Tinkerable Augmentative and Alternative Communication for Users and Researchers

Boyin Yang and Per Ola Kristensson, University of Cambridge

Abstract: Augmentative and alternative communication (AAC) users are highly diverse as their communication needs depend on many factors, such as motor ability, cognition, age, education level, and overall preferences. This chapter introduces the concept of *tinkerability* for AAC. We conjecture that *Tinkerable AAC (TAAC)* has the potential to empower AAC users, developers, and researchers to fine-tune, adapt, and explore better communication solutions *in tandem* with the system. We present a framework for tinkerable AAC that concretely links qualities of a tinkerable AAC system to AAC research and development research areas and subareas. To demonstrate the viability of tinkerable AAC, we present a tinkerable predictive text entry system for AAC. It allows users to adjust many aspects of the system during use, ranging from the text prediction algorithms in the back end to the parameters underpinning the keyboard layout in the front end. In addition, it also allows AAC system developers to adopt state-of-the-art language models and enables AAC researchers to carry out *in-situ* research experiments. We discuss the design space opened up by tinkerable AAC and its potential in empowering the AAC community to co-design individually tailored AAC solutions for users.

1 Introduction

Tinkering is an exploratory and iterative style of working where a user engages with a problem or a project. It is regarded as the opposite of *planning*, which is a more organized activity that is more direct but not repetitious (Resnick and Rosenbaum 2013).

Tailoring is an activity that modifies a system to fit specific use situations. It is distinct from *use*, as *use* refers to modifications made to the subject matter of the tool while *tailoring* refers to modifications of the tool itself. While *use* can give rise to an immediate perceived effect, the effect of *tailoring* is felt at a later stage (Henderson and Kyng 1995).

In this paper we suggest *tinkerability* to be the ability of a system to let it be modified during interaction to fit different use scenarios and conditions required for users to reach specific goals, such as gaining higher efficiency. The modifications are not necessarily executed by the user directly as we envision that they can also be achieved by the assistance of AAC specialists, developers, and researchers.

We see many reasons why thinking about tinkerability in AAC is a promising research direction. First, AAC users form a very heterogeneous user group and, as a consequence, individual preferences, capabilities, needs, and resources vary substantially. Allowing AAC users, specialists, and researchers to tinker with AAC systems provide opportunities to fine-tune systems for individuals.

Second, despite decades of research, AAC solutions are still suboptimal. Tinkerability encourages exploration and may serve two purposes: (1) increasing people's interest in improving AAC systems by explicitly observing the capabilities of sophisticated AAC systems; and (2) assisting AAC designers, developers, and researchers in eliciting user requirements by empowering AAC users to further understand the capabilities and limitations of the systems they rely on.

Third, AAC systems relying on natural language processing and machine learning are becoming increasingly complex. Tinkerable AAC may help AAC designers and researchers in understanding the increasing design space of AAC systems that has emerged due to the tremendous progress in artificial intelligence. By letting AAC designers experiment with AAC systems, tinkerable AAC allows them to observe potential benefits from such advancements first hand, thereby tinkerable AAC has the potential to empower both AAC users and professionals, stimulating new ideas and new formulations of design problems in AAC.

Thus, tinkerable AAC is a potential enabling technology, allowing non-experts in AI to create new futures within a broader AAC design space that encompasses state-of-the-art AI.

In the remaining space of this paper we will further explain tinkerability in AAC, present a framework for tinkerable AAC, and exemplify tinkerable AAC by presenting a tinkerable text prediction AAC system we have built for this purpose.

2 Tinkerability in AAC

AAC technology is used to assist people with communication disabilities to develop or regain their competence. Many computer-based AAC systems are developed using numerous representations for vocabulary concepts, including photographs, symbols, written words, letters of alphabet, and so on (Beukelman and Light 2020). Due to their physical capabilities, these representations can be accessed by fingers, toes, elbows, eyes, or use multimodal approaches.

In the past decades, social inclusion has greatly increased (Mirenda 2014). As a result, literate nonspeaking individuals with motor disabilities tend to adopt a letter-by-letter spelling strategy to communicate as it not only provides a more precise meaning of expression than images or symbols, but also brings a much wider range of expressions than preset text (Beukelman and Light 2020). Text prediction, including word prediction (Vertanen and Kristensson 2011), phrase prediction (McKillop 2018), sentence retrieval (Kristensson, et al. 2020), and sentence generation (Shen, et al. 2022) have been used to augment the language composition function of the AAC system, where vocabulary selection and message management are key points of system interaction.

Moreover, some AAC users, such as amyotrophic lateral sclerosis (ALS) patients, have progressive diseases which means their physical capabilities, and thus, AAC needs, may change over time. For example, an ALS patient may originally type using a physical keyboard and later rely on eye gaze to communicate using an eye-typing system. Switching from one particular AAC system to a new one incurs upfront learning costs from both the user and the system, the latter which likely needs to be configured and adapted to the user. This can be frustrating and challenging for AAC users, in particular, as AAC systems tend to be presented as black boxes. While tinkerability will not remove any costs in switching systems it may help reduce frustration by allowing exploration of a new system to best fit new needs and wants.

In terms of what can be tinkered with, a computer-based system with a user interface (UI) can be partitioned into two components: the front end—what the user interacts with, and the back end—necessary system logic, such as prediction algorithms and language models. A tinkerable AAC system allows users to manipulate both the front and back end to meet their requirements and satisfy their curiosity in exploring the design space of the AAC system.

However, even though a tinkerable AAC system would be beneficial, it remains difficult to develop one. Prior work (Henderson and Kyng 1995, Ellis, et al. 2021) identifies four key points about developing a general tinkerable system. First, it needs to balance complexity and tinkerablility, as tinkering requires additional buttons, switches, and mechanisms for adding behavior to code. Second, it becomes necessary to provide mechanisms for allowing design in use. Third, the system must support preserving and reestablishing the state of the system. Fourth, the three prior points all increase resource requirements in terms of time, money, experience, etc. to provide a working robust tinkerable AAC system.

There is also the challenge that the sheer complexity of state-of-the-art natural language processing systems means that the underpinning AI may be difficult to understand for users, designers, researchers, and developers alike. To allow a target audience with widely different educational backgrounds and interests to meaningfully tinker with a state-of-the-art word prediction and sentence prediction system, supporting sentence retrieval and sentence generation, requires careful considerations of which parameters to expose, how resulting system behavior can be analyzed and communicated to end-users, and how sometimes perplexing AI behavior can be explained to users.

3 Tinkerable AAC Framework

To illustrate how a TAAC system contributes to AAC research and development, we propose a framework (Fig. 1.) that links areas of AAC research and development to the qualities of a TAAC system. These links are indicated as purple arrows in the figure. Note that for clarity not all possible links are shown.

The framework divides the qualities of a TAAC into three areas: (1) input devices and techniques; (2) context sensing and AI; and (3) interaction parameters. Some of these qualities are easier to change than others. For example, to change input device from a joystick to an eye-tracker necessitates either a very versatile TAAC hardware platform or an explicit change of hardware device. In contrast, changing the number of word suggestions shown in a word prediction display normally only involves simply changing a software parameter and ensuring there is space in the UI for an additional word suggestion.

The framework also separates out AAC research and development areas. These are tentative, as a full scoping of this space necessitates a systematic literature review and interviews with AAC researchers. For now, we identify three key areas: (1) accessibility; (2) personalization; and (3) vocabulary and communication. We then break down each of these areas into subareas, covering research questions such as the expressiveness of the keyboard (under *Accessibility* → *Language and vocabulary interface*), and support for small talk (under *Vocabulary and communication* → *Communication*).

A way to use the framework is to use it as a map for better understanding the design space of TAAC system qualities that can be explored for a particular area, or subarea, of interest. For example, tinkering with system utterances that reflect users' emotions can potentially benefit re-establishing the user's native language and further contribute to maintaining their social role, such as a family member. As another example, altering different language models and their underpinning parameters may improve language predictions in different communication settings.

The framework can also inform TAAC system design by enabling designers and researchers to work backwards from the research areas and subareas to ideate additional TAAC system qualities that can be incorporated into a TAAC system. In this way the framework effectively assists TAAC researchers in eliciting requirements for a TAAC system that incorporates features with a direct research need.

Finally, in the long run we anticipate the use of systematic design engineering methods to make inroads in AAC as a complementary methodology (Kristensson, et al. 2020). In this context, the TAAC framework can be used a map for AAC designers to understand which TAAC system qualities to investigate, for example, the number of generated sentence suggestions, to gain an understanding into appropriate areas of AAC research and development interest.

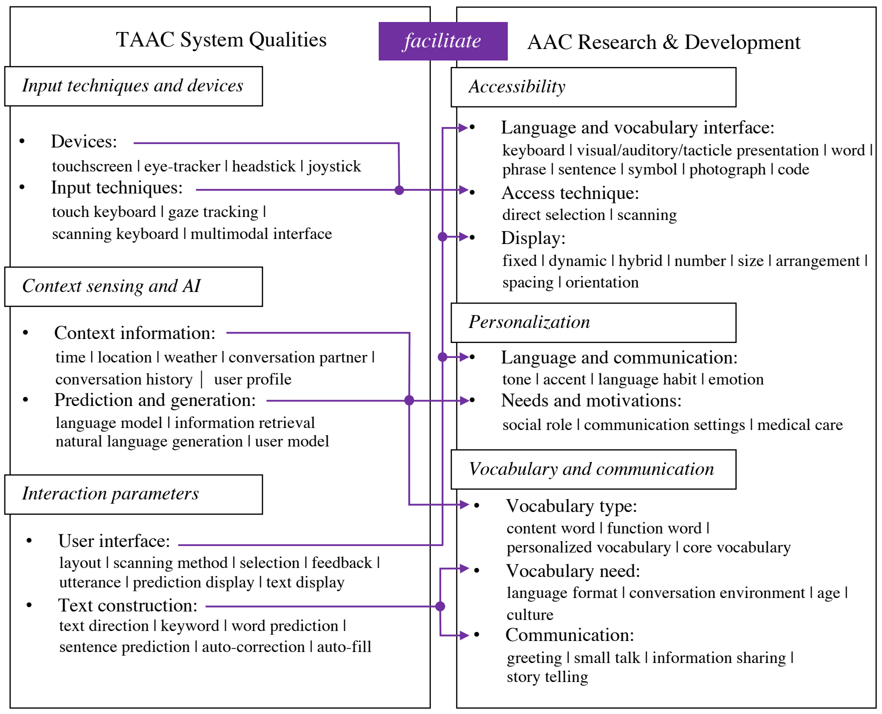
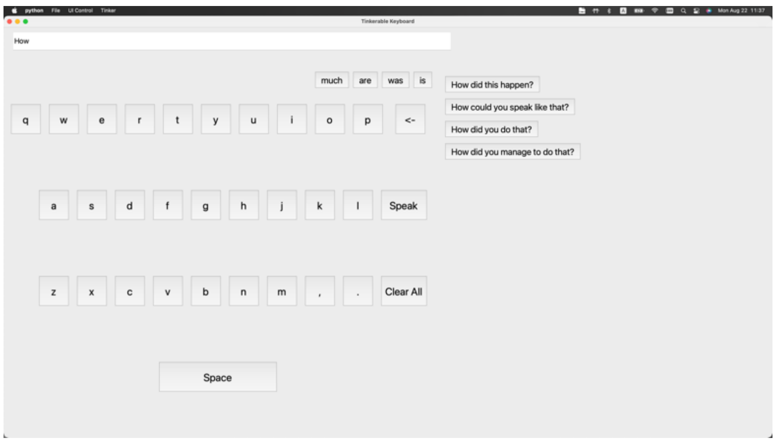

TAAC System Qualities
facilitate
AAC Research & Development
Input techniques and devices
Devices:
touchscreen | eye-tracker | headstick | joystick
Input techniques:
touch keyboard | gaze tracking |
scanning keyboard | multimodal interface
Context sensing and AI
Context information:
time | location | weather | conversation partner |
conversation history | user profile
Prediction and generation:
language model | information retrieval
natural language generation | user model
Interaction parameters
User interface:
layout | scanning method | selection | feedback |
utterance | prediction display | text display
Text construction:
text direction | keyword | word prediction |
sentence prediction | auto-correction | auto-fill
Accessibility
Language and vocabulary interface:
keyboard | visual/auditory/tacticle presentation | word |
phrase | sentence | symbol | photograph | code
Access technique:
direct selection | scanning
Display:
fixed | dynamic | hybrid | number | size | arrangement |
spacing | orientation
Personalization
Language and communication:
tone | accent | language habit | emotion
Needs and motivations:
social role | communication settings | medical care
Vocabulary and communication
Vocabulary type:
content word | function word |
personalized vocabulary | core vocabulary
Vocabulary need:
language format | conversation environment | age |
culture
Communication:
greeting | small talk | information sharing |
story telling

Fig. 1. A conceptual framework for explaining how qualities of a TAAC system (left) facilitate AAC research and development (right). The purple arrows indicate pathways linking TAAC system qualities that can be used for investigating related areas in AAC research and development. There are many pathways but we only show a few in the figure for clarity.

4 A Tinkerable AAC Text Prediction System

To demonstrate the viability of tinkerable AAC we have developed a tinkerable predictive AAC text entry system. This section introduces the design and features of this system and explains how it may assist AAC researchers and developers to iteratively improve the system. Fig. 2. shows the system, including its keyboard, word predictions, and sentence predictions.



How

How did this happen?

How could you speak like that?

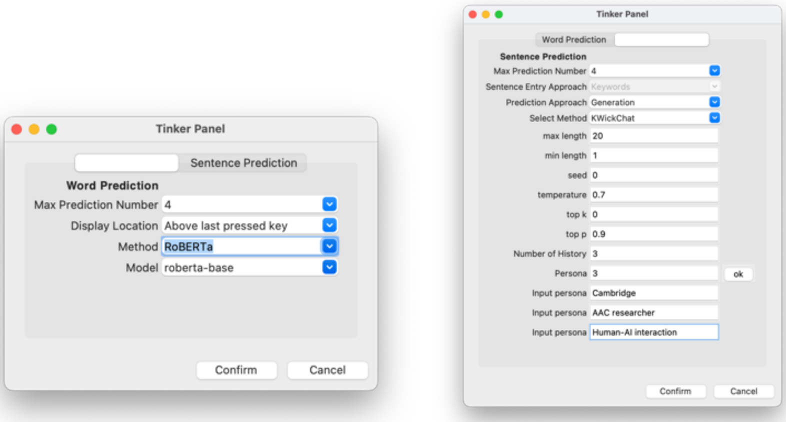
How did you do that?

How did you manage to do that?

Fig. 2. The main user interface of the tinkerable AAC system.

4.1 Interface

As shown in Fig.2., the menu bar is separated into three sections: *File*, *UI Control*, and *Tinker*. *File* allows the user to save the current settings and reload previous settings as readable text files that can thus be edited outside the system. *UI Control* allows the user to move keys on the keyboard around by dragging them and thus changing the layout and geometry of the keyboard. Similarly, the position of predicted words and sentences can also be moved around by dragging them. Further system tinkering features associated with the back end are integrated into the *Tinker Panel* (see Fig. 3.), which can be accessed by choosing *Tinker* in the menu.



Word Prediction

Max Prediction Number	4
Display Location	Above last pressed key
Method	RoBERTa
Model	roberta-base

Sentence Prediction

Max Prediction Number	4
Sentence Entry Approach	Keywords
Prediction Approach	Generation
Select Method	KWickChat
max length	20
min length	1
seed	0
temperature	0.7
top k	0
top p	0.9
Number of History	3
Persona	3
Input persona	Cambridge
Input persona	AAC researcher
Input persona	Human-AI interaction

Fig. 3. The tinker panel for word prediction (left) and sentence prediction (right).

The system allows configuring gaps between each key and supports two types of word prediction position mechanisms. The first provides word predictions at pre-fixed static locations while the second dynamically places word predictions in the vicinity of the last pressed key. Hence, gaps between each keyboard row provides space for dynamic word prediction displays, should they be of interest to the AAC user. Sentence predictions can be shown in static locations. The number of word and sentence predictions are adjustable from one to four.

By clicking the *Speak* button the system reads out any output text.

As previously alluded, the *Tinker Panel* provides the user with an opportunity to configure back end parameters. Currently it includes five means of word prediction and six approaches for sentence prediction using both AI-based generative sentence methods and conventional sentence retrieval algorithms. For each method, there are a list of relevant parameters that can be tinkered with. This feature is mainly intended for AAC researchers and designers so that they can alter parameters and assess different models without the need for them to be able to program or have any expertise in AI or information retrieval.

4.2 Text Prediction

Providing rich tinkerability around the central text prediction function has two main benefits. First, it allows customization for individual users. A wide range of language prediction methods allows the AAC user or specialist to choose the most suitable method for the individual use case. For example, some users require sending repeated commands (for example, "I need water" or "I am tired"). Such text retrieval can be achieved reliably using information retrieval methods, such as the included BM25 (Trotman, Puurula and Burgess 2014) and SBERT (Reimers and Gurevych 2019) approaches.

For other users, communication can potentially be greatly sped up by enabling users to provide keywords that are used to prompt a language model to generate entire sentences, for example GPT-2 (Radford, et al. 2019) and KWickChat (Shen, et al. 2022).

These different methods have different tradeoffs and are hence complementary. For example, information retrieval methods may be beneficial in routinary conversation with limited and repeated vocabulary, whereas sentence generation may be more efficient in open conversations, such as a causal chat where an exact utterance is not critical.

In addition, *how* to enable the AAC user to tell the system to generate or predict a sentence matters. Fundamentally, there are two approaches: either successive input of a series of words or the input of a series of keywords that serve as a prompt for a sentence generation algorithm. The two methods require a different interaction style, and one method may be preferred or more suitable for an individual user, or for a specific interaction context, such as a different environment or situation.

4.3 System Architecture

The tinkerable AAC system uses the model-view-controller (MVC) system architecture. MVC is one of the most widely used design patterns in software engineering of graphical user interfaces as it is simple and effective. It consists of three parts: (1) the model; (2) the view; and (3) the controller. In this system, the *model* corresponds to the back end, the algorithms and language models, the *view* is the user interface, including the keyboard (the front end), and the *controller* allows the user to interact with the front and back end. This modular design facilitates iterative development of the system. For example, if an AAC user is used to inputting letters via a touchscreen but suddenly requires using an eye-tracker, then an AAC developer merely needs to add an eye-tracker module to the view and link it to the controller, rather than developing a completely new system. Thereby the information retrieval and AI-based sentence generation subsystems, which require considerable development expertise, are reusable for a wide variety of different software development objectives.

We also conjecture that another advantage of this architecture might be that it simplifies integration of AAC systems that go beyond direct text entry. Many AAC researchers are interested in supporting conversations that are scenario-based, such as telling stories (Reiter, et al. 2009), giving instructions (Todman, et al. 2008), chat (Shen, et al. 2022), and so on. With a flexible system architecture, developers can more easily integrate state-of-the-art language models into the system as a new UI option for AAC researchers with minimal changes in the front end and thereby benefit from the set of existing subsystems and algorithms that are already provided by the system.

5 Discussion and Conclusions

This paper has introduced Tinkerable AAC (TAAC) and exemplified it with a tinkerable AAC system. While this system is a start towards a versatile enabling technology for user-involved AAC co-design, there are many improvements that are possible and many design considerations that need to be investigated.

First, from an end-user perspective any manipulation of front end aspects, such as the position of keys on the keyword and word- and sentence prediction slots, needs to be easy to understand and access. Further, any such manipulation needs to be easily reversible by the user.

Second, back end aspects, such as parameter choices and subsystem decisions need to be thoroughly explained. It will also be necessary to provide users, including designers, researchers, and developers, with easy methods to assess the efficacy of various parameter choices and activated subsystems. Ideally, a systematic approach is followed which could potentially be supported by the system by, for example, having workflows in the system for exploring and assessing parameter choices as a function of usage. This would enable designers and researchers to explore *what if* scenarios in a systematic way.

Third, an important objective of TAAC is to enable lightweight *in-situ* experiments that do not demand substantial development effort from AAC researchers. Thus, it will be critical to provide easy-to-use and understand interfaces for setting up experiments and for logging and analyzing data.

Fourth, it will be important to systematically evaluate any TAAC both in terms of benefits to end-users and in terms of the experience of designers, researchers, and developers.

In addition, the TAAC framework is a starting point for linking TAAC system qualities to AAC research and development areas and subareas. Further work is required, such as a systematic literature review and interviews with AAC researchers, to tease out additional research areas, and expand and refine existing research areas in the framework. With a more complete TAAC framework it is possible to work backwards from the research areas to the TAAC system qualities to identify gaps in support for research in a TAAC system.

In summary, we see TAAC as a promising direction in AAC but many challenges remain for TAAC to truly take the form of an enabling technology democratizing AAC design using state-of-the-art AI technologies. As examples of what is already possible, our existing system provides the scaffolding to assist with the following investigations: (1) assessing the impact of different text prediction layouts on text entry rate, cognitive load, and physical load; (2) understanding the impact of different language models on the AAC user experience; (3) analyzing how language behavior changes when AAC users are in different environments; and (4) assessing the effect of context-aware text entry on communication rate. We hope TAAC will serve as a platform for incorporating recent AI advances for the benefit of AAC research and thus act as a catalyst for advanced AAC design exploiting state-of-the-art AI to ultimately improve the communication rate and user experience with AAC systems.

6 Open Science

Complete source code for the Tinkerable Keyboard can be found here:
doi:10.17863/CAM.91650

References

Beukelman DR, Light JC (2020) Augmentative & Alternative Communication: Supporting Children and Adults with Complex Communication Needs. Baltimore: Brookes.

Ellis K, Dao E, Smith O, Lindsay S, Olivier P (2021) TapeBlocks: A Making Toolkit for People Living with Intellectual Disabilities. Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems. Yokohama, Japan: Association for Computing Machinery. 1-12.

Henderson A, Kyng M (1995) There's no place like home: Continuing design in use. In Readings in Human-Computer Interaction, by Morgan Kaufmann, 793-803. Elsevier.

Kristensson PO, Lilley J, Black R, Waller A (2020) A Design Engineering Approach for Quantitatively Exploring Context-Aware Sentence Retrieval for Nonspeaking Individuals with Motor Disabilities. Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems. Honolulu, HI, USA: Association for Computing Machinery. 1-11.

McKillop C (2018) Designing a Context Aware AAC Solution. Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility. 468--470.

Mirenda P (2014) Revisiting the Mosaic of Supports Required for Including People with Severe Intellectual or Developmental Disabilities in their Communities. Augmentative and Alternative Communication 19-27.

Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I (2019) Language models are unsupervised multitask learners. OpenAI blog 9.

Reimers N, Gurevych I (2019) Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084.

Reiter E, Turner R, Alm N, Black R, Dempster M, and Waller A (2009) Understanding the storytelling of older adults for AAC system design. Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009). 1-8.

Resnick M, Rosenbaum E (2013) Designing for tinkerability. In Design, make, play, 163-181. Routledge.

Shen J, Yang B, Dudley JJ, and Kristensson PO (2022) KWickChat: A Multi-Turn Dialogue System for AAC Using Context-Aware Sentence Generation by Bag-of-Keywords. 27th International Conference on Intelligent User Interfaces. Helsinki Finland: Association for Computing Machinery. 853-867.

Todman J, Alm N, Higginbotham J, and File P (2008) Whole utterance approaches in AAC. Augmentative and alternative communication 235-254.

Trotman A, Puurula A, and Burgess B (2014) Improvements to BM25 and Language Models Examined. Proceedings of the 2014 Australasian Document Computing Symposium. Melbourne, VIC, Australia. 58-65.

Vertanen K, and Kristensson PO (2011) The imagination of crowds: conversational AAC language modeling using crowdsourcing and large data sources. Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. 700-711.