Flick-Input TeX Interface: Lowering Barriers to Mobile Equation Entry

Takuya Kitamoto
kitamoto@yamaguchi-u.ac.jp
Faculty of Education
Yamaguchi University
753-8513
Japan

Hisashi Usui
usui@gunma-ct.ac.jp
Gunma College
National Institute of Technology
371-8530
Japan

Naoki Hamaguchi
hama@nagano-nct.ac.jp
Nagano College
National Institute of Technology
381-8550
Japan

Masataka Kaneko
masataka.kaneko@phar.toho-u.ac.jp
Faculty of Pharmaceutical Sciences
Toho University
274-8510
Japan

Takeo Noda
noda@phar.toho-u.ac.jp
Faculty of Science
Toho University
274-8510
Japan

Abstract

Recent advances in flick-based input methods for smartphones have demonstrated significant improvements in the speed and usability of both mathematical and textual entry compared to traditional on-screen keyboards. While TeX remains the de facto standard for precise representation of complex formulas – particularly when submitting equations to large language models such as ChatGPT – it imposes burdens on learners: the requirement to memorize TeX commands—such as \sqrt and \frac—and the necessity of understanding TeX's grammatical rules.

To address these challenges, we introduce a novel interface that harnesses flick gestures to invoke common TeX commands effortlessly. Users perform directional swipes to select macros, enabling rapid entry of both prose and mathematical expressions with dramatically fewer touch events. The layout can be customized to accommodate specialized command sets for various mathematical subfields, and real-time TeX conversion coupled with instant KaTeX rendering provides immediate visual feedback. A classroom trial with undergraduate students in a set theory course revealed that over 90% of participants – 70% of whom had never previously used TeX – successfully completed formula-entry tasks after minimal practice, and approximately 65% felt confident using the system for assignment submissions. By combining flick-input's efficiency with TeX's expressiveness, our approach lowers barriers for novices, accelerates mobile math authoring, and paves the way for seamless integration with AI-assisted learning workflows.

1 Introduction

Recent years have seen a surge of interest in flick-based input methods on smartphones, aimed at improving the speed and usability of both mathematical and textual entry in mobile contexts. Studies have demonstrated that flick operations can dramatically reduce the number of touch events needed to enter complex expressions and text, yielding faster input and higher user satisfaction compared to conventional on-screen keyboards ([1]-[2]). Usability evaluations across diverse user groups – including students, older adults, and specialists – consistently find that, after a brief learning period, flick interfaces rival or surpass tap-based alternatives in efficiency and ease of use ([3]). Moreover, adaptations of flick input for niche form factors (e.g., smartwatches) and non-touch interfaces highlight its broad applicability in next-generation mobile and wearable learning environments ([4]).

The accurate rendering of mathematical expressions in plain text is a nontrivial challenge; however, by leveraging TeX, even highly complex formulas can be represented precisely within a single sentence, and TeX has become the de facto standard for submitting equations to large language models such as ChatGPT. Accordingly, requiring learners to input mathematical content in TeX format within an E-Learning environment constitutes a highly promising approach. Nevertheless, this strategy presents two primary drawbacks: first, encoding formulas in TeX necessitates entering command sequences – e.g., \frac and \sqrt – which increases the volume of text input; second, users are required to memorize the repertoire of TeX commands, such as \frac and \sqrt, thereby imposing an additional cognitive load.

In this paper, we propose an interface that enables the effortless entry of TeX commands via flick-based gestures. The key features of the proposed system are as follows:

1. **Efficient Text Entry** : By leveraging flick input, users can enter both ordinary text and TeX command sequences with greater speed and fewer keystrokes than with a conventional on-screen keyboard.

2. **No Command Memorization Required** : End users need not commit TeX macros (e.g. '\frac', '\sqrt') to memory; instead, they can input complex formulae through intuitive swipe gestures.

3. **Customizable Key Mappings** : The flick-input layout supports easy reconfiguration, allowing instructors or learners to tailor gesture-to-command assignments for specialized TeX libraries or distinct mathematical subfields.

In this paper, we present an interface that enables the straightforward entry of TeX commands via flick gestures. The proposed system offers three main advantages: (1) by leveraging flick input, it supports highly efficient text entry; (2) it allows users to input mathematical expressions accurately without the need to memorize TeX command syntax; and (3) its flick-input key mappings can be easily reconfigured, thereby accommodating the specialized TeX commands used across various fields of mathematics.

This paper is organized as follows. First, Section 2 provides an overview of the flick-input system. Next, Section 3 introduces applications of the flick input system developped, and Section 4 presents the results of a student questionnaire conducted during classes using the system. Section 5 then discusses application examples integrating the system with large language models, and Section 6 offers concluding remarks.

2 Overview of flick input system

The present flick input system is built upon the E-Learning frameworks described in References [6]-[9] and on KeTCindyJS and KeTMath, as developed by Professor Takato. For comprehensive technical details concerning KeTCindyJS, the reader is directed to Reference [10]. The flick input system uses a flick keyboard for input due to the following advantages:

- Buttons can be made larger to facilitate easier typing on small screens, such as on smartphones.
- Mathematical symbols of the same type can be grouped into a single button.
- It can be easily customized according to the situation.

When using flick input, characters or other elements are selected by holding down a button and moving, and input is determined by the timing of release. The functionality could not be implemented into KeTCindyJS buttons, and HTML5 buttons cannot beautifully display mathematical symbols. Therefore, instead of using buttons, a method was adopted to process by recognizing the position of clicks or touches.

Mainly, the CindyScript command "mouse()" is used to determine positions. However, "mouse()" does not function on buttons or within input frames and retains the values from the previous point instead. When using a mouse, values can continuously be obtained outside buttons and input frames, making it usable. However, for devices with touch panels like smartphones, if the next touch is on a button or within an input frame, the position information of the previous touch is retained. This results in a phenomenon where touching a button or input frame after entering text causes text input to occur again. To address this, position tracking via HTML5 was also utilized.

The code related to KeTCindyJS is written in CindyScript, while the code concerning HTML5 is written in JavaScript. By writing "cdy.evokeCS(∗1)" in JavaScript, "∗1" is executed as CindyScript. Likewise, by writing "javascript(∗2)" in CindyScript, "∗2" is executed as JavaScript. This enables seamless information exchange between KeTCindyJS and HTML5.

Formulas and other inputs are entered in KeTMath format. KeTMath was developed with the aim of enabling the swift and straightforward transmission and reception of mathematical expressions using minimal data. KeTMath utilizes a simplified mathematical expression format based on the output method of KaTeX. The overview of Flick input is as follows.

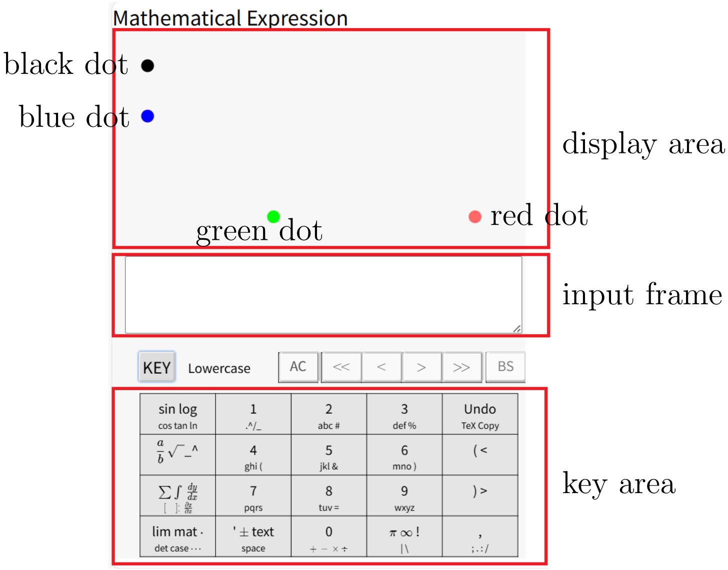

Fig. 1: screen

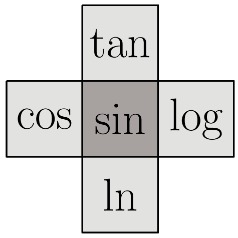

Fig. 2: cross-shaped figure

When you click or touch anywhere on the key area, according to the region, a cross-shaped figure, as shown in Fig. 2, appears at the red dot located in the bottom-right corner of the display area. By holding and moving, the color of the selected item changes, and at the moment you release, the input is completed. The entered content, which is written in KeTMath format, will be displayed in the input frame at the center of the screen, and the converted TeX formula will be shown at the black dot at the top of the display area. When "//" is entered, a line break occurs, and it is displayed at the blue dot below the black dot. The spacing between the black dot and the blue dot can be adjusted, and from the third line onward, it will be displayed with that spacing. When entering matrices, integral expressions, or similar items, an input assistant display appears at the green dot at the bottom of the display area. A "?" mark is shown at the input position.

Three arrays, array1, array2 and array3 are prepared based on the type of characters, corresponding to lowercase alphabets, uppercase alphabets, and Greek letters, respectively. These can be cyclically switched by pressing the "KEY" button. Each array contains information corresponding to the 20 regions of the key area. Code 1 presents the program configuration used to generate the screen shown in Fig. 1.

Code 1: array1

```
array1=[
[["sin log","cos tan ln"],["sin","cos","tan","log","ln"],["sin(?)","cos(?)
    ","tan(?)","log(?)","ln(?)"]],
[["1",".^/_"],["1",".","^","/","_"],["1",".","^","/","_"]],
[["2","abc #"],["2","a","b","c","#"],["2","a","b","c","\#"]],
[["3","def %"],["3","d","e","f","%"],["3","d","e","f","\%"]],
[["Undo","TeX Copy"],["Undo","","Tex","","Copy"],["Delete()","Delete()","
    copytex()","Delete()","copydata()"]],
[["$\dfrac{a}{b}\ \sqrt{\phantom{a}}$_^",""],["$\dfrac{a}{b}$","$\frac{a}{b}
    $","$\sqrt{\phantom{a}}$","^","_"],["fr(?,)","tfr(?,)","sq(?)","^(?)","_
    (?)"]],
[["4", "ghi ("],["4","g","h","i","("],["4","g","h","i","("]],
```

```
[["5","jkl &"],["5","j","k","l","&"],["5","j","k","l","\&"]],
[["6","mno )"],["6","m","n","o",")"],["6","m","n","o",")"]],
[["( <",""],["(","{","[","<","$\leq$"],["(","\{","[","<","(leq)"]],
[["$\sum \int \ \frac{dy}{dx}$","$[\phantom{aa}]_\cdot^\cdot \ \ \frac{\
   partial z}{\partial x}$"],
["$\int$","$\sum$","$[\phantom{a}]_\cdot^\cdot$","$\frac{dy}{dx}$","$\frac{\
   partial z}{\partial x}$"],
["int(?,,,)","sum(?,,)","br(?,,)","diff(?,)","par(?,)"]],
[["7","pqrs"],["7","p","q","r","s"],["7","p","q","r","s"]],
[["8","tuv ="],["8","t","u","v","="],["8","t","u","v","="]],
[["9","wxyz"],["9","w","x","y","z"],["9","w","x","y","z"]],
[[") >",""],[")","}","]",">","$\geq$"],[")","\}","]",">","(geq)"]],
[["lim mat $\cdot$","det case $\cdots$"],["lim","mat","det","case","$\cdot$
   "],["lim(?,,)","mat(?,;,)","det(?,;,)","case(?,;,)","dot()"]],
[["' $\pm$ text","space"],["$\pm$","$\mp$","'","text","space"],["(pm)","(mp)
   ","'","tx(?)","(sp)"]],
[["0","$+ \ - \ \times \ \div$"],["0","-","+","$\times$","$\div$
   "],["0","-","+","\times ","\div"]],
[["$\pi\ \infty$ !","| \"],["$\pi$","$\infty$","!","|","\"],["pi","(inf)
   ","!","|","\"]],
[[",","; . : /"],[",",";",":",".","/"],[",",";",":",".","/"]],
[["",""],["","","","",""],["","","","",""]]
];
```

By simply rewriting this, you can customize the display format and the contents to be input as desired.

If you click the upper-right region, you can select “Copy” or “TeX”. When “Copy” is selected, the content inside the input frame will be saved to the clipboard. Using “Copy” allows you to paste and interact in KeTMath format. When “TeX” is selected, the converted TeX code will be saved to the clipboard. Using “TeX” allows you to paste and obtain equations in TeX format, which can also be used for questions to generative AI.

3 Applications of the Flick Input System

In this chapter, we describe the construction of a flick-input page dedicated to set theory and the formula-entry system that it enables.

3.1 System Features

The proposed system offers the following characteristics:

1. **Customized Flick-Input Page for Set Theory** : We have designed and implemented a flick-input interface tailored specifically to set-theoretic notation. By prearranging frequently used symbols and operators, users can enter common set-theory expressions with a single gesture, greatly accelerating input.

2. **Real-Time TeX Rendering from Flick Gestures** : Users perform intuitive flick gestures on a touchscreen device, and the system maps each gesture to its corresponding

TeX command. The resulting TeX expression is immediately rendered via the KaTeX engine, providing instant visual feedback and simplifying correction and verification.

3. **No Prior TeX Knowledge Required** : Traditional TeX entry demands familiarity with macro syntax, posing a barrier for novices. In contrast, our flick-based design obviates the need to memorize commands: users generate precise TeX code through simple gestures, combining ease of use with accurate formula conversion.

3.2 System Architecture and Implementation Overview

3.2.1 Design of the Input Interface

Symbol Entry via Flick Gesture : The left-hand panel of Fig. 3 illustrates the flick-input interface tailored for set-theoretic notation. The layout is optimized for touch interaction so that users can select desired symbols simply by flicking the on-screen keys. This design significantly reduces input time and minimizes entry errors.

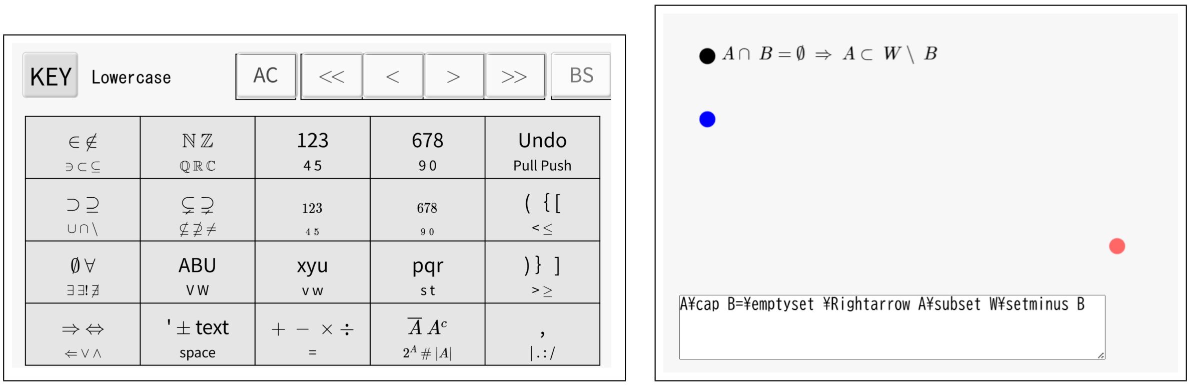

Fig. 3 : Interface for set-theoretic notation (left), set-theoretic formula inputs (right)

Dedicated Layout Configuration : By centering the most frequently used set-theoretic symbols (e.g., ∈, ∪, ∩, ⊆, ⊇) within an optimized key arrangement, the interface enhances both operational efficiency and overall usability for the user.

3.2.2 Automatic Conversion and Rendering Engine

TeX Conversion Engine : Each flick gesture performed by the user is captured and automatically mapped to the corresponding TeX command. This conversion process eliminates the need for users to possess prior TeX expertise, enabling them to generate precise formula representations through intuitive gestures.

Real-Time Rendering : The generated TeX code is immediately passed to the KaTeX rendering engine, which produces a visual representation of the formula on the screen (refer to the right-hand panel of Fig. 3). Instant visual feedback ensures that users can quickly verify correctness and make any necessary adjustments.

4 Trial Testing of the Flick-Input System and Results

We conducted a trial test of the flick-input system with students enrolled in a set theory course. Participants were asked to enter the two set-theoretic expressions shown in Fig.4 using the system and subsequently complete a questionnaire.

$$A \cap B = \emptyset \Rightarrow A \subset W \setminus B$$

$$A = \{ p \in \mathbb{N} \mid p \leq 10 \}$$

Fig. 4 : The set-theoretic expressions to be entered

4.0.1 Questionnaire Overview

The test featured five questions, and we measured the distribution of student responses for each. The questions and key findings are summarized below:

Question 1: How frequently do you use flick input versus keyboard input when entering text on a smartphone?

Always flick input: 72.5%, Always keyboard input: 10%, Depends, but mostly flick input: 15%, Depends, but mostly keyboard input: 2.5%, Use both equally: 0%

These results indicate that the majority of students regularly rely on flick input for text entry on their smartphones.

Question 2: What is your level of familiarity with TeX?

Can use TeX: 0%, Have heard of TeX: 30%, Hearing about TeX for the first time: 70%

Approximately 70% of students were encountering TeX for the first time, underscoring the limited prior knowledge and experience among participants at the time of system introduction.

Question 3: Were you able to input mathematical expressions smoothly using the system?

Very smoothly: 40%, Some difficulty, but managed: 50%, Considerable difficulty, but managed: 10%, Could not complete: 0%

Most students completed the input “very smoothly” or with “some difficulty,” suggesting that core functionality operates effectively, though minor usability issues remain.

Question 4: Self-assessment of ability to submit assignments using this system

No problem: 35%, Difficult this time, but manageable with practice: 30%, Seems difficult: 27.5%, Absolutely impossible: 7.5%

These responses reveal that while approximately 65% of students believe they could use the system effectively after gaining familiarity, about 35% still feel uneasy about its current usability.

Question 5: How would you rate the system's overall usability on a 0–100 scale?

80–100: 17.5%, 60–80: 50%, 40–60: 25%, 20–40: 5%, 0–20: 2.5%

Overall, 67.5% of students rated the system at 60 or above, indicating generally favorable evaluations of its basic functionality and convenience.

4.0.2 Discussion

The results of the trial test reveal several key insights:

- **Lack of TeX Familiarity and System Significance** : With 70% of participants encountering TeX for the first time, the high barrier presented by traditional formula-entry methods was effectively eliminated by our system. This finding underscores the value of an intuitive, gesture-based input mechanism that requires no prior command-syntax knowledge
- **Smoothness of Entry and Practicality for Assignment Submission** : Nearly 90% of students managed to complete the formula-entry tasks, demonstrating that the system's core functionality is robust. However, feedback on assignment readiness indicated that further practice and refinements are needed. Future work should address remaining usability issues and enhance system stability based on user feedback.
- **Overall Evaluation and Future Directions** : Although the usability ratings were generally positive, some students still perceived the system as challenging. To support broader adoption, we must develop comprehensive user guides and tutorials and further refine the UI. In preparation for deeper integration with generative AI tools, establishing a strong user-support framework will be critical.

In summary, the trial confirms that a dedicated flick-input system for set theory is fundamentally effective and offers practical value for students. While its intuitive design is especially advantageous for TeX novices, additional simplification of operations and expansion of user-assistance features will be necessary to support real-world assignment workflows. Ongoing system enhancements and deployment in actual teaching scenarios are expected to further improve user satisfaction and practical utility.

5 Integration with LLM

In this section, we introduce an instructional tool that integrates a large language model (LLM) with a flick-input system. Fig. 5 depicts the tool's interface: users enter mathematical expressions via flick gestures and, upon clicking the "Differentiate" button, the step-by-step differentiation process is displayed beneath the input area.

This tool has three principal features. First, as noted above, users are not required to memorize any LaTeX commands; they can input formulas naturally through flick gestures. Second, the entered expressions are rendered on-screen in their proper mathematical form,

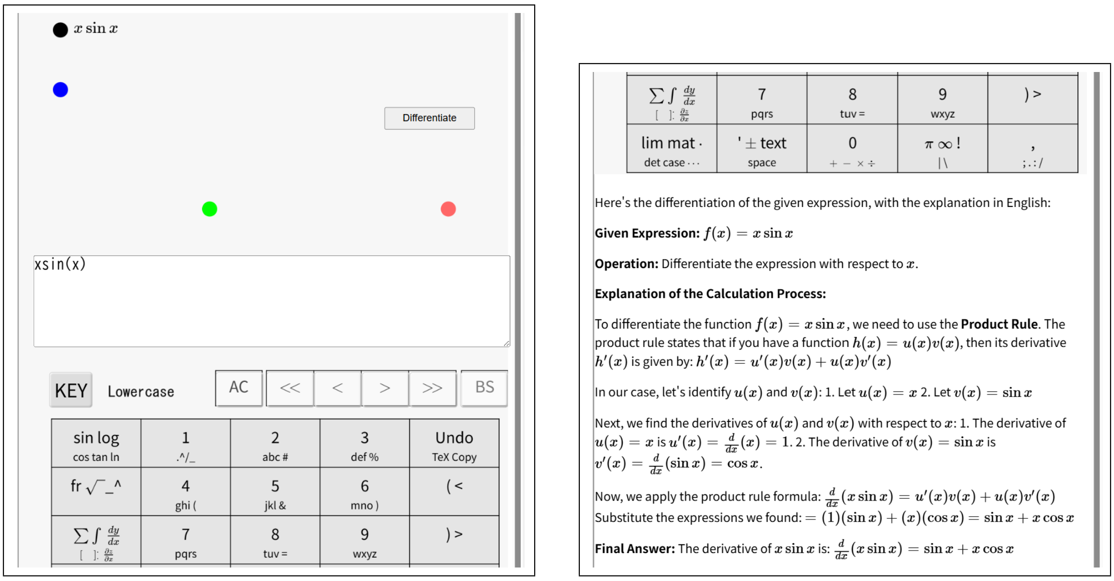

Fig. 5 : Top and bottom views of the flick-input, LLM-powered instructional tool

allowing users to verify and, if necessary, immediately correct their input (real-time rendering is performed using the JavaScript library KaTeX). Finally, by leveraging an LLM, the system can provide not only the final result but also a detailed explanation of the calculation process, as shown in Fig. 5. In operation, flick-input expressions are converted to LaTeX format and sent via API to the LLM: the tool then receives the computed steps and displays them in the interface.

Together, these features enable users – without any prior knowledge of LaTeX – to investigate the differentiation of a wide range of mathematical expressions.

6 Conclusion

In this paper, we have detailed the design of a flick-input system dedicated to set theory and reported the results of its classroom trial as a novel approach to lowering the barriers of traditional formula-entry environments while pursuing practical applicability in educational settings. Survey responses revealed that most students, already familiar with flick input, could accurately enter mathematical expressions through intuitive gestures – even without prior TeX experience – yet also indicated room for improvement in usability and assignment-submission workflows. Furthermore, integration with generative AI promises to expand educational support capabilities, such as automated problem generation and solution explanation.

Looking ahead, further refinement of the user interface and exploration of advanced, automated AI integration are planned. At the same time, systematic collection of feedback from actual classroom deployments will be essential to iteratively enhance both instructional materials and system functionality, and to investigate applicability beyond set theory to other mathematical and interdisciplinary domains.

Overall, this study represents a first step toward realizing an innovative educational environment that transcends traditional learning modalities by uniting intuitive flick-input and formula-entry tools. By actively incorporating user feedback and deepening system interconnectivity, we aim to elevate this platform into a fully mature support tool for teaching and research.

Acknowledgment

This work was supported by JSPS Grant in Aid for Scientific Research (C) KAKENHI 25K06706 and 21K02752.

References

[1] Nakamura, Y. and Nakahara, T., "Development of a Math Input Interface with Flick Operation for Mobile Devices", *12th International Conference Mobile Learning*, Portugal, 2016.

[2] Nakamura, Y., "A New Mathematics Input Interface with Flick Operation for Mobile Devices", *MSOR Connections*, 15(2):76, January 2017, DOI:10.21100/msor.v15i2.413.

[3] Nakamura, Y., "Usability Evaluation of Mobile Interfaces for Math Formula Entry", *21st International Conferences e-Society and 19th Mobile Learning*, Portugal, 2023.

[4] Akamine, K., Tsuchida, R., Tsuneo Kato, T. and, Tamura, A., "PonDeFlick: A Japanese Text Entry on Smartwatch Commonalizing Flick Operation with Smartphone Interface", *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pp. 1-11, 2024.

[5] Takato, S and Makishita, H, "Development of a Question Distribution and Answer Collection System for Mathematics Classes Using Only One Line of Text", *Proc. of The 29th Asian Technology Conference in Mathematics*, Yogyakarta, Indonesia, 2024.

[6] Kitamoto, T., Kaneko, M. and Takato, S., "E-learning system with Computer Algebra based on JavaScript programming language", *Proc. of The 23rd Asian Technology Conference in Mathematics*, Yogyakarta, Indonesia, 123–132, 2018.

[7] Kitamoto, T., Kaneko, M. and Takato, S., 'Standalone web application for teachers to create teaching materials on a browser", *Proc. of The 25rd Asian Technology Conference in Mathematics*, Bangkok, Thailand, 258–267, 2020.

[8] Kitamoto, T., "Framework for building an E-Learning system using existing teaching materials", *14th MathUI Workshop 2023*, September 6-7,Cambridge, UK, 123–132, 2023.

[9] Kitamoto, T., Kaneko, M., Noda, M. and Kihara H., "E-Learning material creation system that utilizes existing teaching materials", *Proc. of The 29th Asian Technology Conference in Mathematics*, Yogyakarta, Indonesia, 2024.

[10] KeTCindy official homepage, URL https://s-takato.github.io/ketcindyorg/indexe.html (Accessed July 19, 2025)