enrolled in various higher education programs and have completed at least one full year of studies (60 ECTS points).

4.2.1 Challenge

Most of the students report that they have learned mathematics in a way that makes sense **to them**, but **not necessarily to others**. When they need to submit assignments or take exams, they rely on a secretary or support person who "translates" their math into standard notation.

Many describe situations where they were able to solve a task with no difficulty, but because they accidentally entered some "gibberish" without noticing, they received a lower grade or were perceived by teachers as "less intelligent" - sometimes even as "mildly mentally impaired."
Experiencing this a few times leads many to feel that "it's just more comfortable to have someone check for typos."

Most students continue to use the software they learned in primary school, or the software required in upper secondary school. Many of them make extensive use of Excel for creating tables and graphs, while a teacher, secretary, or support person assists with the visual setup of the graph.

4.2.2 Primary School: WordMat, GeoGebra, MatematikFessor

WordMat (GitHub, 2025) can be read aloud, but the student must click the field to trigger the screen reader.
Therefore, blind students typically write their math in Word using "their own language" and do calculations on the computer's calculator, which is navigable with NVDA/JAWS and a braille display.

GeoGebra (What is GeoGebra?, u.d.) can be navigated and used to generate graphs but generally requires a good friend to assist with navigation, as it can be confusing or overwhelming.

Desmos (Desmos accessibility, n.d.) allows graphing and even displays input in Nemeth Braille. Once a graph is created, the "graph tracing menu" can be activated with Alt+T.
The user can scroll through the graph using arrow keys while the screen reader reads out the coordinates. Pressing Tab jumps between "points of interest" such as axis intersections or extrema.

The highlight, however, is pressing H, which plays the graph from left to right using pitch to indicate function values (low pitch for low values, high for high). Variations in sound characteristics (volume, background noise) further indicate whether a point is above or below the x-axis or to the left or right of the y-axis. Desmos is only used in higher education in Denmark, unless a blind student has managed to "push it through" with the teacher.

MatematikFessor is Not Fully Accessible

MatematikFessor (MatematikFessor, 2025) is a very visual platform. Many elements (task descriptions, buttons, graphs) lack appropriate alt text or have poorly structured ARIA roles, making navigation difficult or impossible.

Math is often rendered as images, not as semantic text (like MathML or LaTeX), which makes it unreadable by screen readers - this includes fractions, exponents, and symbols.

Interactive exercises such as drag-and-drop, multiple-choice, and click-based tasks are inaccessible without sight, especially when answers aren't read aloud or detectable by a screen reader. Keyboard navigation is often inconsistent.

MatematikFessor is used extensively in primary school, from grade 0 to grade 6.
This undermines blind students' independence, as they need support staff to help them "solve the tasks correctly."

4.2.3 Biggest Challenge: VLE:s in Primary School

These platforms often don't respond to screen readers. One student shared:

"The other day he had an English test, but he couldn't complete the grammar part, because the screen reader couldn't read the tasks coherently. So he had to submit only part of the test and email it to the teacher.
'There's not much to be done about it. I can get as frustrated as I want, but it doesn't help. It's better to focus on how I can do the task anyway,' says X, whose teachers are understanding and cooperative in finding alternative solutions. Even so, X is dependent on his secretary, who translates the tasks into an accessible format.
'Teachers upload assignments in OneNote, which isn't very screen reader friendly. It's hard to navigate and find the files. So my secretary receives the assignments and converts them into a readable format, then sends them to me. He's basically a go-between for me and my teachers,' says X."

4.2.4 Conclusion: Writing Math in Primary School

Students focus on solving the problems correctly, while someone else translates them into "standard math."
They often skip visual or graphical elements and focus on what they're good at. This can lead to lower grades, simply because they didn't have the time or tools to handle the graphical tasks.

4.3 Upper Secondary School Tools

4.3.1 TI-Nspire, Maple, Math Platforms, and CAS Tools

The TI-Nspire CAS graphing calculator (Texas Instruments, 2025) is a robust, hands-on learning tool that supports math and science instruction from middle school and further. In Denmark, it plays a central role in General Upper Secondary Education Programmes (STX) and Higher Technical Examination Programmes (HTX). Students typically use both the handheld calculator and the TI-Nspire Computer Software, which mirrors the calculator interface, for a wide range of tasks including algebra, graphing, statistics, and equation solving.

In HTX, and to some extent in STX, students also work with Maple (Maple, n.d.), a powerful Computer Algebra System (CAS) well-suited for symbolic and numeric mathematics. It is especially valued for its capabilities in modelling, solving complex equations, and exploring advanced mathematical functions.

GeoGebra remains a popular choice across upper secondary schools, particularly for visualizing geometry, functions, and data. Its free availability as both an app and web-based tool makes it widely accessible for interactive learning.

To support classroom instruction and independent study, platforms like Matematikbogen.dk, Restudy, and Ma.fi offer structured assignments, interactive exercises, and explanatory videos. In addition, students frequently supplement their learning with online CAS tools such as WolframAlpha (Wolfram Alpha, 2025) and Symbolab (Symbolab, 2025) for quick problem solving, differentiation, and equation handling.

4.3.2 Maple Is Clearly Better for Visually Impaired Students.

Maple is text-based, while TI-Nspire is heavily dependent on visuals - small icons, menus, drag-and-drop, and pop-ups.
Keyboard navigation in TI-Nspire is often confusing, and screen readers can't interpret math correctly.

4.3.3 Comparison between Maple and TI-Nspire

Feature	Maple	TI-Nspire
Screen reader support	Partially possible	Very limited
Keyboard navigation	Support	Limited and illogical
Math as text (e.g., MathML)	Can be exported and edited	Often unreadable
CAS access and interactivity	Scriptable and accessible	GUI-heavy and visual
Real use by blind students	Used in higher education	Nearly unusable without sight
Support for assistive tech	Third-party integration possible	No official support

In Maple, you can enter commands like:
```
diff(x^2 + 3*x, x);
```

And get a text-based answer that the screen reader can read aloud.
This makes it possible to write and understand math without visual interaction.

Maple works relatively well with screen readers. While not perfect, it allows keyboard-only navigation, reading math as text (not images), and exporting results in screen reader-accessible formats.

Blind students can use Maple with JAWS or NVDA for speech output, a braille display (if they read braille), keyboard shortcuts instead of menus, and export to MathML or plain text. It allows for conversion to braille.

4.4 Higher Education

In STEM higher education programs in Denmark, LaTeX-based math is always used.

Most blind students have a decent foundation in LaTeX if they're interested in math, but up to C-level math in high school, only a few elements are commonly used: \frac{}{}, \sqrt{}, ^, _, and * (for multiplication). Only those with an active interest in math will expand their "LaTeX vocabulary."

All science and engineering degrees require A-level math from high school. In higher education, students use whatever already works - unless they're introduced to something smarter and faster.

Only two of the students openly stated that they routinely use Snapchat (SnapChat, 2025) or AI (mainly ChatGPT, but sometimes ChatGPT would spit out such obscure and strange values that the students also tried Copilot and Gemini (Barda, Jensen, & Singh, 2023)). They used Snapchat or AI to find the solution first and then reverse-engineered the process. Snapchat is widely used for this purpose when students work on math problems independently.

4.4.1 How Snapchat Can Help Solve a Math Equation

Snapchat has a built-in feature powered by **Scan** and **AI tools**, including integration with **Photomath** (Photomath, 2025), a math-solving app. Here's how it works:

Step-by-step:

1. Open Snapchat on your phone.
2. Point the camera at a math problem (handwritten or typed).
3. Tap the Scan icon (usually appears as a small square or magnifying glass).
4. Snapchat will analyze the math problem and show a solution suggestion.
5. It may link to Photomath, which gives: The final answer, Step-by-step solution and an explanation of the method used.

What it can solve:

- Basic arithmetic (e.g., 42 divided by 6).
- Algebraic equations (e.g., 2x+3=11).
- Fractions, square roots, linear equations.

Limitations:

- Doesn't handle very complex math (e.g., integrals or matrices).
- Might misread messy handwriting.
- It's a quick-help tool, not a full math tutor.

4.4.2 Request from BVI Danish High School Students

Just like students with physical disabilities (e.g. blindness, amputation, arthritis, or paraplegia) can be exempted from physical education, students with blindness or severe visual impairment should be eligible for dispensations in math.

Why? It's extremely hard for a blind student to learn math symbols and their meaning. Math is a visually oriented subject, where graphs, geometry, and symbolic notation are central.

There are currently no tools or methods that allow blind students to access math on anything resembling an equal footing. Screen readers struggle to correctly pronounce mathematical notation.

The subject includes graphs and geometry, which must be observed, interpreted, and produced by the student.

Math is especially difficult to communicate and understand purely verbally, without visuals or written content.
Any verbal interaction around math is usually accompanied by written notation.

From visual and special education consultants' perspectives, exams should be designed so that blind students aren't tested on visual tasks.
This includes film analysis (where mood and lighting must be interpreted) or math tasks requiring graph interpretation, as the only solution.

4.5 Authoring Math Accessibly: Overleaf and Alternatives

When it comes to writing mathematics independently and accessibly - especially in higher education and professional contexts – Overleaf (Overleaf, 2025) is frequently cited as a preferred tool among blind and visually impaired students. This section draws on two sources:

1. Direct observations from the Danish student participants, particularly those enrolled in STEM programs at the university level.
2. Supplementary desktop research, used to explore international alternatives and fill gaps where student experience was limited or inconclusive.

Overleaf was used by students consistently and independently, primarily due to its cloud-based interface, collaborative features, and compatibility with screen readers such as JAWS and NVDA However, students also highlighted challenges, including the inaccessibility of the live preview pane and difficulties troubleshooting LaTeX errors non-visually (e.g., missing braces or unmatched commands).

To better understand the broader ecosystem of accessible math authoring tools, a targeted desktop review was conducted to identify additional LaTeX editors and alternative environments. These tools were identified through accessibility community forums, research publications, and assistive technology resource pages (e.g., W3C Math Accessibility Task Force, 2023; American Printing House for the Blind, 2022). The following were examined, though not reported by any of the Danish students in this study:

- TeXstudio, TeXworks, and WinEdt: Offline LaTeX editors with customizable interfaces and some potential for screen reader integration when properly configured (Pimentel & Freitas, 2019).
- Visual Studio Code with LaTeX Workshop Extension: A widely used, customizable text editor that supports LaTeX via extensions. While accessible in theory, it requires technical skill and comfort with extensions and terminal commands (Bigham et al., 2020).
- Markdown editors with MathJax/KaTeX: Used in lightweight or collaborative contexts for rendering inline math, particularly in educational blogs and note-sharing platforms (Smith & Miesenberger, 2021).
- Emacs with AUCTeX and speech output extensions (e.g., Emacspeak): A highly customizable option used by a small community of expert users (Raman, 2021), though its steep learning curve and niche status limit broader adoption.

These tools provide additional insight into the variety of environments where blind users may write or edit mathematical content, especially outside of formal education. However, their accessibility varies widely depending on user skill level, local support, and platform configuration.

4.5.1 Overleaf

Overleaf is a cloud-based LaTeX editor widely used in academia, especially in STEM fields. Its popularity among blind students comes from a few key features:

- Code-based input: LaTeX is purely text-based, meaning students can write complex math without needing to interact with visual symbols or layouts.
- Screen reader compatibility: The editor itself is mostly accessible with screen readers like JAWS, NVDA and VoiceOver, though improvements are still needed, especially in the live preview pane.
- Collaboration: Students can write in LaTeX and have teachers or sighted peers review and edit the same file - without translation steps.
- Standardization: LaTeX is the accepted format in most higher education STEM programs, meaning blind students using Overleaf are already working in the required academic format.

However, some challenges remain:

- The live PDF preview pane is not accessible.
- Syntax errors (missing brackets or commands) can be difficult to troubleshoot with a screen reader alone.
- Overleaf requires a stable internet connection and a degree of familiarity with LaTeX packages and document structure.

Despite this, Overleaf is one of the most-used platforms globally for blind students studying STEM subjects, particularly in countries with strong academic traditions in LaTeX.

4.5.2 Alternatives

Desktop LaTeX Editors

TeXstudio (TeXstudio, 2025), TeXworks (Sourceforge, 2025), and WinEdt (WinEdt, 2025) are offline LaTeX editors that offer greater control over document structure and output, which can benefit screen reader users in some cases. These editors may allow more customized navigation and speech output compared to web-based alternatives. However, they typically require users to:

- Install a LaTeX distribution such as MiKTeX or TeX Live manually
- Configure the editor for accessibility
- Navigate local file systems and manage document compilation manually

Because of these technical requirements, these tools are better suited to users with prior experience or strong support systems.

Visual Studio Code (Visual Studio Code, 2025), when paired with the LaTeX Workshop extension, offers a powerful and flexible environment for writing LaTeX. With the right configuration, it can work well with screen readers and supports user-defined shortcuts and settings. However, this setup is generally recommended for advanced users, as it demands a high degree of computer literacy and the ability to troubleshoot and maintain a custom workflow independently.

Markdown with MathJax or KaTeX

In more lightweight or web-based environments, blind students sometimes use Markdown (Markdown Guide, 2025) with embedded LaTeX-style math (rendered via MathJax or KaTeX). This allows:

- Authoring math in plain text
- Dynamic rendering in screen-reader friendly formats
- Use within LMSs or shared notes systems
- Platforms like Notion (Notion, 2025), HackMD (Markdown Guide, 2025), and some learning platforms support this method.
- Emacs (GNU, 2025) with AUCTeX and Speech Extensions

Advanced users sometimes use Emacs with speech output extensions (like Emacspeak), offering full control over LaTeX writing and reading. However, this solution is extremely niche and requires a steep learning curve.

Usage Trends

Overleaf is used in Europe, North America, and parts of Asia in academic contexts. It's less common in secondary education or in countries where LaTeX is not part of the academic tradition.

In resource-limited environments or early education, students are more likely to use plain text, email, or dictation methods rather than formal math authoring tools.

Among blind university students in math or physics, LaTeX (often via Overleaf) is the most standardized and empowering tool available – but its accessibility depends on prior training and institutional support.

4.6 Mobile math authoring tools and Audio-Based Math Input systems

While desktop tools like Overleaf or Maple are widely used in structured academic settings, mobile authoring and audio-driven math entry are becoming increasingly relevant - especially for blind students who need to take quick notes, participate in class, or study on the go. These tools aim to reduce friction by leveraging touch, voice, and simplified syntax.

4.6.1 Audio-Based Math Entry Systems

Dictation to LaTeX/Math Notation. Some apps and voice systems allow users to dictate mathematical expressions, which are then converted into LaTeX, MathML, or rendered as visual math notation.

EquatIO (Voice Input Mode)

Allows dictation of math expressions using plain English (e.g., "x squared plus three x plus two") which the app translates into structured math.

Integrates with Google Docs and Chrome. Output can be read by screen readers or exported as LaTeX.

4.7 Writing Math as a Blind Student: An International Perspective

Blind and visually impaired students encounter many of the same structural barriers as those documented in the Danish context. However, the ways in which they write and access mathematics vary significantly depending on regional infrastructure, available assistive technology, institutional support, and educational philosophy. There is no single method that works for all; rather, students piece together workflows using a patchwork of tools and strategies.

4.7.1 Text-Based Input

In higher education - particularly in mathematics-heavy fields - many blind students use LaTeX to write mathematics. This method provides precise control over mathematical notation, works well with screen readers and braille displays, and is fully compatible with scientific publishing tools.

For example, a derivative might be written as:

```
\frac{d}{dx} \left( x^2 + 3x \right)
```

This format is both semantically clear and screen reader friendly when properly supported.

Alternatively, some students use simplified or ASCII-style math input, especially in informal contexts like note-taking or messaging. For instance:

```
d/dx (x^2 + 3x)
```

While this plain-text approach is easier to type and often sufficient for basic communication, it lacks the structural detail and semantic richness of LaTeX or MathML. As a result, it may be less predictable or ambiguous when parsed by screen readers - particularly in more complex expressions.

In general, LaTeX provides better accessibility and consistency in STEM settings, while ASCII-style syntax serves as a lightweight alternative where full LaTeX is impractical.

4.7.2 Accessible Digital Tools and Math Editors

Some modern math editors support screen readers or braille displays to varying degrees:

EquatIO (Oribi, 2025) allows students to type or speak math aloud, offering real-time audio feedback and integration with screen readers.

MathML-based environments (used in some LMS systems) can be read aloud or interpreted with assistive software, especially when rendered properly via MathJax.

MathCAT, a newer open-source library, is being developed to make math more accessible across environments by offering improved speech output and navigation.

In practice, students rely heavily on whether the editor supports keyboard-only navigation and exposes math as semantic text rather than images.

4.7.3 Braille Displays and Math Braille

Where available, students use refreshable braille displays in combination with math braille codes. The most widespread standards are:

- Nemeth Code (used in the United States)
- UEB Math (Unified English Braille)
- Marburg (used in parts of Germany)
- Kanji Braille extensions (in some East Asian countries)

However, it's important to stress that no universal math braille system exists, and many countries lack support entirely. Even where standards exist, they are not always taught or supported at scale. In multilingual or post-colonial countries, math braille systems may be underdeveloped or non-standardized.

As such, many students bypass braille math altogether and rely on linear input or audio-first approaches.

4.7.4 Audio-Based Math Access

In educational settings where braille resources are limited or unavailable, blind and visually impaired students often rely on screen readers to access mathematical content by their ears. However, conventional screen readers typically struggle to convey the structure of complex mathematical expressions accurately. This can lead to significant misunderstandings, especially when interpreting

- Nested fractions
- Exponents and roots
- Matrices or piecewise functions.

To address these challenges, several regions – particularly the United States – have developed structured math speech grammars such as MathSpeak and ClearSpeak (APH, 2022). These systems provide not only verbal equivalents for mathematical symbols but also emphasize the syntactic relationships between elements (e.g., "start fraction," "end root"). Such conventions make it easier for students to follow and mentally reconstruct the expression without visual cues.

An important recent development in this area is MathCAT, an open-source library designed to improve how math is spoken and navigated by screen readers. MathCAT supports both MathML and LaTeX and enables users to customize speech output, navigate expressions by element, and use structured reading strategies - offering a major step forward in non-visual math access (Sorge, 2023).

In parallel, innovative software such as Desmos is pushing boundaries in a different way. Its auditory graphing features allow users to explore functions by listening to variations in pitch and tone, with different sound characteristics representing function values, slope changes, or axis crossings. This sonification enables students to "hear" the behavior of a graph in real time, supplementing or replacing visual inspection.

Together, these tools reflect a growing effort to make math not just visible or tactile, but audible and navigable - supporting a wider range of learners in gaining meaningful access to mathematical information.

4.7.5 Human Support as Infrastructure

Due to technology gaps, a large number of blind students still rely on secretaries, interpreters, or support workers to transcribe handwritten work or translate their input into visual math. This mirrors the Danish findings.

Often, students will dictate math using a spoken shorthand or linear syntax. Their assistant then types it into a standard format for grading or classroom display. While this introduces dependency, it is often the most reliable way to participate in mainstream instruction.

4.7.6 Common Global Barriers

Inaccessibility of math platforms: Many digital learning tools (e.g., Khan Academy, IXL, or country-specific systems) render math as images or use inaccessible HTML structures.

Lack of trained instructors: Many educators are unfamiliar with screen reader workflows or alternative math input methods and cannot support their blind students effectively.

Exam constraints: High-stakes assessments rarely accommodate non-visual math workflows, which forces students to rely on human intermediaries or accept partial credit.

VoiceOver + MathPad (iOS)

MathPad (Pcmacstore, 2025) allows VoiceOver users to enter math using spoken descriptions or Braille screen input. While the app is designed for touch interface, blind users can navigate and enter math via a Braille display or by voice.

Output can be exported as LaTeX or MathML. Students can also draw expressions by hand (which VoiceOver won't interpret but may help low-vision users).

4.7.7 Braille notetakers

Braille notetakers include devices such as BrailleSense (braillesense, 2025) and BrailleNote Touch+ (Humanware, 2025). These specialized devices often run Android-based systems with built-in math editing tools.

Some include Nemeth Braille support or export to LaTeX. Keyboard-based math input is reliable, but integration with mainstream tools (e.g., Overleaf or Moodle) can be limited.

4.7.8 Math Solver

Math Solver (Microsoft, 2025) is a popular app available on mobile platforms (iOS and Android) and by browser.

It is a free app developed by Microsoft that uses AI to solve a wide range of math problems. It supports arithmetic, algebra, trigonometry, calculus, statistics, and more. The app can interpret problems typed in by the user, scanned from handwriting, or captured using the device's camera.

What Can MathSolver Do?

- Solve equations: linear, quadratic, systems of equations, etc.
- Step-by-step explanations: shows how the problem is solved, not just the final answer.
- Graphing: it can graph functions and equations.
- Handwriting recognition: you can write math problems by hand using your finger or a stylus.
- Scan math problems: take a photo of a printed or handwritten problem.
- Multiple languages: supports many languages, including those used in European and Asian countries.
- Additional learning tools: for example video tutorials and related concepts.

Advantages for visually impaired

- Screen reader compatibility (partial): on Android, it works to some extent with TalkBack, and on iOS with VoiceOver. Typed input can be read aloud.
- Keyboard input: students can type math problems using an on-screen math keyboard, which can be more accessible than handwriting or scanning.
- Step-by-step output: helps students understand the logic behind the solution.

Limitations

- Camera-based features are inaccessible: Scanning handwritten or printed math is not usable by blind users.
- Graphs and visual content: Graphs are not described in alternative text, making them inaccessible.
- Touch-based navigation and gestures: These can be challenging or poorly labeled with screen readers.
- No braille support: The app does not integrate with braille displays or braille math codes like Nemeth or UEB.

Microsoft Math Solver is a powerful tool for sighted students and those with partial vision who can use screen readers and magnification. However, it is **not fully accessible** for blind students, especially those who rely entirely on non-visual interfaces.

4.7.9 Online platforms accessible

Systime (Systime, 2025), GEOS (Gyldendal, 2025) and Gyldendal (Gyldendal, 2025) are all accessible "enough" for visually impaired students to get a hang of what is going on. Images, figures and equations may seem a bit "strange", with a small alt-text or accessible with a small amount of help from a secretary.

4.7.10 Challenges

While mobile and voice-based math tools offer increased flexibility for blind and visually impaired students, they also come with significant limitations – especially in educational settings where precision and structure are essential. The following outlines key challenges:

Known Challenges with Speech-to-Math Systems

- **Accuracy declines with complexity:** Voice recognition systems often struggle with nested structures like fractions within square roots or piecewise functions.
- **Screen readers lack structure:** Even when math is successfully transcribed, screen readers may not present the output in a way that clearly communicates the math's structure, requiring manual verification by the user.
- **Spoken math syntax is unintuitive:** Users must learn specific verbal conventions such as "start fraction," "left parenthesis," or "end exponent." This learning curve can slow adoption, especially among beginners.

Challenges with Mobile Math Authoring Tools

- **Notability** (Notability, 2025) **and GoodNotes** (Goodnotes, 2025)**, with VoiceOver:**
  These apps are primarily used for note-taking and audio recording. Students with residual vision or strong auditory skills may
  - Annotate accessible math PDFs
  - Pair voice labels with handwritten notes
  - Use braille displays for navigation. However, these apps are not optimized for entering structured math, and lack formal math support like LaTeX or MathML compatibility.
- **Overleaf Mobile (web version):**
  Although the Overleaf platform can be accessed via mobile browsers with screen readers like VoiceOver or TalkBack, small screen sizes and limited layout control make it difficult to edit longer LaTeX documents efficiently. LaTeX input via Bluetooth keyboards or braille displays is possible but not always practical.
- **MathML-Compatible Editors (experimental):**
  A few mobile tools - such as some LMS apps or e-book readers - allow limited MathML or MathJax rendering. These apps might support commenting or basic editing of math content. However, they are currently inconsistent in accessibility and rarely support full math authoring workflows.

4.7.11 Practical Use in Education

Mobile math tools are primarily used for informal or support tasks: note-taking, drafts, collaboration, and checking solutions.

Voice-based entry is popular for dictating quick equations during tutoring sessions or when solving homework with assistance.

Students typically do not use these tools to submit formal math assignments, due to formatting limitations or lack of compatibility with institutional platforms.

4.8 Conclusions

This has shown that blind and visually impaired students in Denmark, like their international peers, face persistent structural barriers when writing and accessing mathematics – particularly due to inaccessible educational platforms, reliance on human intermediaries, and the visual nature of math instruction.

Despite these challenges, students demonstrate resilience and creativity in adapting tools and workflows to their needs, often combining screen readers, text-based input methods (such as LaTeX), and selective use of accessible technologies like Maple or Desmos.

Solutions most useful at present include accessible, text-based tools such as **Maple** (in upper secondary and higher education), **LaTeX workflows** (especially via **Overleaf**), and **audio-enhanced environments like Desmos**, which can sonify graphs for non-visual interpretation. Screen reader-compatible calculators, braille displays, and the structured use of Word or plain-text math syntax also continue to be valuable, particularly when supported by trained assistants who can bridge gaps in accessibility.

For future development, organizations should monitor the progress of tools that can read math out loud; completely accessible (like MathCAT) as well as evolving standards for MathML and improved ARIA practices in learning management systems. Additionally, the emergence of speech-to-math systems, more robust mobile math editors, and AI-supported tutoring tools like Microsoft Math Solver should be followed closely – especially as these tools become more integrated with assistive technologies.

Further research is needed to systematically evaluate the effectiveness and user experiences of newer math accessibility tools across different age groups and educational levels. There is also a need for longitudinal studies that examine how early exposure to accessible math tools impacts later academic and career outcomes for blind students. Finally, the role of institutional practices – such as teacher training and exam accommodations – should be explored further, as these social and procedural factors are often just as critical as the technology itself.

5. Calculators and graph programs

by Marthe Gjelstad and J. M. Kvile

5.1 Introduction

In this study different calculators and graph programs have been investigated with the aim to answer the following research question: "Calculators and graph programs – what are available and how accessible are they with keyboard and screen reader?"

The following programs were tested on a Windows machine with NVDA as screen reader: GeoGebra graphing, GeoGebra CAS[1], Desmos and Microsoft Excel. The app SenseMath was tested on an iPhone 12 mini, and the Desmos Graphing Calculator app was tested both on an iPhone 12 mini and on an iPad Air. The apps were tested together with VoiceOver. In addition, some information about the proof assistant Isabelle is included, as well as some information about the program Maple. More information about Maple can be found in section 4.3.1.

5.2 GeoGebra

GeoGebra is a mathematics software where you can work with geometry, algebra, spreadsheets, graphing, statistics and calculus (What is GeoGebra?, n.d.). In this study the graphing tool and the CAS program have been tested. They were tested in a portable desktop program, GeoGebra Classic 6 for the graphing tool and GeoGebra CAS for CAS, downloaded from GeoGebra Calculators and Apps - Free Downloads - GeoGebra. They were also tested in Google Chrome on the webpage Calculator Suite - GeoGebra. Chrome was used as browser based on the instructions from GeoGebra (Accessibility, n.d.). The examples and results described in this report are based on the English versions, but they work almost the same in Norwegian. It is also possible to choose Swedish, Danish, Finnish and Icelandic as language.

5.2.1 GeoGebra Graphing

GeoGebra Graphing is used a lot in Norwegian secondary education. It is also one of the most used graph programs in Denmark.

To test the graphing tool some simple tasks were done:

1. Write a function, $f(x) = x^2 + 1$, and draw the graph.
2. Write the expression of a straight line, $x = 2$.

[1] CAS is an abbreviation for Computer Algebra System, i.e. mathematical software with the ability to manipulate mathematical expressions in a way similar to the traditional manual computations of mathematicians and scientists.

3. Find the intersection point between the graph and the line.
4. Find the derivative of the function from item 1.

All these are typical tasks students are asked to do the first year in Norwegian upper secondary education.

When entering the algebra field in GeoGebra, the screen reader speaks: "enter your equation or expression here". To write an expression, type the expression and press enter. For the function f the expression was read "f, x, equals, x 2, plus 1". In other words, it was not clarified that the number 2 is an exponent.

To find intersections between objects, type "intersect", and a menu will appear. The screen reader speaks the word "intersect", but if you use the up and down arrows to navigate in the menu, it only says "blank" for the different options. This makes it impossible to choose the right command if you can't see the different options in the menu. To find the derivative of f, you can write "derivative". Again, a menu shows up, but the different options are just "blanks".

You can use "tab" and "shift + tab" to go back to an expression/input field. When you reach an input field, the screen reader speaks the name of the function/line followed by "press slash to hide object, press enter to edit, press tab to select controls". If you press enter, the expression is read, and you can edit the expression. When going back to the line where the derivative of f was calculated, the screen reader speaks "function f prime". However, if you enter the expression, the value of the expression is not spoken.

It is difficult to use the keyboard and a screen reader to navigate in GeoGebra. If you are at an expression in the algebra field and accidentally press "delete" or "backslash", the expression is deleted without a warning. In addition, both in the desktop program and in the online version there are menus that are not possible to reach with the keyboard, and menus that are difficult to navigate in.

5.2.2 GeoGebra CAS

With the CAS calculator in GeoGebra you can solve equations, systems of equations and calculate both numeric and algebraic expressions among other things. The portable desktop version and the online version in Chrome were identical[2]. To test the calculator two simple tasks were done:

1. Simplify the expression $a + a$.
2. Solve the equation $x^2 = 2$.

To simplify an expression, you write the expression and press enter. To solve an equation, type "solve", and a menu will appear. Since the example in this study was an equation with one unknown x, the option "Solve(Equation in x)" was chosen. Like in GeoGebra's graphing tool, the different options in the menu were just spoken as "blanks". It is also possible to write "Solve(x^(2)=2)" and press "enter" to solve the equation directly. To hear the answers of the simplification and the equation, you need to tab back to the line with the expression, press

[2] There is a CAS editor in the desktop program GeoGebra classic 6, but this was impossible to use with the keyboard and a screen reader.

enter and then “shift + F4”. Then the answer is written on a new line and spoken out loud. The solution can be copied with "ctrl + a", "ctrl + c", and the solution is presented in a linear text form: "{x=-sqrt(2),x=sqrt(2)}".

Like in the graphing tool, it is difficult to navigate with the keyboard.

5.3 Desmos

In Desmos, the graphing tool was tested, which is one of the most used graph programs in Denmark. The program was tested both in Google Chrome and Firefox. There is no CAS in Desmos.

The testing was done with the language set to English both in Desmos and NVDA. In Desmos, it is possible to choose all the Nordic languages as language. However, this is not working well in Norwegian. When choosing Norwegian as language in Desmos, some buttons/menus switch to Norwegian, while some remain in English. The expressions are also still in English. So, with NVDA set to Norwegian, the Norwegian voice tries to read the English version of the expressions.

In Desmos the expression for the function $f(x) = x^2 + 1$ was read by the screen reader as “f, left parenthesis, x, right parenthesis, equals, x squared plus 1”. So, in contrast to GeoGebra, here the exponent was spoken correctly. After entering an expression and the screen reader has read it out loud, it continues to tell you that you can press “alt + t” if you want to go to “audio trace”. In audio trace you can hear the graph of an expression.

To find the points of intersection between the graph and the straight line $x = 2$, you must go back to one of the expressions. In this test f was chosen. If you press “alt + t” you reach “audio trace”, where you can listen to the graph. You can use the left and right arrow keys to move along the graph. If you reach a point of intersection, the coordinates are spoken. You can also use “tab” and “shift + tab” to move between points of interest.

To find the derivative of f, you can write $\frac{d}{dx}\big(f(x)\big)$ or $f'(x)$. The graph of the derivative appears, which you can explore with audio trace, but the expression for the derivative is not visible.

One of the most important features of Desmos related to accessibility is the audio trace. To both start and end audio trace of a graph you press “alt + t”. Changes in the stereo field reflect the value of the independent variable, and the pitch reflects the dependent variable. When the dependent variable is negative, a buzzing sound is heard. It is easy to change settings like volume and speed (Desmos accessibility, n.d.).

In general, Desmos is easy to use, and to navigate you can just use, tab, shift + tab, enter, escape and the arrows. You can reach all the menus with the keyboard.

5.3.1 Desmos app

There is also an app for the Desmos Graphing Calculator. For this report the app was tested on an iPad Air (iPadOS 18.3.2) and on an iPhone 12 mini (iOS 18.3.2) with VoiceOver. The testing was done in English and Norwegian, meaning that both the operating system and the Desmos app were set to the same language. On an iPhone it is not possible to change the language in the Desmos app. Therefore, the testing was only done in English on this device.

With the language set to Norwegian, the app is not working well. Some buttons/menus are in English, while some are in Norwegian. Also, both the Norwegian voice in VoiceOver and the English voice in the Desmos app are talking, which is confusing.

With the language set to English, things are working better. All buttons/menus are in English, and the expressions are read out correctly. However, it can be difficult to use the keyboard with touch for a blind user. Desmos recommends using a Bluetooth Keyboard (Desmos accessibility, n.d.).

To conclude, to use the Desmos app, you need to set your device/operating system to English.

5.4 SenseMath

SenseMath is an app for iPhone where you can enter expressions for functions, and the expressions are displayed visually, in speech and in braille (SenseMath - Making Sense of Math, n.d.). For this study only the visual and the speech representations were tested. The app is designed to be used together with VoiceOver. It makes you get a quick overview of a graph by representing it with sound. The testing of SenseMath was done on an iPhone 12 mini with iOS 18.3.2, where the language was set to Norwegian. The language in SenseMath is English.

To write an expression and listen to its graph you press the plus sign in the top right corner. “add button” is spoken when you press the plus sign. Below the heading “functions” you can add the expression for a graph. If you press the “explanation button”, you get information about the look of the keyboard, for example that the numbers are on the right side, while functions like “sine” are on the left side. All information is read out loud. When you have written all your expressions, you press “done” in the top right corner. Then, to hear the graphs you press play, and the sound of the different graphs are played in order. You can choose what you want to hear, for example, if you want to hear points of intersection for graphs or not.

Since the operating system and the SenseMath app were set to different languages, both languages were spoken during the testing. For example, the buttons on the keyboard were spoken in Norwegian, like “fem” for the number 5, while most of the information, like the look of the keyboard was in English. This is not a big issue because in contrast to the Desmos app, the Norwegian and English voice are not talking at the same time. However, when not in “edit” mode, the mathematical expressions are spoken by VoiceOver in Norwegian, which is not good. For example, the expression $y = x^2 + 1$ is read as “y er lik x minus kvadratrot pluss 1”. While in “edit” mode, the expressions are spoken in English, which is correct, but not perfect. For example, $x^2 + 1$ is spoken as “x caret 2 plus 1” instead of for example “x squared plus 1”.

To use SenseMath you should be familiar with VoiceOver. Since it is an app you can have on your phone, it is a great tool to use when you just want to get an overview of the shape of a graph. All buttons and information are read out loud.

5.5 Microsoft Excel

In Microsoft Excel general accessibility was tested. The program was only tested in Norwegian, but it exists for all the Nordic languages. The program has many different keyboard shortcuts, see for example Microsoft's page about shortcuts in Excel (Keyboard shortcuts in Excel, n.d.). With many different available shortcuts, it needs practice to use them.

In general, the main menu bar at the top can for example be reached by pressing "alt + h". Then you get to the home menu. To go into the menu, simply use the down arrow key. To navigate in the menu, use tab or the arrows. The menu is divided into different blocks, like "clipboard". When you enter a block, the name is spoken. Every button is spoken with a description of what the button does. If a button contains a dropdown menu, press "alt + down arrow" to open the menu and use the arrows to navigate in the menu.

To summarize, Microsoft Excel is usable with screen reader and keyboard, but it requires practice to become familiar with the various shortcuts and to get used to working in the program.

5.6 Maple

Maple is mathematics toolbox. In the program you can do simple numeric and algebraic calculations, but you can also work with integrals, differential equations and matrices. In the program you can also make different plots, like making a 3D plot of a two-variable function (Maple, n.d.; Maplesoft, 2025). See section 4.3 for more detailed information.

5.7 Isabelle Proof Assistant

Isabelle Proof Assistant (Technische Universität München, 2025) is a computer program that helps the user define concepts and prove properties about them in mathematics and computer science. Among other things, Isabelle checks whether a proof is constructed correctly and conducts some parts of the proof automatically (Schlichtkrull, 2018). Due to time constraints related to this project, the program has not been tested. Therefore, it needs further examination, especially related to accessibility.

5.8 Conclusion

To conclude, **Desmos** seems like the preferable graphing program. It is easy to use with the keyboard and a screen reader. Also, the audio trace functionality is working well and is a good tool if you can't see a graph. **GeoGebra** has a lot of different functions, and it is a useful tool if you can use the mouse to navigate. However, it is difficult to use with only the keyboard and a screen reader. The app **SenseMath** is a quick tool to use when you just want to get an overview of how a graph sounds.

The **Desmos app** is not working well if the operating system is not set to English. This makes it cumbersome to use, since you must change the language of your device to English for the app to work properly. Also, **GeoGebra** has a lot of interesting functionality, but the accessibility needs further development. **Isabelle** seems like an interesting tool, but it needs to be tested and examined further, especially related to accessibility.

6. Multiline tactile displays

by Tim Arborealis Lötberg

6.1 Introduction

Multiline tactile displays are devices containing a display composed of pins raised through holes in a flat surface. These fall into three general categories:

1. Devices that are primarily designed to **display multiline braille**. Just as in a regular one-line braille display, these can form braille characters that refresh into text in focus on a device through a screen reader (or an onboard UX). These can only display tactile graphics to a very limited extent.
2. Devices that are primarily designed to **display tactile graphics** and where the pins are designed and positioned in a way which is not optimal for braille.
3. Devices that are designed to **display both multiline braille and tactile graphics**.

Though the technology has only been available for commercial use for about five years, several different models have currently been developed. In this review we have focused on the ones which are available for purchase or released for extended user testing. These include

- Cadence (category 1)
- Canute 360 (1)
- Dot Pad 320 (3)
- Graphiti and Graphiti Plus (2)
- Monarch (3)
- Tactonom Pro (3).

The displays can be used connected to a device such as a PC or smartphone, and a few (Graphiti Plus and Monarch) can also function as stand-alone devices.

Note that this review focuses on giving an overview of what devices are available and what features they are claimed to have. Systematic testing of the displays falls outside the scope. Therefore, no conclusions can be drawn concerning how well they work in conjunction with our agencies' products and services.

6.2 Overall comparison

Characteristics and basic features of the devices are compared in the table below, followed by sections with a more detailed list of features for each respective display.

Number of pins conveys the size of the display.

All **prices** have been converted into EUR for comparison, using the exchange rates by 2025-04-17 and rounded to the nearest 10.

Weight gives an indication of how portable the device is.

Variable pin height asserts whether the pins can be partly raised. This opens the possibility for nuances in tactile images to convey e.g. colour.

Zooming means the possibility to investigate part of a graphic in greater detail directly from the device.

Drawing means the possibility to create tactile drawings directly on the display.

Cursor manipulation means the possibility to adjust the area in focus directly from the display.

Table 1 Comparison overview of multiline tactile displays. Information on price, weight and zooming is missing for Cadence because Tactile Engineering has not responded to enquiries.

Device	Number of pins	Price (€)	Weight (Kg)	Variable pin height	Zooming	Drawing	Cursor manipu-lation
Cadence	384	?	?	No	?	No	Yes
Canute 360	2,160	3,830	2.8	No	No	No	Yes
Dot Pad 320	2,400[3]	8,660	1.2	No	No[4]	No[5]	No
Graphiti	2,400	13,170	3.7	Yes	Yes	Yes	No
Graphiti Plus	2,400[6]	15,810	4.2	Yes	Yes	Yes	Yes[7]
Monarch	3,840	15,730	2.1	No	Yes	No[8]	Yes[9]
Tactonom Pro	10,472	12,000	12	No	Yes	No[10]	Yes[11]

6.3 Cadence

The Cadence (Tactile Engineering, 2025) is developed in the USA. It has a display of 384 pins (corresponding to 4 rows of 12 braille cells, equally spaced vertically but not horizontally), making it comparable in size to a large smartphone. The pins are spaced primarily for braille rather than graphics. It

[3] Plus an additional line of 20 braille cells.

[4] Possible to a limited extent, see section 6.5.

[5] Possible on external device via drawing app.

[6] Plus an additional line of 20 braille cells.

[7] Single line only.

[8] Not at present, see section 6.7.

[9] Touch.

[10] Not at present, see section 6.8.

[11] Optical recognition of finger position.

can be connected to a PC or a mobile unit via USB or Bluetooth. Up to four units can be connected to form a larger screen.

Features include:

- Possible to read while the display is reimaging without interfering with the pin pattern, meaning that dynamic changes can be followed in real-time.
- 4-key Perkins keyboard (meaning that 2 connected devices are needed for typing).
- Several days of battery life during heavy use.

6.4 Canute 360

The Canute 360 (Bristol Braille Technology, 2025) is developed in the UK. It has a display of 2160 pins, corresponding to 9 lines of 40 braille cells. There is spacing without pins between the braille lines. Graphics shown are constructed from sets of full braille cells via a custom format, meaning they show up as they would having been embossed on paper. It can be connected to a PC via USB or HDMI. It is portable, comparable in size and weight to a laptop computer.

Features include:

- Supports all pre-formatted Braille files, such as BRF and PEF.
- Can be used as a stand-alone device to read books directly from a USB drive or SD card.
- Compatible with all OS that run BRLTTYY, including Windows, Linux and Mac.
- Compatible with Narrator and NVDA.

6.5 Dot Pad

The Dot Pad 320 (Dotincorp, 2025) is developed in Korea. It has a display of 2400 pins, corresponding to 300 braille cells, making it suitable for tactile graphics as well as multiline braille. It also has an additional 20 cell text braille display below the main screen. It can be connected to a PC, tablet or phone via Wi-Fi, USB or Bluetooth. It is portable, size and weight comparable to a lightweight laptop computer.

Features include:

- Zooming is possible to a certain extent via VoiceOver but must be prepared in the images. Not possible directly from the image viewing tool.
- Handles formats PDF and DAISY as well as a variety of image formats such as JPEG, PNG and TIFF.
- It comes with an app for iPad and a web app for creating presentations with tactile images and text.

- Instant conversion of images of any format to line drawings via internal format SDK. This means that it’s possible to draw on e.g. a tablet and see the result on the Dot Pad in real time.
- Compatible with VoiceOver and NVDA. JAWS support is currently in beta testing.

6.6 Graphiti

The Graphiti (Orbit Research, 2025) and Graphiti Plus (Orbit Research, 2025) are developed in the USA. They have a display of 2400 pins. The pins are bigger than braille dots typically are, meaning that braille on the screen could be difficult to read. The Graphiti plus, however, has an additional 20 cell text braille display below the main screen. Both devices can be connected to a PC, tablet or phone via HDMI, USB or Bluetooth. Graphiti plus can also be used as a stand-alone device. They are both portable, comparable in size and weight to a heavier laptop computer.

Features include:

- Variable pin height, e.g. for showing different colours as different textures.
- Possible to zoom and pan in tactile graphics.
- Possible to draw directly on the display and save in a visual output format.
- Possible to read while the display is reimaging without interfering with the pin pattern, meaning that dynamic changes can be followed in real-time.
- Possible to use the tactile display as a touch screen for a connected device.
- Provides haptic output.
- Can read directly from USB drive or SD card.
- 6-key Perkins keyboard (8-key for Plus).
- Built-in TTS (Plus).
- Built-in apps including book-reader, note-taker, calculator and Braille translation in 40+ languages (Plus).

6.7 Monarch

The Monarch (APH, 2025) is developed by HumanWare in collaboration with American Printing House for the Blind (APH) in the USA. It has a display of 3840 pins, corresponding to 10 rows of 32 braille cells, making it suitable for tactile graphics as well as multiline braille. It can be connected to a PC via Wi-Fi, USB or Bluetooth but can also be used as a stand-alone device with built-in applications. It is portable, comparable in size and weight to a laptop computer.

Features include:

- Possible to zoom and pan in tactile graphics.

- Handles a variety of formats, including EPUB, DAISY, JPG, PNG, PDF, BRF and PEF.
- > 20 hours of battery life during active use.
- Built-in TTS that can read MathML aloud.
- Cursor manipulation by touch directly on the display.
- Customisable spacing between braille lines.
- Built-in apps include graphic calculator, tactile graphics viewer and word processor.
- 8-key Perkins style keyboard.
- Possible to connect directly to accessible book libraries and APH:s tactile graphics library (APH, 2025).
- Possible to write math in braille (Nemeth, UEB and French currently supported) and output in visual formats.
- An app for creating tactile drawings for Monarch is currently in beta testing.

6.8 Tactonom Pro

The Tactonom Pro (Tactonom, 2025) is developed by Inventivio in Germany. It has a display of 10 472 pins, corresponding to 22 lines of 40 braille cells. It is designed to function as a stationary tactile display to a PC and connects via HDMI. Only Windows is currently supported but support for all operating systems is on the roadmap. A camera is mounted above the display which can track finger position.

Features include:

- Can read all formats which a PC can handle. Automatic conversion of images which are not already in a tactile format.
- Possible to zoom and pan in tactile graphics.
- Reading math in braille (currently only Nemeth supported).
- Compatible with NVDA (JAWS support is on the roadmap).
- There is hardware support for drawing directly on the display via the camera, but no app has been developed for it yet.
- Direct access to thousands of tactile graphics in the ProBlind database (ProBlind, 2025). Automatic translation of the text in all graphics is available to many languages.

6.9 Future development

Multiline and dynamic tactile displays open up possibilities for tactile interaction with STEM content in a groundbreaking way. However, the technology is expensive (comparable to the price of the first refreshable braille displays in the 1980's), which in itself can be an accessibility issue. In the Nordic countries, braille readers may be prescribed a braille display if it is classified as aid equipment and would then not have to bear the cost themselves. Multiline devices are not, however, classified as aid equipment at present. In terms of resources, the cost of a device should also be compared to production of printed braille books and tactile graphics.

Research is underway to develop more effective actuator technology, meaning that fewer components would be required to move a certain number of pins. This will make it possible to create cheaper displays where the cost does not scale with size. One such initiative is the EU project ABILITY (ABILITY, 2025), where ultrasonic vibrations from just a few transducers is used to activate pins remotely. Another example of alternative actuator technology use is the Tactonom Pro, which despite its remarkably large screen is cheaper than e.g. the Monarch and Graphiti Plus.

An example of an affordable analogue solution is the BrailleDoodle (TouchPad Pro Foundation, 2025). It is a tablet where the pins consist of stainless-steel ball bearings that can be raised or lowered with a magnetic stylus. One side is for drawing with the stylus on a grid of 1 033 balls, while the other side is for practicing the braille alphabet with fixed letters to read and copy. Costing 170 €, it is made to be significantly more affordable than its digital counterparts while still useful for a variety of classroom activities.

An important aspect of usability of multiline devices is screen reader support. Work is already underway to support multiline and dynamic tactile devices with screen reader integration. NVDA has since been the first screen reader to offer functional support (GitHub, 2025). The 2025 CSUN conference marked a milestone, with all major screen reader developers showing a commitment to supporting multiline and tactile displays.

What will reading tactile images be like in the future? Currently the multiline displays have different ways of translating and rendering images into a tactile format. Some take regular image formats such as JPEG or PNG and translate on the fly via an internal format. These graphics will be of unpredictable quality since they have not necessarily been prepared for tactile viewing.

The SVG format supports scripting information about e.g. tactile presentation and interactivity into a digital image (W3C, 2025). The EPUB format in turn supports the inclusion of scripted SVG (W3C, 2025). This means that in theory, tactile and interactive graphics could be embedded in an e-book and be accessible directly through a multiline tactile display. However, a standard doesn’t necessarily imply that full hardware and software support exists. In practice, the parts have yet to be fitted into a cohesive solution and tested by users.

It should also be kept in mind that while the technology matures and multiline tactile displays might become more affordable, they can’t necessarily replace traditional techniques such as swell paper or embossing for all purposes. Regardless of medium, tactile image production will be dependent on standards and recommendations. MTM and SPSM will carry out a study during 2025 with the aim of describing and analysing knowledge about tactile images for the purpose of producing up-to-date recommendations (MTM, 2025). Further on, user-focused studies will be required to shed light on how the analogue and digital technologies might complement one another.

6.10 Conclusions

Multiline tactile displays can do quite a lot of things, depending to some extent on how graphics files have been prepared. Features and performance vary between models, but all can be used to access digital STEM graphics in a tactile way. Monarch, Dotpad, Graphiti, Tactonom & Cadence all seem good for this purpose. Canute doesn’t come close in resolution but could be sufficient for bar charts and very basic graphs or geometry. Monarch and Tactonom Pro, on the other hand, stand out in terms of resolution because of their large displays.

Both Graphiti models and Cadence allow for dynamic viewing since the display can be read while the pins are rearranging. The resolution of the graphics will be lower on the Cadence because of the spacing of the pins, while the Graphiti models allow for an additional layer of nuance since the pins have variable height.

For creating content, the Graphiti models and the analogue BrailleDoodle (and in the future the Tactonom Pro) stand out with the ability to draw directly on the display.

While all displays except the Tactonom Pro are portable, Cadence stands out especially for being small enough to fit in a pocket. Graphiti Plus and Monarch also stand out in terms of portability since they can be used as stand-alone devices. Tactonom Pro, while being stationary, is designed to display anything that a computer can do, meaning that the user won’t be locked into any developer-specific apps.

Overall, Graphiti Plus and Monarch seem to be the most versatile, but also the most expensive displays. In a stationary setting, the Tactonom Pro offers many of the same features for a lower price.

7. Graphs and diagrams

By Tim Arborealis Lötberg

Graphs and diagrams are an integral part of many STEM fields. A heavy reliance on visual representation is often necessary, which leads to accessibility challenges. As new technologies emerge, the possibility for multi-modal representation of graphic content opens up. Examples include image descriptions, tactile images, sonification and haptics. In this review we take a broad perspective and list a selection of solutions and projects as examples of what is possible. Testing and evaluating solutions for specific purposes falls outside the scope.

7.1 Image descriptions

Image descriptions are often essential for making graphic STEM content accessible. When the user doesn't have vision as a means to take in the big picture or small details, an image description provides an overview which facilitates orientation and understanding of what is being presented. This is essential also in combination with alternative modalities such as sound and haptics.

7.1.1 Poet

Poet (DIAGRAM Center, 2025) is a web-based image description resource. The idea is to help people learn when and how to describe images frequently found in educational books. The tool includes best practice guidelines as well as exercises. It is developed by the DIAGRAM Center (**D**igital **I**mage **A**nd **G**raphic **R**esources for **A**ccessible **M**aterials), an initiative by Benetech.

7.1.2 Seeing AI

Seeing AI (Seeing AI, 2025) is a free app which narrates the surroundings as seen through a mobile phone's camera. It is one of many similar solutions for auto-generated image description, whose proficiency is accelerating (DAISY, 2025).

7.2 Haptics

Haptics is the technology of vibrational feedback. In STEM contexts, it can for example be used to explore scatterplots on a touchscreen, where a higher intensity of vibrations corresponds to a higher density of data points.

7.2.1 Multimodal Digital Graphics on Touchscreens

Multimodal Digital Graphics on Touchscreens (CHROME Lab, 2025) is a research project which explores how vibrations and additional multisensory feedback can be used to enhance the accessibility and usability of touchscreens, particularly in visual content such as graphics.

7.3 Sonification

Sonification is the use of non-speech sound in an intentional, systematic way to represent information (Walker & Nees, 2011). For example, the pitch of a tone can be mapped onto the value on the y-axis of a graph, or tempo can be mapped onto data representing speed. Sound can also be used in many other ways in STEM simulations. The technology has been around for decades (Pollack & Ficks, 1954) and new implementations continue to emerge. A couple of solutions for student use are listed below:

7.3.1 Astronify

Astronify (Astronify, 2025) is a Python package for sonification of astronomical data.

7.3.2 SenseMath

Sensemath (Q42, 2025) is an iOS app for sonification of graphs, developed by Q42 in collaboration with Royal Visio. This app is discussed in detail in section 5.4.

7.4 Tactile images

Tactile images are raised-line or textured representations of graphic content accessible by touch. Traditional techniques include embossing, thermoforming, swell-paper printing, 3D printing, laser etching and different collage techniques. Multiline tactile displays (see section 6) open up the possibility for accessing digital images in a tactile way directly.

7.4.1 Blind SVG

Blind SVG (Blind SVG, 2025) is a resource to help teach BVI persons how to code their own graphics with SVG.

7.4.2 Dimensions

Dimensions (New York Public Library, 2025) is a set of community tools for making tactile graphics and objects.

7.4.3 ProBlind

ProBlind (ProBlind, 2025) is a free and open-source database for tactile graphics.

7.4.4 Tactile Vega-Lite

Tactile Vega-Lite (MIT News, 2025) is a system for streamlining the tactile chart design process. It is developed at MIT CSAIL.

7.5 Other tools and projects

Below are listed a selection of projects and solutions for enhancing and checking the accessibility of graphics.

7.5.1 Chartability

Chartability (MIT News, 2025) is a set of testable questions for ensuring that data visualisations, systems, and interfaces are accessible. It is a free open-source tool.

7.5.2 Data Navigator

Data Navigator (CMU Data Interaction Group, 2025) is a free, open-source tool for rendering a semantic, navigable structure on top of graphics.

7.5.3 Highcharts

Highcharts (Highcharts, 2025) is a charting library for accessible data visualisations for web and mobile platforms.

7.5.4 Inclusio

Inclusio (Inclusio Community, 2025) is a project by the National Science Foundation in the USA. It aims to connect content providers, educators, and individuals with blindness and low vision through a one-stop shop, saving time and resources in obtaining high-quality accessible information on multiple platforms.

7.6 Conclusions

As new technologies develop and mature, more modes open up for experiencing graphic content than just visually. Sonification, tactile images, haptics and image descriptions all have great potential in the STEM fields. Rather than replacing, these techniques complement one another and ideally a user should have access to several modes of representation.

From the examples above, it is evident that there is a great number of promising solutions for accessible STEM graphics. However, graphs and diagrams is a large area, of which this review has only skimmed the surface. MTM and SPSM are launching a more in-depth overview of knowledge of tactile images (MTM, 2025) during 2025. This will be used as a basis for future development and guidelines.

How is the support for interactivity and multi-modal presentation for graphic content in e-books? The SVG image format has support for embedding multiple layers and modalities of information into it. The EPUB standard supports embedded SVG (W3C, 2025), but how the information can be accessed in various reading situations remains to be investigated.[12]

Lastly, technological development and the development of standards go hand in hand, as always in accessibility. It is important to adhere to existing standards, for example Guidelines and Standards for Tactile Graphics developed by BANA (Braille Authority of North America, 2012). We recommend that the Nordic agencies take an active part in following both the development of technology and standards so that our users will get access to the cutting-edge of accessible books.

[12] The eBraille project (The DAISY Consortium, 2025) is currently looking into this.

8. Programming interfaces

By Lars Henrik Johansen

8.1 General considerations

8.1.1 Definitions

This section deals with the accessibility of programming interfaces, but this necessitates a clear definition of the expressions ‘programming interface’ and ‘programming’. Firstly, we will limit ourselves to the digital realm, including neither Joseph Marie Jacquard’s weaving device, nor Microsoft’s Code Jumper – at least not the tactile part of it. Secondly, we *will* include interfaces where the programming language is inseparable from the interface, such as Scratch. Thirdly, we will focus on interfaces that handle higher level programming, excluding simulators of electric circuits and similar. Fourthly, we focus on editors that facilitate programming, in the sense of instructing the computer, as opposed to markup, in the sense of structuring a document, and hence we do not include text editors such as Word, but we do include text editors such as Notepad.

8.1.2 Previous work

Statped in Norway has done work in the field of programming with visual impairment, and collected considerations and tips on statped.no (Statped, 2025).

In addition to this, Øivind Rønning (senior adviser, Statped) has collected evaluations about specific programming interfaces in a survey from 2020 onwards (Rønning, coding-without-seeing, 2023).

Not only is this a field developing day by day, but it is also complex in a way that makes different users focus on different sets of aspects and hence may come to different sets of conclusions. Some parameters remain relatively objective: if you cannot read the text, then it does not work. Others are more a matter of taste and convenience, and Vim, for example, might appear clumsy to some users, but elegant to others, and any survey, the present text included, will be biased in some way.

8.2 Evaluation criteria

The evaluation criteria described here serve a double function: they have guided our testing but are also meant as suggestions for ordering further work in this field.

8.2.1 Installation and setup

This seemingly simple phase might be the cause of frustrations, sometimes because the producer’s web site might be less accessible than the actual program, sometimes because computer onto which we want to download the program might be subject to restrictions by the organization distributing it, and sometimes because of incommensurability between the program and the system on which we wish to use it.

8.2.2 Basic navigation

It happens that although the elements that we want to navigate – such as programming blocks – conceptually might perfectly well be navigable, but that the program itself – such as Scratch – does not provide navigability for the navigation method such as a screen reader.

8.2.3 Code editing

Typing, drag-and-drop, tactile input devices and voice controllers are all relevant code editing methods. However, it happens that the very methods that render beginner level understanding more efficiently hinder the progress to higher levels of programming or that an input method is very efficient for an expert user while too complex for the beginner. This effect is by no means particular to programming but is very relevant for us in this context and makes it less obvious how to judge an input method as “good” or “bad”.

8.2.4 Debugging and Error Messages

Coding is a close relative to *troubleshooting*, and a programming environment that does not facilitate finding and correcting errors is less useful than one that does.

8.2.5 Autocomplete and Suggestions

On higher levels, efficiency in programming is highly improved by tools that aid the completion of specific tasks. These include AI copilots, editor autocomplete, language servers, predefined snippets and templates, linters, refactoring tools, shell autocomplete tools, chat services, and different experimental functions for an Integrated Development Environment (IDE). Since modern programming integrates these tools in its routine, the accessibility of them is relevant when summing up on the general accessibility of the IDE to be evaluated.

8.2.6 Accessibility Settings and Customisation

There are two considerations that are not entirely unrelated: what the user *needs* and what the user *likes*. Often the user would *like* to have what is *needed*, but it not always obvious if a *liking* is a necessity. High contrast mode, font and zoom control, screen reader support, customizable keybindings, speech-to-code and voice commands, error and warning narration, caret and focus indicators, minimap toggle, distraction-free mode and accessible extensions are all accessibility aids, but most of them are also subject to taste.

8.2.7 Terminal/Console Access

Many programming tasks – from installing packages to compiling code – are often performed via the terminal, and hence the accessibility of the terminal might affect the overall accessibility of the programming environment. The navigability of terminal output is essential for tasks like troubleshooting.

8.2.8 Documentation and Community Support

When we evaluate an IDE, the *information ecosystem* could be viewed as an integral part of the product. Accessible documentation in multiple structured formats and an inclusive, responsive community help the user learn independently, solve problems quickly, and participate fully in the programming ecosystem.

8.3 Specific programming interfaces

8.3.1 Jupyter Note-book

Anaconda installs like any desktop program and ships web-based Jupyter Note-book and JupyterLab (editor, terminal, etc.) that run in a browser. JupyterLab lets sighted users enable a dark theme, pick alternate editor colour schemes, and enlarge fonts for the UI, code, and terminal. Moderate magnification, Windows High-Contrast, and colour-filter modes also work reasonably well.

However, current versions remain poorly accessible to screen-reader users (NVDA, JAWS, Windows Narrator). In forms/typing mode the arrow keys give no speech or braille feedback, cursor-movement can drop into virtual/browse mode unexpectedly, and back-spacing is silent—serious barriers that outweigh the otherwise solid editing features. Because of these limitations, Jupyter Note-book/Lab is not recommended for learners who rely on speech or braille output, though it can suit low-vision students who benefit mainly from visual customisation.

See (Rønning, Jupyter Notebook og JupyterLab, 2021).

8.3.2 Notepad++

Notepad++ is a free code editor that supports many programming languages with localisation for the Nordic countries except Iceland (according to the official list). It works well with screen readers such as JAWS, NVDA, and SuperNova—though users often need to tweak both the editor (e.g., hide line numbers, turn off autocompletion) and their screen reader settings for optimal braille output. Visual users can customise colour themes, dark mode, fonts, zoom, and hide UI elements to reduce clutter.

Installation is straightforward from notepad-plus-plus.org, but the editor ships without a built-in console. To run programs (e.g., Python) you must install the NppExec plugin, create a script that launches the interpreter, and assign a keyboard shortcut; similar steps apply if you use uflash to program BBC micro:bit boards.

Additional tips include defining a default language and working folder, fixing Scandinavian characters in the console, and consulting the built-in hot-key manager. While Notepad++ lacks integrated linting or rich error panels, plugins or console output can partly fill that gap. Overall, with the right setup it is a lightweight yet accessible option for both visually impaired and sighted students.

See (Rønning, NotepadPlusPlus, 2023).

8.3.3 PyCharm

PyCharm Community installs with its own Java runtime (just enable Java Access Bridge) and supports only Python, plus a micro:bit plug-in. Sighted users get light/dark themes, red-green colour-blind filters, per-pane colour schemes, custom fonts and mouse-wheel zoom—though zoom does not affect every panel.

With NVDA PyCharm is mostly usable, but both NVDA and JAWS lose proper braille cursor routing; JAWS also drops entire lines from the display after a backspace. Code-completion pop-ups are unreadable and should be disabled, while the "Problems" window (Alt + 6) is navigable and returns focus to offending lines. The Python console gives speech and braille while typing but no feedback when you move with arrow keys, and the integrated terminal is effectively silent. All shortcuts are re-mappable via Settings › Keymap, which helps compensate for these gaps.

See (Rønning, PyCharm, 2022).

8.3.4 Scratch

Scratch's block-based interface is unusable with screen readers: there are no keyboard-navigable links, headings, or landmarks, so blind users cannot reach or identify code blocks. Low-vision learners can manage with moderate magnification, ZoomText, or colour-enhancement, but results vary, and high-contrast modes often produce clashing colours or leave parts of the interface unaffected—especially in the desktop app. In short, Scratch suits sighted beginners but is not a practical option for students who rely on speech or braille output.

See (Rønning, Scratch, 2020).

8.3.5 Visual Studio Code

VS Code is a highly customisable, screen-reader-friendly Python editor: you can zoom the whole UI, switch among high-contrast or dark/light colour themes, simplify the window with Zen Mode, and navigate almost everything by keyboard. NVDA, JAWS and VoiceOver read the editor, Problems panel, explorer tree and command palette well (installing the "Screen Reader Mode" extension fine-tunes defaults), while audio cues and the Problems list make finding errors straightforward; however, the integrated terminal still requires an "Accessible View" workaround or an external console. With Python, Pylint and other extensions, VS Code becomes a fully-featured yet accessible environment for both low-vision and blind learners—though pointer-size/colour tweaks must be set in Windows, not in the app itself.

See (Rønning, Visual Studio Code, 2024).

8.3.6 Other editors

The above selection is meant as a glance into different directions of IDE variants. There are of course many others, and some, like Emacs and Vim appear particularly usable, particularly with the refreshable braille display.

8.4 Conclusions

The comparative testing above points to Visual Studio Code as the most universally recommendable environment today: with its extensive screen-reader support, flexible theming, keyboard-centric workflow and rich ecosystem of extensions, it accommodates both blind and low-vision programmers while remaining appealing to sighted peers. Notepad++ is a strong second choice when a lightweight, Windows-only editor is needed; it installs easily, behaves predictably with major screen readers once tuned, and can be extended with plugins such as NppExec to cover basic run-and-test loops. PyCharm merits consideration for more advanced Python work provided the user accepts its current braille-display limitations and invests time in remapping shortcuts and disabling inaccessible pop-ups. By contrast, Jupyter Note-book/Lab and Scratch should not be recommended to learners who rely on speech or braille output until their fundamental navigation issues are resolved, although they remain valuable for sighted or low-vision beginners. Veteran text-centric tools like Emacs and Vim can be highly effective—especially in combination with a refreshable braille display—but demand steep learning curves and were therefore kept outside the core recommendations for mainstream teaching contexts.

Looking ahead, the field would benefit from systematic, longitudinal studies that measure real-world productivity across diverse disability profiles rather than short accessibility checklists. Three gaps are especially pressing: (1) terminal integration—even in otherwise strong editors, command-line panes often lag behind the main interface in screen-reader feedback; (2) block-based and drag-and-drop paradigms—current solutions marginalise blind users, yet these paradigms dominate early-years curricula; and (3) evaluation of emerging AI copilots and language-server features—their rapid evolution risks outpacing accessibility work. Regularly updated test suites, shared configuration recipes, and closer collaboration with upstream developer teams will be crucial for maintaining accessible programming pathways as tooling and pedagogical targets continue to shift.

9. Learning platforms

By Evelina Frischenfeldt Bååth

Learning platforms are widely used in an educational context all over the world today. Yet, their level of accessibility does not cater to every student's needs.

9.1 Introduction to learning platforms

Learning platforms are digital platforms that offer educational content. They can be either online websites or applications for download, and they are frequently used in modern education all over the world. In the educational setting, the term covers both solutions which are more administrative and organisational, henceforth referred to as learning management systems (LMS), as well as solutions which are more directly aimed at learners, called virtual learning environments (VLE). Due to this major difference in target group, the properties of LMS and VLE differ greatly so for the purpose of this study, the following definitions of the terms LMS and VLE were accepted:

1. Learning management systems (LMS): A platform where the school makes the learning content and controls the system to some extent. However, most of these platforms are originally created by other parties, such as companies or organizations, and therefore the school's possibilities to influence the platform are limited. For example, Canvas (Infrastructure, 2025) is a learning management system used by universities in all Nordic countries. Other examples are PING PONG (Ping pong, 2025), Unikum (Unikum, 2025), Vklass (Vklass, 2025) and Moodle (Moodle, 2025).
2. Virtual learning environments (VLE): A platform where the school does *not* make the learning content. Also, since these platforms are not aimed at school management, they typically do not offer tools for documentation or communication with students or guardians. These platforms can be made by for example companies, non-profit organizations, associations or governmental authorities. Examples: Khan Academy (Khan Academy, 2025), Coursera (Coursera, 2025), Skillshare (Skillshare, 2025), Skolon (Skolon, 2025) and Bingel (Bingel, 2025).

9.2 Background and previous studies

In a world that is becoming more and more dependent on digital tools every day, so is also the learning system becoming increasingly digitised, including learning platforms. Maidenbaum et al.[13] has found that virtual environments are turning into increasingly central parts of our lives, yet numerous studies[14] report that the virtual environments included in these studies were not accessible for people with visual impairment. Furthermore, several other studies which have analysed websites specifically prove that a majority of these sites are not accessible (Armstrong, 2009). This further adds to the digital divide and, consequently, affects the learning potential for students with visual impairment. More recent studies[15] build their research on a pre-existing acceptance that virtual learning environments are not accessible and focus therefore instead on locating the problems and finding solutions.

As welcome as this progress in revealing and amending the issues of accessibility in virtual learning environments is, the fact that essentially all school systems today utilize a learning management system needs to also be addressed. Burke et al.[16] states that even courses attended in person have a digital learning management system which is most likely not accessible for students with disabilities such as visual impairment. This hypothesis is strengthened by the fact that people with disabilities are amongst the least considered when it comes to the educational context of online learning (Armstrong, 2009).

The transition to more digital, mobile and online learning, has been well on its way ever since the creation and employment of digital tools but it has increasingly intensified ever since the start of the green transition which requires a significant increase in the number of STEM educated people (Kaufhold & Steinert, 2024). Today, many visually impaired students drop out of such studies due to obstacles in accessibility[17]. This might however be affected by present and future legislation such as the Americans with Disabilities Act (ADA) or other national counterparts.

Another most compelling factor for the increase of digitalisation in schools was the covid pandemic which added to the already existing demands of long-distance learning. Together with the possibilities for BVI students in developing countries to continue education (Kamaghe, Luhanga, & Kisangiri, 2020), this sums up to a major demand on LMS and VLE as mobile, online tools for learning to be more accessible.

[13] (Maidenbaum, Levy-Tzedek, Chebat, & Amedi, 2013)

[14] See (Gutiérrez Gómez-Calcerrada, Solera Hernández, & García González, 2005); (Park, So, & Cha, 2019); (Gutiérrez Gómez-Calcerrada, Solera Hernández, & García González, 2005) and (Nganji & Brayshaw, 2017).

[15] See (Kamaghe, Luhanga, & Kisangiri, 2020); (Kaufhold & Steinert, 2024) and (Riley-Ancar, 2022).

[16] (Burke, Clapper, & McRae, 2016)

[17] See (Kamaghe, Luhanga, & Kisangiri, 2020); (Kaufhold & Steinert, 2024) and (Riley-Ancar, 2022).

9.3 Research questions

Since learning platforms are widely used in modern education all over the world, the question of their accessibility becomes a high priority for all manners of learning. Yet, despite the increased digitalisation in every instance from pre-school to higher education, research on the accessibility of learning platforms seems quite rare based on the number of previous studies found in the making of this report. Most of these studies were mainly focused on higher education and online learning[18], and no study found was performed in any of the Nordic countries. However, most of these studies do persistently prove that learning platforms are not accessible and therefore it becomes essential to continue with further research on the topic.

The most immediate aspect to study is whether the LMS and VLE used in Nordic schools today meet the AA level standards of the WCAG 2 (W3C, 2024). This question would benefit from being studied separately for each level of schooling, i.e. elementary, high school and university or higher studies, since such research could offer more precise conclusions and the possibility to identify potential obstacles in progression between the different levels of education. Also, a special focus on STEM subjects when researching the accessibility of VLE would offer further knowledge on the accessibility of VLE used in STEM education in the Nordic countries specifically.

Another aspect to consider is the question of how well learning platforms interact with other applications, for example screen readers and conversion tools, to become more accessible.

9.4 Findings

The proposed research questions have not been investigated within the scoop of this report. Further research on the accessibility of learning platforms has therefore yet to be done and in such work, the above defined research questions offer a foundation for future studying.

However, in preparations of any future research, it has been found that the lack of access to several existing learning platforms causes a major problem in the analysis, since most platforms require a paid subscription and/or a school license.

[18] (Hadian & Storey, 2005); (Kamaghe, Luhanga, & Kisangiri, 2020); (Kaufhold & Steinert, 2024); (Ready, 2017) and (Riley-Ancar, 2022).

10. Conversion tools

By Tim Arborealis Lötberg

MathML markup has the most potential for accessible math content (The DAISY Consortium, 2025). Producing high-quality MathML, however, remains a challenge and few publications today are born containing MathML. There are several solutions for converting math content from a less accessible format (image, PDF) to a more accessible format (LaTeX or MathML), a selection of which are listed below. Note that intended functionality is only mentioned in brief since evaluating specific solutions falls outside the scope of this review.

10.1 MathKicker

MathKicker (MathKicker, 2025) is a web application using AI to transform mathematical expressions into an accessible format. It is designed to be used by blind students for uploading PDFs or images of mathematical documents for conversion into word or HTML. It also provides a browser-based math editor.

10.2 MathPix

MathPix (MathPix, 2025) is an OCR tool for transforming PDFs and images into LaTeX or markdown. It handles mathematical content as well as tables and structural markup such as headings.

10.3 Mistral

Mistral (Mistral, 2025) provides an AI-based OCR tool for transforming PDF and images into LaTeX or markdown. It handles mathematical content as well as tables and structural markup such as headings. It does not output MathML.

10.4 Morf

Morf (UNAR labs, 2025) is an AI-based app for math document conversion. It converts PDF, Word, PNG or JPG into the formats DOCX, PDF, SVG and BRF. More output formats are to be implemented; EPUB and eBraille are currently being looked into. Math content outputs in LaTeX, MathML, Nemeth or UEB. The app is currently in beta testing with release planned for September 2025.

10.5 Pandoc

Pandoc (Pandoc, 2025) is a free document conversion tool that handles a lot of different formats. LaTeX math is converted (depending on the output format) to Unicode, native Word equation objects, MathML, or roff equation[19] .

[19] Roff is an old-school document formatting language for Linux systems.

10.6 WordToEPUB

WordToEPUB (DAISY, 2025) is a free open-source tool developed by the DAISY consortium. It converts Word documents into EPUB format. Math content in Word’s format OMML is converted into MathML.

10.7 Conclusions

The possibilities and limitations of the conversion tools listed above remain to be investigated. The following questions should be considered for an in-depth study:

- Is the produced markup semantically correct? Would a screen reader read the result correctly?
- Automatic conversions will likely need a human in the loop for quality assurance at some point. To how great a degree does the tool produce accurate results, thus lessening the time needed for QA?
- What happens to the material the tool is used on? Might it be used for e.g. AI training? What limitations does this imply for conversion of copyrighted material?
- What does it cost in terms of money? What is the imprint of the tool in terms of sustainability?
- What are the possibilities for using the tool as a part of a large-scale production process?

11. Large Language Models

By Tim Arborealis Lötberg

Large language models (LLMs) are AI systems trained on massive text corpora to predict and generate human-like language, enabling them to answer questions, solve problems, and produce explanations. In recent years, benchmarks such as OpenAI’s MathVista (Lu, et al., 2024) have emerged to evaluate LLM performance on mathematical reasoning tasks—ranging from algebra and geometry to calculus—demonstrating that top models can now solve many textbook-style problems with high accuracy (Yan, et al., 2025).

In STEM classrooms, teachers can leverage LLMs to create accessible learning materials on the fly. For instance, an instructor might ask ChatGPT to convert a complex calculus proof into a step-by-step narrative at varying reading levels, or to generate image descriptions of chemical structures for students with visual impairments. LLMs can also produce customized problem sets, complete with hints and worked solutions, freeing educators to focus on one-on-one support.

Beyond text generation, LLMs equipped with multimodal capabilities can process images of mathematical notation and render them into spoken descriptions. A student with low vision could snap a photo of a geometry diagram or a matrix equation, and the model would read aloud each element—vertices, labels, entries—while conveying relational structure (“the second row has entries three, minus one, and five”). Conversely, via dictation interfaces, learners who struggle with fine motor control can speak an equation (“integral from zero to one of x squared dx equals one third”), and the LLM will transcribe it into formatted LaTeX or MathML.

It is beyond the scope of this review to investigate LLMs’ possible applications for STEM accessibility in depth. We will, however, note that this field develops rapidly enough that what was unfeasible last month might be possible today. To understand the promise and limitations of LLMs, future studies should investigate:

- **Accuracy and misinterpretation rates** – How often do models misread symbols in images or mistranscribe spoken math, and what error patterns emerge? What is the quality of the MathML markup produced and how much semantics is preserved?
- **Usability for diverse learners** – Which interface modalities (voice, text, haptic) work best for students with different disabilities?
- **Impact on learning outcomes** – Do LLM-generated accommodations measurably improve comprehension, retention, and confidence in STEM subjects? If so, in what way should they be used to enhance rather than avoid learning? Teachers already have experience of interest in this field, see e.g. (AI Lund, 2025).
- **Bias and equity considerations** – Are there disparities in performance on notation styles, languages, or curricula from different regions?

- **Integration with AT** – How can LLMs best interoperate with screen readers, refreshable braille displays, and tactile graphics printers?
- **Resource impact** – What do the use of LLMs for STEM accessibility cost in terms of money and environmental impact?
- **Copyright** – What happens when various LLMs are used on copyrighted materials? Is the material used for training?

12. Overall conclusions

This tech review has provided an overview of hardware and software with STEM applications. With such a broad scope, we have only scratched the surface when it comes to evaluating specific solutions. We have, however, identified what gaps in our knowledge need filling, and sifted out a few suggestions for development which could be undertaken within the near future.

12.1 Suggestions for further study

Further study is required for in-depth analysis of solutions with promising accessibility aspects. Integration of separate technologies into a cohesive user experience also requires further investigation. Naturally, this should involve user testing.

An important aspect in all future studies is security. Introducing connected devices unavoidably introduces potential cyber security risks. A risk management plan could include data traffic analysis (as in the Lion Cage project (LinkedIn, 2025)) and similar testing in controlled environments. This work could be combined with encouraging transparency by vendors.

Another aspect to consider is potential specialised accessibility needs for specific STEM fields. For example, chemistry- or physics-related content could require alternative context-based markup in MathML. The support for this has not been explored within the scope of this review. Ideally, expert users within a variety of fields should be consulted regarding their accessibility needs.

An additional factor to consider in accessible book production is copyright. With rapidly emerging AI technologies, transparency regarding what happens to the material that AI-based tools are used upon is essential.

Below are listed suggested follow-up studies:

12.1.1 E-book testing

Create an e-book containing systematic representations of different math elements, including chemistry notation. The book should be in EPUB format and markup should follow the new Nordic guidelines for MathML (GitHub, 2025). This should be tested across different reading systems, in particular Thorium/Readium, EasyReader, Calibre e-reader and Adobe Digital Editions. The tests should also include the use of a screen reader, in particular NVDA, JAWS and VoiceOver. The tests should be performed across different operating systems where possible. E-ink devices could also be looked at. The tests should convey whether the math elements display and are spoken correctly.

12.1.2 Use of tools in education

Launch a study to **systematically evaluate the effectiveness and user experiences of newer math accessibility tools** across different age groups and educational levels. Preferably the same study could be carried out in parallel in the Nordic countries with the results compared.

This ought to be complemented by **longitudinal studies** that examine how early exposure to accessible math tools impacts later academic and career outcomes for BVI students, including the role of institutional practices.

12.1.3 Evaluate learning management systems

Perform a **systematic evaluation** of whether **LMS:s meet WCAG:s accessibility criteria** at the AA level. Canvas merits priority because it is used by universities throughout the Nordic countries. Additional LMS:s to include are PING PONG, Unikum, Vklass and Moodle. Investigations should include screen readers and other TTS solutions, along with support for evolving standards for MathML and improved ARIA practices.

Perform a similar study **evaluating the accessibility VLE:s**, including e.g. Khan Academy, Coursera, Skillshare, Skolon and Bingel.

12.1.4 Further testing of calculators and graph programs

Launch a study to **test the accessibility limits of graph programs, calculators and equation solvers**. In particular, check integration with AT and what modes of input are available. Document areas of use for the respective tools.

- In Desmos, explore the tools “Scientific”, “Four Function”, “Matrix”, “Geometry” and “3D”.
- Explore Microsoft Excel to determine what functions are available and how easily they can be accessed using only a keyboard and a screen reader.
- Examine the features and accessibility possibilities of Isabelle Proof Assistant.
- Examine the features and accessibility possibilities of Microsoft Math Solver and similar tools.

12.1.5 Test multiline tactile displays

Purchase one or several multiline tactile displays. Primarily Monarch, Graphiti Plus and Tactonom Pro are of interest, but Dot Pad could also be relevant. Test the user experience of reading graphics in various formats, along with the overall experience of interacting with STEM e-books through the display. Features such as authoring mathematics in braille should be tested.

12.1.6 Monitor traditional tactile image techniques

Keep an eye on the results of the **study on tactile images by MTM and SPSM** (MTM, 2025). Note possible STEM applications for the techniques and perform user-centred studies on **how analogue and digital tactile graphics might complement one another**.

12.1.7 Further studies of programming interfaces

Perform **systematic, longitudinal studies that measure real-world productivity** across diverse disability profiles rather than short accessibility checklists. Especially focus on terminal integration, block-based or drag-and-drop interfaces and AI copilots or language-server features.

12.1.8 Evaluate conversion tools

Evaluate the following **tools for OCR/conversion into MathML**: MathKicker, MathPix, Mistral, Morf, Pandoc and WordToEPUB. These aspects should be taken into consideration:

- Semantic quality of markup.
- Degree of accuracy based on context.
- Legal aspects of how the material the tool is used upon is handled.
- Sustainability in terms of cost and environmental imprint.
- Possibilities of using the tool as a part of a large-scale production process.

12.1.9 Investigate Large Language Models

Launch a study to **investigate the accuracy of LLMs** in the following use cases:

- Reading math aloud from an image.
- Generating MathML from an image or dictation.
- Compatibility with various applications and AT:s.

This could be done in collaboration with e.g. Lund University where there is experience of working with LLM:s in math teaching (AI Lund, 2025). Aspects that should be taken into consideration include impact on learning outcomes, bias, copyright and resource impact.

12.2 Suggested development projects

Below are listed suggested developments which could be undertaken now, or when necessary, in combination with studies from section 12.1 relevant to the work.

12.2.1 MathCAT translations

MathCAT has the greatest potential for accessing math through a screen reader. Of the Nordic languages, MathCAT TTS currently exists in Swedish and Finnish, and there is partial support for Swedish and Finnish braille. Translations into Norwegian (Nynorsk and Bokmål), Danish and Icelandic are in progress. These **localisation efforts should be of high priority** as many students of all ages would benefit from access to spoken math in their native tongue, as well as in a braille code with which they are familiar. Not only will

MathCAT benefit screen reader users, but it can also be used for producing STEM books with pre-recorded TTS[20].

12.2.2 Future-proofing MathCAT

MathCAT was founded by Neil Soiffer, who remains the project's main developer. However, being dependent on one single developer makes the project vulnerable, and Neil Soiffer has expressed a need for support in handling the growth of MathCAT, including software maintenance, development, and project planning.

We recognise that **securing the stability and future development of MathCAT** is of strategic importance when it comes to accessible math reading and learning. Therefore, we suggest that the Nordic agencies **take an active part in securing funding and expertise** for MathCAT in the long term. An appropriate organisation to do this through would be the DAISY consortium.

12.2.3 Advocate for native MathML support in readers

Request that developers of **e-book readers develop native MathML support**. The following are especially relevant:

- Encourage EDRLab to implement native MathML support into Thorium Reader and make MathJax customization controlled by the user or very lightweight. MathML support on par with Chromium-based browsers should be developed for the mobile app Readium.
- Encourage Colibrio to improve mobile app MathML support.
- Encourage Dolphin to improve MathML support in EasyReader.

12.2.4 Advocate for use of MathML

The Nordic agencies should **encourage use of MathML** in web- and e-book environments. This could be done through referring e.g. publishers and educational institutions to current standards and guidelines on a web page, in leaflets or in talks at conferences. An example STEM e-book (see section 12.1.1) could be provided as an example of an accessible publication.

12.2.5 Develop support for writing math into e-books

Launch a project to **develop a cohesive solution for users to write math into e-books** (fill-in-the-blanks style). Desirable functionality includes (but is not necessarily limited to)

- Options for input method, including braille.
- Support for editing directly through a reading system.
- Editing features to be accessible with a screen reader, braille display and multiline tactile display.

[20] MTM is currently working on building MathCAT into their TTS book production system (YouTube, 2024).

12.2.6 Develop accessible and localised graphing tool

Develop improved graphing tools that support the Nordic languages and that are easy to use with a screen reader and keyboard.

12.2.7 Develop support for scripted SVG in e-books

The Nordic agencies should participate actively in the **development of scripted SVG images used in EPUBs**. The E-braille project (The DAISY Consortium, 2025) is already working on this. The work should investigate how the parts fit together and do user testing with cohesive solutions for creating and exploring images as a goal. Especially multiline tactile displays should be considered.

12.2.8 Develop standards for tactile graphics

The Nordic agencies should take active part in the **development of guidelines and standards for tactile graphics in STEM** worldwide. Both digital and traditional methods are of interest. The work should involve user testing and take multiline tactile displays into consideration for the digital parts.

13. References

ABILITY. (2025, May 19). Retrieved from ABILITY: https://www.ability-project.eu/ability

Accessibility. (n.d.). Retrieved April 25, 2025, from GeoGebra: https://help.geogebra.org/hc/en-us/articles/20048444963869-Accessibility

Adobe Digital Editions. (n.d.). Retrieved 5 22, 2025, from https://www.adobe.com/solutions/ebook/digital-editions/download.html

AI Lund. (2025, March 26). Retrieved from AI Lund lunch seminar: The Good, the Bad, or the Ugly? Reflections on the impact of AI in Teaching First-Semester Calculus: https://www.ai.lu.se/evenemang/ai-lund-lunch-seminar-good-bad-or-ugly-reflections-impact-ai-teaching-first-semester-calculus

APH. (2022). Retrieved from MathSpeak and ClearSpeak: Structured speech for math accessibility: www.aph.org

APH. (2025, May 19). Retrieved from Tactile Graphic Image Library: https://www.aph.org/blog/the-tactile-graphics-image-library-helping-students-succeed/

APH. (2025, May 19). Retrieved from Monarch: https://www.aph.org/product/monarch/

Armstrong, H. (2009). Advanced IT education for the vision impaired via e-learning. *Journal of Information Technology Education, 8*, 243-256.

Astronify. (2025, May 23). Retrieved from Astronify: https://astronify.readthedocs.io/en/latest/

Barda, D., Jensen, M., & Singh, K. (2023). AI-based math tutoring:evaluating real-world use of ChatGPT, Gemini and Copilot in higher education. *Journal of Educational Technology Research*, 210-232.

Bingel. (2025, June 18). Retrieved from Bingel: https://www.bingel.se/

Blind SVG. (2025, May 23). Retrieved from Blind SVG: https://blindsvg.com/

Bookshare. (n.d.). Retrieved 5 23, 2025, from https://www.bookshare.org/

Braille Authority of North America. (2012, February). Retrieved from Guidelines and Standards for Tactile Graphics: https://www.brailleauthority.org/tg/web-manual/index.html

braillesense. (2025, June 17). Retrieved from braillesense: https://www.braillesense.tech/

Bristol Braille Technology. (2025, May 19). Retrieved from About Canute: https://bristolbraille.org/about-canute/

Burke, D., Clapper, D., & McRae, D. (2016). Accessible online instruction for students with disabilities: Federal imperatives and the challenge of compliance. *Journal of Law and Education, 45*, 135-180.

Calibre - E-book management. (n.d.). Retrieved 5 22, 2025, from https://calibre-ebook.com/

Cantook by Aldiko – Google Play. (n.d.). Retrieved 5 22, 2025, from https://play.google.com/store/apps/details?id=com.aldiko.android

ChatGPT. (2025, May 27). Retrieved from ChatGPT: https://chatgpt.com/

CHROME Lab. (2025, May 23). Retrieved from Multimodal Digital Graphics on Touchscreens: https://sites.google.com/slu.edu/gorlewicz-lab/research-projects/multimodal-graphics

Chromium. (n.d.). Retrieved 5 22, 2025, from https://www.chromium.org/Home/

Clusive. (n.d.). Retrieved 5 23, 2025, from https://clusive.cast.org/

CMU Data Interaction Group. (2025, May 23). Retrieved from Chartability: https://dig.cmu.edu/data-navigator/

Colibrio Reader - Colibrio Reader. (n.d.). Retrieved 5 22, 2025, from https://www.colibrio.com/

Coursera. (2025, June 18). Retrieved from Coursera: https://www.coursera.org/

DAISY. (2025, May 23). Retrieved from Exploring Artificial Intelligence: Image DEscriptions: https://daisy.org/news-events/articles/exploring-artificial-intelligence-image-descriptions/

DAISY. (2025, May 22). Retrieved from WordToEPUB: https://daisy.org/activities/software/wordtoepub/

Desmos accessibility. (n.d.). Retrieved April 25, 2025, from Desmos: https://www.desmos.com/accessibility

DIAGRAM Center. (2025, May 23). Retrieved from Poet Training Tool: https://poet.diagramcenter.org/

Dotincorp. (2025, May 19). Retrieved from Dot Pad: https://www.dotincorp.com/en/product/pad

EasyReader App | Dolphin Computer Access. (n.d.). Retrieved 5 22, 2025, from https://yourdolphin.com/EasyReader-App

epubtest.org. (2025). Retrieved from Accessibility Tests Mathematics v 1.1.1: https://epubtest.org/test-books/math/1.1.1

epubtest.org: Test visual-550 in Visual Adjustments (2.0.0). (n.d.). Retrieved 5 23, 2025, from https://epubtest.org/test-books/visual-adjustments/2.0.0/visual-550

European Comission. (2025, May 20). Retrieved from European accessibility act: https://commission.europa.eu/strategy-and-policy/policies/justice-and-fundamental-rights/disability/union-equality-strategy-rights-persons-disabilities-2021-2030/european-accessibility-act_en

Gecko — Firefox Source Docs documentation. (n.d.). Retrieved 5 22, 2025, from https://firefox-source-docs.mozilla.org/overview/gecko.html

GitHub. (2025, May 20). Retrieved from Support HID standard multi-line braille displays: https://github.com/nvaccess/nvda/issues/16993

GitHub. (2025, May 21). Retrieved from Nordic MathML Guidelines: https://github.com/nlbdev/mathml-guidelines/blob/2024-1/Guidelines/Nordic%20MathML%20Guidelines%202024-1.md

GitHub. (2025, May 20). Retrieved from MathCAT: Math Capable Assistive Technology: https://nsoiffer.github.io/MathCAT/

GitHub. (2025, May 27). Retrieved from WordMat: https://github.com/Eduap-com/WordMat/releases

GNU. (2025, June 17). Retrieved from GNU Emacs: https://www.gnu.org/software/emacs/

Goodnotes. (2025, June 17). Retrieved from https://www.goodnotes.com/

Google Drive. (2025). Retrieved from modified added math accessibility tests mathematics: https://drive.google.com/file/d/1Dv4OvTQ_5aohbpHi5LlJh9mnlHxtffkJ/view

Google Play Books. (n.d.). Retrieved 5 23, 2025, from https://play.google.com/store/apps/details?id=com.google.android.apps.books

Gutiérrez Gómez-Calcerrada, S., Solera Hernández, E., & García González, J. (2005). Una Aproximación a La Realidad De Las Plataformas Virtuales De Las Universidades Españolas: El Primer Reto Para Una Educación Personalizada en Personas Con Discapacidad Motórica O Visual. *Enseñanza & Teaching*, 59-78.

Gyldendal. (2025, June 17). Retrieved from GEOS: https://geos.gyldendal.dk/

Gyldendal. (2025, June 17). Retrieved from Gyldendal: https://www.gyldendal.dk/

Hadian, S., & Storey, M.-A. (2005). Accessibility in a Virtual Classroom: A Case Study for the Visually Impaired Using Webct. *Proceedings of the IADIS International Conference on Cognition & Exploratory Learning in Digital Age*, (pp. 1-3).

Highcharts. (2025, May 23). Retrieved from Highcharts: https://www.highcharts.com/

Humanware. (2025, June 17). Retrieved from Braille notetakers: https://store.humanware.com/hus/braille-devices/braille-notetakers

Inclusio Community. (2025, May 23). Retrieved from Inclusio: https://inclusiocommunity.com/

Infrastructure. (2025, June 18). Retrieved from Canvas LMS for higher ed: https://www.instructure.com/higher-education/products/canvas/canvas-lms

Kamaghe, J., Luhanga, E., & Kisangiri, M. (2020). The Challenges of Adopting M-Learning Assistive Technologies for Visually Impaired Learners in Higher

Learning Institution in Tanzania. *International Journal of Emerging Technologies in Learning*, 140-151.

Kaufhold, N., & Steinert, J. (2024). Work in progress: Expanding learning opportunities in STEM courses: The potential of haptic VR laboratories for students with and without visual impairment. Berlin: Springer Nature.

Keyboard shortcuts in Excel. (n.d.). Retrieved April 25, 2025, from Microsoft support: https://support.microsoft.com/en-us/office/keyboard-shortcuts-in-excel-1798d9d5-842a-42b8-9c99-9b7213f0040f#bkmk_datawin

Khan Academy. (2025, June 18). Retrieved from Khan Academy: https://www.khanacademy.org/

Kobo | Rakuten Kobo. (n.d.). Retrieved 5 23, 2025, from https://www.kobo.com/fi/fi/p/apps

LaTeX. (2025, May 27). Retrieved from The LaTeX project: https://www.latex-project.org/

Legimus. (n.d.). Retrieved 5 23, 2025, from https://www.legimus.se/

LinkedIn. (2025, May 20). Retrieved from Project Lion Cage: https://www.linkedin.com/pulse/project-lion-cage-part-1-tor-indst%C3%B8y/

Lithium: EPUB Reader – Google Play. (n.d.). Retrieved 5 22, 2025, from https://play.google.com/store/apps/details?id=com.faultexception.reader

Lu, P., Bansal, H., Xia, T., Liu, J., Li, C., Hajishirzi, H., . . . Gao, J. (2024). MathVista: Evaluating Mathematical Reasoning of Foundation Models in Visual Contexts. *ICLR.*

Maidenbaum, S., Levy-Tzedek, S., Chebat, D.-R., & Amedi, A. (2013). Increasing accessibility to the blind of virtual environments, using a virtual mobility aid based on the "EyeCane": Feasibility study. *PLoS One, 8*.

Maple. (n.d.). Retrieved April 25, 2025, from NTNU: https://i.ntnu.no/wiki/-/wiki/English/Maple

Maplesoft. (2025, March 24). Maple Fundamentals Guide. Retrieved from https://www.maplesoft.com/support/training/quickstart.aspx

Markdown Guide. (2025, June 17). Retrieved from https://www.markdownguide.org/

Markdown Guide. (2025, June 17). Retrieved from https://www.markdownguide.org/tools/hackmd/

MatematikFessor. (2025, May 27). Retrieved from MatematikFessor: https://www.matematikfessor.dk/

MathCAT: Math Capable Assistive Technology | MathCAT. (n.d.). Retrieved 5 22, 2025, from https://nsoiffer.github.io/MathCAT/

MathJax. (2025). Retrieved 5 22, 2025, from MathJax | Beautiful math in all browsers.: https://www.mathjax.org/

MathKicker. (2025, May 22). Retrieved from MathKicker: https://mathkicker.ai/

MathML Core. (n.d.). Retrieved 5 22, 2025, from https://www.w3.org/TR/mathml-core/

MathPix. (2025, May 22). Retrieved from MathPix: https://mathpix.com/

McCabe, F. &. (2023). *Matematik och programmering En studie om läromedel, metoder och teknik som fungerar för punktskriftsanvändare.* Punktskriftsnämnden.

Microsoft. (2025, June 17). Retrieved from Math Solver: https://mathsolver.microsoft.com/en

Mistral. (2025, March 6). Retrieved from Mistral OCR: https://mistral.ai/news/mistral-ocr?fbclid=IwY2xjawJ-zypleHRuA2FlbQIxMABicmlkETBLSE12NzhPZ3Awb2JPU0hkAR5wHmgPa3CkLDWtV9Ff3ZhyEuJ0Q6FAZfkYRyhA2hKdLqKfsyV83BqiHGnUOA_aem_ebXGPItR7cM2b50YFTqKZg

MIT News. (2025, May 23). Retrieved from A new way to make graphs more accessible to blind and low-vision readers: https://news.mit.edu/2025/making-graphs-more-accessible-blind-low-vision-readers-0325

MIT News. (2025, March 25). Retrieved from A new way to make graphs more accessible to blind and low-vision readers: https://news.mit.edu/2025/making-graphs-more-accessible-blind-low-vision-readers-0325

Moodle. (2025, June 18). Retrieved from Moodle: https://moodle.com/

MTM. (2025, May 19). Retrieved from MTM och SPSM söker forskare för uppdrag med fokus på taktila bilder: https://www.mtm.se/nyheter/mtm-och-spsm-soker-forskare-for-uppdrag-med-fokus-pa-taktila-bilder/

New York Public Library. (2025, May 23). Retrieved from Dimensions: Community Tools for Making Tactile Graphics & Objects: https://www.nypl.org/about/locations/heiskell/dimensions

Nganji, J., & Brayshaw, M. (2017). Disability-aware adaptive and personalised learning for students with multiple disabilities. *The International Journal of Information and Learning Technology,* , 307-321.

Nota Bibliotek 2.0 app | Nota bibliotek. (n.d.). Retrieved 5 23, 2025, from https://nota.dk/services/nota-bibliotek-20-app

Notability. (2025, June 17). Retrieved from Notability: https://notability.com/en

Notion. (2025, June 17). Retrieved from https://www.notion.com/product

Orbit Research. (2025, May 19). Retrieved from Graphiti: https://www.orbitresearch.com/product/graphiti/

Orbit Research. (2025, May 19). Retrieved from Graphiti Plus: https://www.orbitresearch.com/product/graphiti-plus/

Oribi. (2025). Retrieved from Equatio: https://www.oribi.se/produkter/equatio/

Overleaf. (2025). Retrieved from www.overleaf.com

Pandoc. (2025, May 22). Retrieved from Pandoc - a universal document converter: https://pandoc.org/

Pandoc. (2025, May 22). Retrieved from Pandoc - a universal document converter: https://pandoc.org/

Park, K., So, H.-J., & Cha, H. (2019). Digital equity and accessible MOOCs: Accessibility evaluations of mobile MOOCs for learners with visual impairments. *Australasian Journal of Educational Technology*, 48-63.

Pcmacstore. (2025, June 17). Retrieved from MathPad for PC and Mac: https://pcmacstore.com/en/software/1146553245/mathpad

Photomath. (2025). Retrieved from https://photomath.com/

Ping pong. (2025, June 18). Retrieved from Ping pong: https://www.pingpong.se/

Pollack, I., & Ficks, L. (1954). The Information of Elementary Multidimensional Auditory Displays. *The Journal of the Acoustical Society of America*.

Pratsam Reader App – Pratsam. (n.d.). Retrieved 5 22, 2025, from https://www.pratsam.com/pratsam-reader-app-tuote.html

ProBlind. (2025, May 19). Retrieved from ProBlind Database: https://www.problind.org/en/

ProBlind. (2025, May 23). Retrieved from ProBlind: https://www.problind.org/en/

Q42. (2025, mAY 23). Retrieved from SenseMath: https://www.q42.nl/en/work/sensemath-app

ReadSpeaker. (n.d.). *Bring natural text to speech to any content or application*. Retrieved from ReadSpeaker: https://www.readspeaker.com/

Ready, S. (2017). Making Virtual Learning Accessible for the Visually Impaired. *Exceptional Parent*, 36-37.

RedShelf. (n.d.). Retrieved 5 23, 2025, from https://www.redshelf.com/

Riley-Ancar, H. (2022). *Accessibility challenges of online learning affecting successful degree completion among visually impaired Undergraduate/Graduate students: An exploratory case study.* Social Science Premium Collection. doi:https://www.proquest.com/docview/2728985279

Rønning, Ø. (2020, September 23). *Scratch*. Retrieved from GitHub: https://github.com/oivron/coding-without-seeing/wiki/Scratch

Rønning, Ø. (2021, January 20). *Jupyter Notebook og JupyterLab*. Retrieved from GitHub: https://github.com/oivron/coding-without-seeing/wiki/Jupyter-Notebook-og-JupyterLab

Rønning, Ø. (2022, June 29). *PyCharm*. Retrieved from GitHub: https://github.com/oivron/coding-without-seeing/wiki/PyCharm

Rønning, Ø. (2022, June 29). *PyCharm*. Retrieved from GitHub: https://github.com/oivron/coding-without-seeing/wiki/PyCharm

Rønning, Ø. (2023). *coding-without-seeing*. Retrieved from GitHub: https://github.com/oivron/coding-without-seeing

Rønning, Ø. (2023, February 20). *NotepadPlusPlus*. Retrieved from GitHub: https://github.com/oivron/coding-without-seeing/wiki/NotepadPlusPlus

Rønning, Ø. (2024, January 12). *Visual Studio Code*. Retrieved from GitHub: https://github.com/oivron/coding-without-seeing/wiki/Visual-Studio-Code

Schlichtkrull, A. (2018). Formalization of Logic in the Isabelle Proof Assistant. *DTU Compute PHD-2018, 493*. Retrieved from https://findit.dtu.dk/en/catalog/5c18d34ad9001d015134d162

Seeing AI. (2025, May 23). Retrieved from Seeing AI: https://www.seeingai.com/

SenseMath - Making Sense of Math. (n.d.). Retrieved April 25, 2025, from Enviter: https://enviter.eu/sensemath-making-sense-of-math/

Skillshare. (2025, June 18). Retrieved from Skillshare: https://www.skillshare.com/en/

Skolon. (2025, June 18). Retrieved from Skolon: https://skolon.com/sv/hem/

SnapChat. (2025, May 27). Retrieved from SnapChat: https://www.snapchat.com/

Sorge, V. (2023). *W3C*. Retrieved from MathML accessibility and the MathCAT library: www.w3.org

Sourceforge. (2025, June 17). Retrieved from TeXworks: https://sourceforge.net/projects/texworks.mirror/

Statped. (2025). *Programmering for elever med nedsatt syn og blindhet | statped.no*. Retrieved from Statped | statped.no: https://www.statped.no/laringsressurser/syn/temaside-programmering-for-elever-med-nedsatt-syn-temaside/

Symbolab. (2025, May 27). Retrieved from Symbolab: https://www.symbolab.com/

Systime. (2025, June 17). Retrieved from https://systime.dk/

Tactile Engineering. (2025, May 16). Retrieved from The Cadence Tablet: https://www.tactile-engineering.com/cadence

Tactonom. (2025, May 19). Retrieved from Tactonom Pro: https://www.tactonom.com/en/tactonom-pro/

Technische Universität München. (2025, June 17). Retrieved from Isabelle: https://isabelle.in.tum.de/

Texas Instruments. (2025, May 27). Retrieved from TI-nspire: https://education.ti.com/sv/produkter/raknare/grafraknare/ti-nspire-cx-cas

TeXstudio. (2025, June 17). Retrieved from TeXstudio - A LaTeX editor: https://www.texstudio.org/

The DAISY Consortium. (2025, May 22). Retrieved from Braille File Formats: https://daisy.org/activities/projects/ebraille/

The DAISY Consortium. (2025, May 22). Retrieved from DAISY Consortium Position statement on MathML: https://daisy.github.io/transitiontoepub/information-sharing/position-paper-plain-text-math/

Thorium Reader. (n.d.). Retrieved 5 22, 2025, from https://thorium.edrlab.org/en/

TouchPad Pro Foundation. (2025, May 19). Retrieved from BrailleDoodle: https://www.touchpadprofoundation.org/

UNAR labs. (2025, May 22). Retrieved from Morf: https://www.unarlabs.com/

Unikum. (2025, June 18). Retrieved from Unikum: https://www.unikum.net/

Visual Studio Code. (2025, June 17). Retrieved from Visual Studio Code: https://code.visualstudio.com/

Vklass. (2025, June 18). Retrieved from Vklass: https://www.vklass.com/

W3C. (2024, 12 12). Retrieved 5 23, 2025, from Web Content Accessibility Guidelines (WCAG) 2.2: https://www.w3.org/TR/WCAG22/

W3C. (2025, May 19). Retrieved from Draft of SVG standard: https://www.w3.org/TR/SVG/interact.html

W3C. (2025, May 19). Retrieved from EPUB 3.3 standard: https://www.w3.org/TR/epub-33/#sec-scripted-content

W3C. (2025, May 23). Retrieved from EPUB 3.3 W3C Recommendation: https://www.w3.org/TR/epub-33/#sec-xhtml-svg

W3C. (2025, May 20). Retrieved from Mathematical Markup Language: https://www.w3.org/TR/MathML/

W3C. (2025, May 26). Retrieved from WCAG 2.0 Accessibility Criteria: https://www.w3.org/TR/WCAG20/

Walker, B. N., & Nees, M. A. (2011). Theory of Sonification. In A. H. Thomas Hermann, *The Sonification Handbook* (pp. 9-39). Berlin: Logos Publishing House.

WebKit. (n.d.). Retrieved 5 22, 2025, from https://webkit.org

What is GeoGebra? (n.d.). Retrieved from GeoGebra: https://www.geogebra.org/about

What is GeoGebra? (n.d.). Retrieved April 25, 2025, from GeoGebra: https://www.geogebra.org/about

WinEdt. (2025, June 17). Retrieved from WinEdt 11: https://www.winedt.com/

Wolfram Alpha. (2025, May 27). Retrieved from Wolfram Alpha: https://www.wolframalpha.com/

Yan, Y., Su, J., He, J., Fu, F., Zheng, X., Lyu, Y., . . . Hu, X. (2025). A Survey of Mathematical Reasoning in the Era of Multimodal Large Language Model: Benchmark, Method & Challenges. *ACL Findings.*

YouTube. (2024). Retrieved from DPS2024 - 09 - Math books with synthetic speech - Tim Arborealis, Karl Adam Tiderman: https://www.youtube.com/watch?v=J2ba5JPvkY0&list=PLQAmHah-XQKz_GrPsucRHO5oHGkp8Kk_O&index=11