FoMR
Frontiers of Multidisciplinary Research

Vol. 1 | No. 3 | Jul - Sep 2026

Portable Braille Keyboard (Braillex): A Wireless Tactile Input Device For Visually Impaired Users

Mitul Sudhirkumar Nagar[a], Snehalatha N[a], Arjun Mehta[a]*, Adarsh Jaiswal[a]

ABSTRACT

Smartphones, laptops, and tablets form the quiet, load-bearing infrastructure of modern life — the way people study, work, communicate, and navigate the world. A closer look at how these devices work, however, reveals a problem that mainstream technology con-versations often overlook: nearly all of them are designed, from the ground up, on the assump-tion that the user can see. Touchscreens, menus, on-screen keyboards, notifications, and other visual elements are not neutral design choices; they serve the needs of most people while creat-ing severe difficulty for the significant population of people who are blind or severely visually impaired.

Screen readers and voice input tools have partially addressed this need but have not provided a satisfactory solution for accurate, efficient text entry by visually impaired users in practice. For a student taking notes in a lecture, or an employee drafting an email in an open-plan office, these are not minor inconveniences — they are genuine barriers.

In this paper, we describe BrailleX, a small Braille-based wireless keyboard we designed and built as an experimental assistive input device. BrailleX has six physical buttons arranged in the standard Braille cell layout; combinations of these buttons produce characters through chording. Characters are transmitted over Bluetooth to a connected smartphone or computer, while an audio-only feedback system, with no visual indicator, communicates device state to the user.

Keywords: Assistive technology, Braille keyboard, Tactile input devices, Accessibility, Human–computer interaction

Received date: 23.04.2026
Accepted date: 02.07.2026

https://doi.org/10.65601/FoMR.2026.1.3.4

***Corresponding Author:**

[a]*Dept. of Computational Intelligence, SRM Institute of Science and Technology, Chennai, India*

Email: am6367@srmist.edu.in

INTRODUCTION

In the term digital divide, typically the gap between internet access users and non-internet users is implied. There is another digital divide that receives little attention, the gap between people who can and cannot comfortably use the interfaces through which digital services are delivered (World Health Organization, 2026). This second digital divide could be nearly as significant as the first for those who are blind or otherwise severely visually impaired.

The problem is not that technology has ignored visually impaired users entirely. Screen readers, which are decades old, have improved considerably over time. NVDA, JAWS, and VoiceOver, for example, allow blind users to navigate software interfaces that would otherwise be com-pletely inaccessible.

However, for text input specifically, practical solutions for visually impaired users remain unsat-isfying (Kane et al., 2011). Voice dictation requires a relatively silent environment, is not always accurate with technical vocabulary and proper nouns, and compromises the privacy of what is spoken. Conventional on-screen keyboards, with or without screen-reader support, remain a slow and error-prone way of entering text without sight (Azenkot et al., 2012). For Braille users specifically, there is a persistent gap between the skill they already possess and the interfaces they are forced to operate.

Braille itself is worth considering carefully in this respect (National Federation of the Blind, 2009). The Braille writing system was developed in the nineteenth century and remains one of the best tools available for teaching literacy to visually impaired people. Braille literacy is linked to higher educational and employment success rates and improved independence (Azenkot et al., 2012). Its six-dot cell is elegant in its simplicity: sixty-four combinations are enough to cover the entire alphabet, numerals, punctuation, and special characters, and many visually impaired people have already learned it.

It is this insight that led to BrailleX. For a user who can already read and write Braille, a device that accepts Braille input should feel natural from the first session. Rather than design yet another alternative interface and ask users to learn it from scratch, we set out to build a device that met users where they already were.

The rest of this paper describes how we did it, what we observed when testing the prototype, and where this work should go next.

Literature Survey

A critical reading of the assistive technology literature reveals a recurring pattern: researchers often achieve impressive laboratory results but fall short of turning those results into something people can afford and use every day — the kind of device that makes its way into people's homes. The gap between what is demonstrated and what is practical is substantial, and under-standing why is valuable.

The most technically advanced assistive devices for readers who are visually impaired are ar-guably the refreshable Braille displays, which use arrays of mechanical pins that are raised and lowered electronically and can thus display a line of digital text as a row of tactile Braille cells. Southern et al. (2012) describe notable engineering breakthroughs in the reliability and durability of these pin mechanisms, and the resulting devices do function effec-tively. The downside is the price: a good refreshable Braille display can cost several thousand dollars. High-precision machining for reliable pin arrays is expensive, and the market is small enough that economies of scale offer little relief. As a result, access is effectively limited to institutions, research laboratories, and Braille users with access to assistive-technology funding programs (Frey et al., 2011).

Braille input has taken a different path: the hardware chording keyboard, in which six keys correspond to the six Braille dots rather than to individual letters as in standard typing. Pressing combinations of these keys simultaneously to produce characters is a technique that has existed for some time (Southern et al., 2012; Kane et al., 2011).

Azenkot et al. (2012) extended this work by examining finger detection for non-visual touchscreen text entry, along with key spacing, force, and travel distance, and used their findings to establish design rules that we drew on for BrailleX. Kane et al. (2011) compared input approaches in the context of mobile devices, an important step given how many people now rely on smartphones as their primary computing device, and found that input modal-ity matters for both speed and user preference.

The rise of capacitive touchscreens opened a further design space. AInfiai and Sampalli (2017) created BrailleEnter, which allows Braille code to be entered on a standard capacitive touchscreen phone using multi-touch gestures. Its main advantage is that it requires no extra hardware, relying only on software running on a device the user likely already owns, and it achieved higher typing speeds than an on-screen QWERTY keyboard. That advantage should not be overstated, however. Anyone who has tried to use a touchscreen without being able to see it will recognize the fundamental shortcoming of this approach: there are no tactile boundaries. With no key edges to anchor a finger against, users need substantial practice, and still make frequent errors, to know where they are on the screen. Haptic and audio feedback can compensate for some of this (Frey et al., 2011), but in our view it is not a complete solution.

More recent work has explored several novel directions. Liu et al. (2023) proposed a self-powered Braille typing system that harvests kinetic energy from the typing motion it-self, a clever approach to the battery problem that has not received as much attention as it deserves. Begmatov et al. (2023) designed a folding Braille keyboard for im-proved portability. Ramos-García et al. (2022) proposed an IoT-connected Braille device with cloud processing for additional functionality (Mascetti et al., 2012). Each of these represents an incremental improvement along some dimension, though each also carries trade-offs that the respective authors acknowledge.

Broader reports on the state of visual impairment, such as the World Health Organization's fact sheet (World Health Organization, 2026), and usability studies such as Kane et al. (2011), also point to two persistent weaknesses in the landscape — cost and portability. Frey et al. (2011) and Azenkot et al. (2012) both advocate for multimodal feedback that incorporates tactile, auditory, and other non-visual signals to make assistive de-vices more resilient in general use. Oliveira et al. (2011) enumerate a spectrum of HCI challenges facing blind users, offering a conceptual model that identifies where new designs might be most helpful. Together, these sources provided a guideline that we used to form a design strategy for BrailleX.

System Architecture

The design of BrailleX is deliberately simple, and we consider this a point of strength (Begmatov et al., 2023). The system has two major pieces — the embedded hardware the user holds and interacts with, and host-side software running on whatever device is being typed on — connected by a Bluetooth link, as illustrated in Figure 1. Each end does only what it is designed for and is not expected to duplicate work the other end handles better.

The hardware consists of five subcomponents: a six-button Braille input interface, a microcon-troller that runs the device firmware, a Bluetooth module for wireless connectivity, a power management circuit that regulates battery power, and an audio feedback module that conveys device state to the user (Mascetti et al., 2012). These subcomponents are packaged into a single compact enclosure carried in the hand while typing.

The firmware follows an event-driven design, a deliberate choice rather than a default. In this architecture, the system sits idle until an event occurs — a button press, a timer firing, a battery voltage reading crossing a threshold — at which point it handles that event and returns to wait-ing. This is more power-efficient than continuous polling and also makes the firmware logic easier to reason about during development and debugging (Mascetti et al., 2012).

On the host side, BrailleX presents itself as a Bluetooth HID keyboard. This was an intentional and consequential design decision: because the device conforms to the standard HID keyboard protocol, it works on any operating system that implements the keyboard HID profile. Blue-tooth keyboards generally require no drivers or additional software (Begmatov et al., 2023). Characters typed on BrailleX appear on the host exactly like input from any other keyboard, which means the device works in any application, text field, or command prompt that accepts keyboard input.

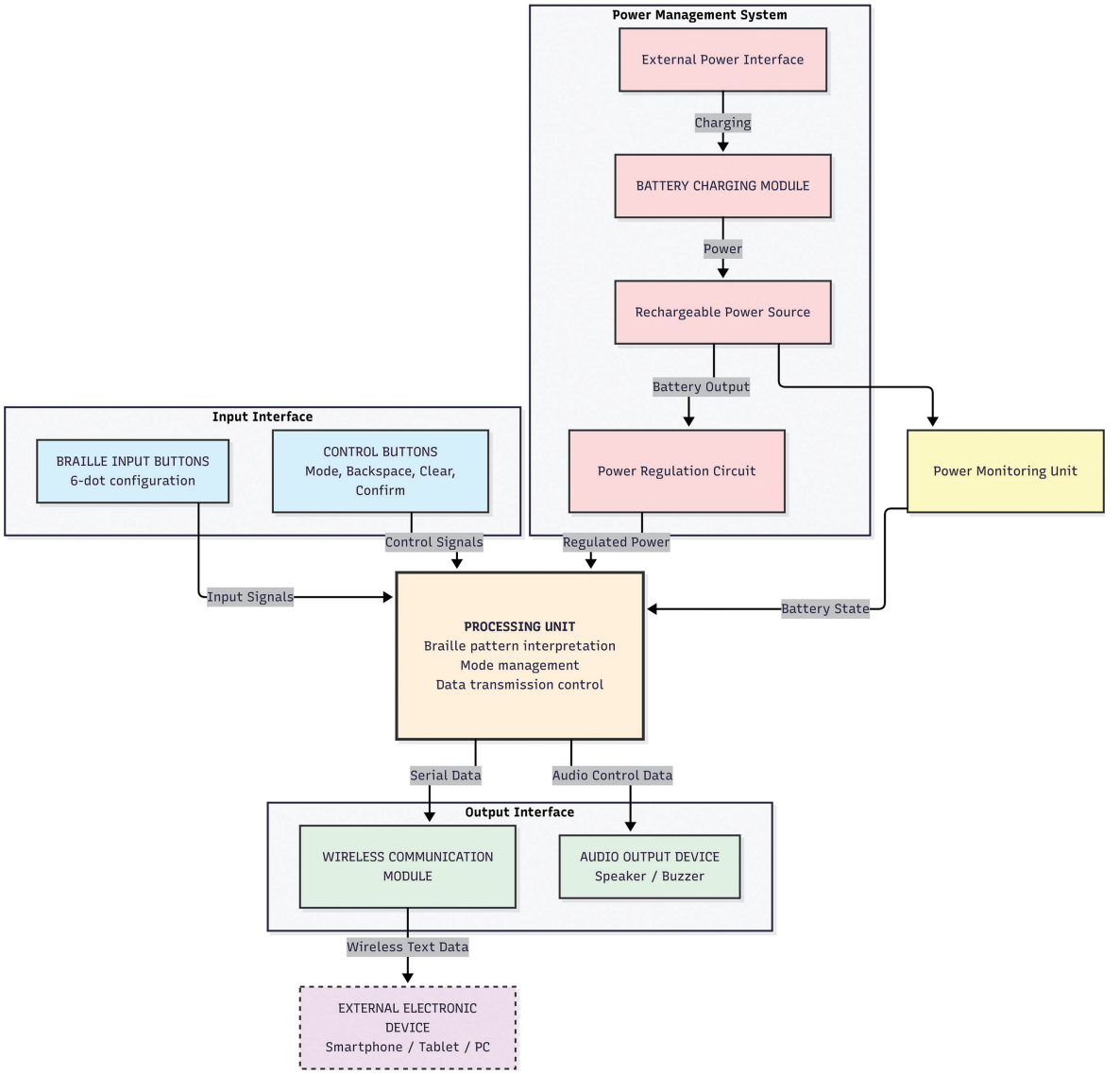

Figure 1 BrailleX system architecture overview

Braille Input Mechanism

The six buttons making up the Braille input interface are the device's primary point of contact, so getting their placement and the feel of their click right was one of the more significant design decisions. The layout had to remain a two-column, three-row standard Braille cell, with dots 1, 2, and 3 on the left and dots 4, 5, and 6 on the right; any other layout would have forced users to remap a spatially learned pattern, which seemed an unnecessary and regressive exercise (National Federation of the Blind, 2009).

Characters are produced by pressing combinations of these buttons simultaneously (Braille chording). The letter A is dot 1; B is dots 1 and 2; C is dots 1 and 4; D is dots 1, 4, and 5; and so on through the full Braille character set. The 63 non-empty combinations of the six binary buttons are enough to represent the alphabet, digits, standard punctuation, and the most common formatting and control characters. For Braille users, this is not a new skill but the same skill applied through a new device (Kane et al., 2011).

The timing aspect is one that anyone who has seriously considered building a chording keyboard will recognize immediately (Azenkot et al., 2012). When a user presses two buttons simultane-ously, the fingers rarely land at exactly the same instant; one finger will always arrive slightly before the other. If the firmware treats each button press as a separate event the moment it oc-curs, it will frequently misinterpret a two-button chord as two separate single-button inputs. A chord detection window addresses this: the firmware starts a short timer when the first button in a group is pressed, then collects any further button presses occurring within that window before determining which character was intended. The length of this window proved to matter a great deal in testing, discussed later. Figure 2 shows the standard Braille cell dot numbering convention used as the basis for the button layout.

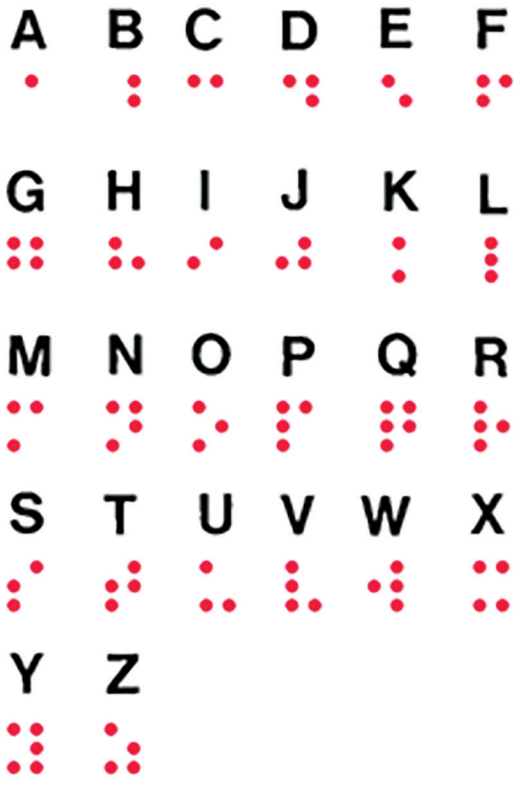

Figure 2 Standard Braille cell layout with dot numbering

Figure 3 BrailleX prototype showing rear ports and auxiliary function buttons

The device has only a few additional dedicated function buttons for operations outside the char-acter vocabulary of Braille: deleting the previous character, toggling input modes (such as letters versus numbers), and pushing the current buffer to the host, as shown in Figure 3. These con-trols are arranged around the periphery of the button cluster and have a distinct shape so they can be distinguished from the Braille buttons by touch alone. Figure 4 shows the six-button interface in its standard two-column, three-row configuration.

Figure 4 Six-button Braille input interface in standard two-column, three-row cell layout

Embedded Processing Unit

The microcontroller handles all processing inside the device. It reads button states, executes the chord detection logic, looks up the character associated with each completed chord, manages the text buffer, drives the Bluetooth module over a serial link, samples the battery voltage at fixed intervals, and triggers audio events when relevant conditions occur (Mascetti et al., 2012). The firmware must do all of this reliably, quickly, and within a power budget that supports reasonable battery life.

Button scanning runs on a short loop that repeatedly checks the state of the inputs at several kHz. Once the chord detection window has closed and the set of currently pressed buttons is finalized, the firmware performs a table lookup — indexing into an array using the button combination as the index — to retrieve the character code, which is fast enough to be effectively instantaneous. The resulting PCB design is shown in Figure 5.

The text buffer is a small ring buffer holding input characters that are either queued for trans-port or, more typically, have already been sent but remain eligible for deletion. Pressing the delete function button removes the most recent character from the buffer and sends a backspace keystroke to the host, which causes the host application to delete that character. The buffer is intentionally small, since the device is designed primarily for real-time typing rather than offline composition (Begmatov et al., 2023).

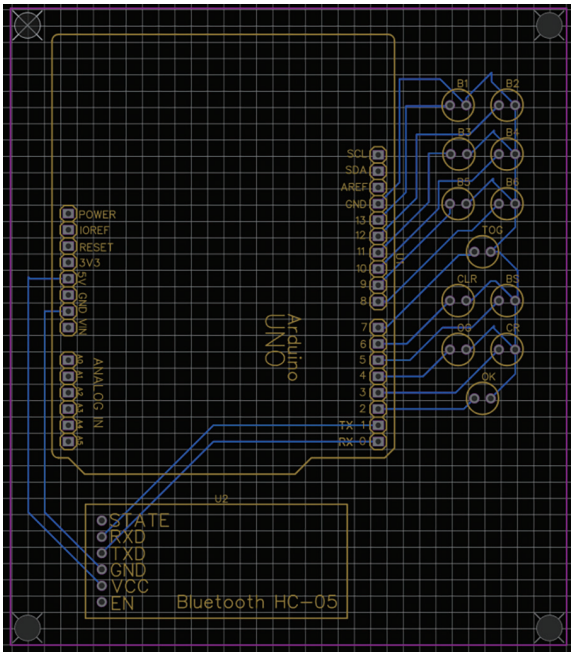

Figure 5 PCB Design of BrailleX

Battery monitoring is performed by sampling the voltage at the battery terminals through an analog-to-digital converter input on the microcontroller (Liu et al., 2023). State of charge is estimated from the sampled voltage using a battery discharge curve, with a threshold below which the battery is considered exhausted. In the current implementation, this threshold is set at 20%, at which point a low-battery flag is raised and the audio warning plays. This is not a precision fuel gauge, but it is accurate enough to give the user a useful estimate of the time remaining before the device shuts down.

Wireless Communication

Wireless connectivity uses a Bluetooth module connected to the microcontroller via a UART serial interface (Ramos-García et al., 2022). The module advertises itself to Bluetooth devices as a Bluetooth HID keyboard, so pairing with a host works the same way as for a commercial keyboard: put the device into pairing mode, locate it in the host's list of Bluetooth devices, confirm the pairing, and it is connected. We wanted an interaction that was familiar and painless, and the standard Bluetooth keyboard pairing flow already provides that, so we adopted it directly via the HID profile (Begmatov et al., 2023).

In operation, the microcontroller writes character codes to the Bluetooth module over the serial link, and the module formats them as HID keyboard reports and transmits them to the host. The host OS receives these reports, passes them through its normal keyboard input stack, and delivers them to whichever application currently has focus. From the application's perspective, BrailleX is simply a keyboard.

We tested the device with several host devices running different operating systems. Pairing succeeded in every case, and characters appeared on the host without any discernible delay between the button press and the character showing up on screen. The connection did not drop, and we did not observe any noticeable retransmission events over the course of long typing sessions (Kane et al., 2011).

Power Management System

Power comes from a small rechargeable lithium-ion battery. Lithium-ion was chosen for its high energy density — a large amount of capacity in a small volume and weight — and its reliability over many charge-discharge cycles (Liu et al., 2023). A rechargeable battery is non-negotiable for a device meant for daily use, since users are unlikely to want the impracticality and recurring cost of replacing disposable batteries.

A power management integrated circuit sits between the battery and the rest of the electronics (Mascetti et al., 2012). Its main function is voltage regulation: because a lithium-ion cell's terminal voltage ranges from 4.2 V fully charged to 3.0 V depleted, the regulator must supply a constant output voltage across that entire range. The same circuit also controls USB charging, limiting the charge current and capping the cell voltage to prevent overcharging.

The low-battery detection described above gives users time to finish their current task before needing to recharge. During testing, we found that users appreciated this warning; a sudden shutdown in the middle of typing, with no visual indicator of remaining charge (Azenkot et al., 2012), was disorienting for someone who cannot see a battery icon on screen.

Audio Feedback System

Designing the audio feedback system required considering what information is useful to the user, and how to present it in a way that is neither too quiet to notice nor loud and annoying. Visual indicators — LEDs, screens, notification badges — are the conventional way devices communicate their state, but are entirely inappropriate here. Whatever the user needs to know about the device must be delivered acoustically or tactilely (Frey et al., 2011).

We chose a small buzzer attached directly to a microcontroller output pin, representing several types of events with distinct sound patterns (Alnfiai & Sampalli, 2017): a short, upward-pitched double beep for a correctly typed character; two long ascending tones for a successful Bluetooth connection; a slowly repeated low tone, made deliberately more prominent than the other sounds because of its time-critical nature, for low battery; and an erratic, stuttering pattern for an error condition. All sounds are short enough not to disrupt the rhythm of typing during normal use, yet distinct enough to be told apart after a short period of familiarization.

We also spent time during testing adjusting the volume and tonal qualities of the feedback sounds based on tester feedback. Some testers initially found the prototype's sounds too similar to one another; after adjusting the tone patterns, this problem diminished for that group of users (Oliveira et al., 2011).

One concern we did not anticipate was that nearly all testers wanted inserted characters read aloud through the speaker — a text-to-speech mode that would announce each character or word as it was typed (Azenkot et al., 2012). The current prototype does not have this capability, but it is planned for the next version.

Device Design And Ergonomics

Physical design matters more for a device like this than it might initially appear (Azenkot et al., 2012). Because the user cannot see the device, the entire experience depends on touch and sound. A button that requires too much force, is difficult to locate with the fingers, or is too similar to an adjacent button will produce more errors and frustration (Frey et al., 2011). Here, ergonomics is as much a functional requirement as a comfort consideration. Figure 6 and Figure 7 show the front and top views of the BrailleX 3D model, respectively.

The enclosure was designed to sit comfortably in two hands, with thumbs or index fingers resting on the Braille button cluster (Begmatov et al., 2023). Buttons are spaced identically to a standard Braille cell, since users develop spatial memory for Braille through years of reading printed documents or using mechanical Braillewriters (National Federation of the Blind, 2009). Changing this spacing would force

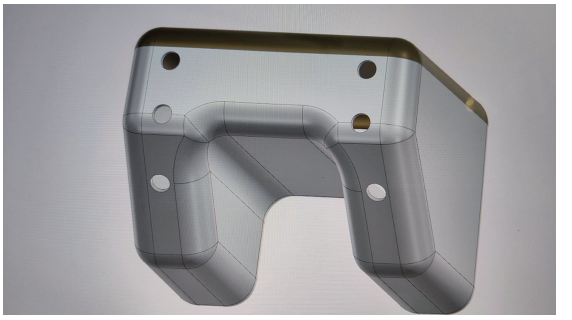

Figure 6 3D model (Front View) BrailleX device

users to consciously reorient their fingers, costing time and causing errors.

We selected buttons that provide clear tactile feedback (Azenkot et al., 2012). Each press produces a distinct mechanical click that confirms to the user that the input was registered, which matters because users cannot see the device to check whether a press was held long enough to register; the click is their only confirmation. The function buttons at the edges of the cluster also have a slightly different cap shape, giving a tactile cue that distinguishes them from the Braille input buttons without requiring the user to count positions (Kane et al., 2011).

The total device is small enough to be carried in a coat pocket and light enough that holding for a long period does not fatigue the hand (Begmatov et al., 2023). A finger along one edge of the enclosure can easily locate the charging port position on that edge and the port opening is wide enough that it can be located without precise visual cues.

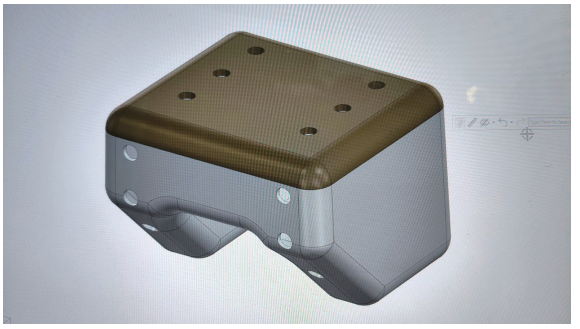

Figure 7 3D model (Top View) BrailleX device

RESULTS AND TESTING

We evaluated the prototype along four performance dimensions: input accuracy, Bluetooth com-munication reliability, audio feedback usability, and general usability observations (Kane et al., 2011). At this stage, testing was designed to demonstrate that the core design works adequately and to identify the areas of greatest remaining improvement.

Preliminary Internal Evaluation

At this stage of development, testing was conducted internally by 2 members of the project team rather than by independent participants recruited for a formal user study. Testing sessions were split between two conditions: some sessions were conducted with the tester's eyes closed (blindfolded), to approximate the non-visual condition BrailleX is designed for, and others were conducted with normal sight, to allow the tester to simultaneously observe and log device be-haviour during debugging. Testers were not certified Braille-literate; team members learned the Braille cell layout and basic chording specifically for this project rather than bringing pre-existing fluency.

We report the results of this internal evaluation below as a preliminary engineering validation of the hardware and firmware, not as a substitute for a formal usability study with visually impaired, Braille-literate users. That larger, independent study is identified as an immediate next step in the Conclusion, and is a precondition for any claims about real-world usability or educational/clinical adoption.

Input Accuracy

Input accuracy was tested by repeatedly entering a defined character set consisting of all 26 letters of the alphabet, the digits 0 through 9, and a set of common punctuation characters, across several sessions (Kane et al., 2011). For each character entered, we logged whether the firmware produced the expected output, an incorrect character, or no character at all.

Overall accuracy was 93% across 100 character entries logged during internal sessions. Errors were not randomly distributed: 7% of all errors occurred on multi-button chords, and of those, the great majority involved one finger landing outside the chord detection window — usually the last finger to arrive, after the window had already begun closing. Extending the detection window from 20 ms to 50 ms reduced this class of error, but introduced a second problem: fast typists occasionally had the start of their next chord merged with the tail of the previous one. This suggests an optimal window length that may vary across users and typing speeds, which is a question we can only answer with confidence once we test a larger and more representative participant pool, as noted above. Given the small internal sample, we report these figures as raw counts in the text rather than as a per-character distribution plot, since a graph would imply a level of statistical reliability that 2 testers cannot support.

Communication Reliability

We paired BrailleX with two laptop and two smartphone hosts running different operating systems. Pairing succeeded in all four combinations without any special configuration beyond the standard Bluetooth pairing flow.

We observed no transmission errors or connection drops across extended typing sessions (Ramos-García et al., 2022). Characters appeared on the host with low enough latency that testers did not report any perceptible delay. Our fastest tester, typing at the highest rate achieved during internal sessions, did not report any transmission throughput bottleneck, suggesting that Blue-tooth data rate is not a limiting factor at the typing speeds observed in this evaluation (Begmatov et al., 2023).

Audio Feedback Testing

We tested the audio feedback system by presenting each defined event type and asking testers to identify which event had occurred (Azenkot et al., 2012). Testers were given a brief familiar-ization period (one repetition of each sound after the sound set was introduced) and were then asked to identify sounds presented in random order.

After this short familiarization, identification accuracy was 90% averaged across all event types among internal testers. The low-battery warning was recognized most reliably, consistent with its design goal of conveying time-critical information. In an initial round of testing, testers occasionally confused the connection-established tone with the character-confirmation tone; after we revised the tone patterns, this confusion diminished in a second round (Oliveira et al., 2011). As with input accuracy, we report these results as raw counts/percentages from a small internal sample rather than as a plotted distribution, given how few data points are available at this stage.

Usability Observations

The most informative sessions were the informal ones conducted within the team. Each tester was given the device to explore for a few minutes, then asked to type a short passage while we observed where they hesitated and how they corrected mistakes, followed by a brief discussion afterward.

Even without prior Braille fluency, testers were able to locate the button layout within the first 5 minutes of practice, and were producing legible text within 30 minutes, which we take as an early, encouraging sign for the chord layout's learnability rather than a claim about performance for fluent Braille users, whose learning curve and error patterns may differ substantially.

The most consistent request across the internal testers was for some form of text-to-speech feedback — specifically, having each typed character or word voiced aloud through the device's speaker (Azenkot et al., 2012). Several testers noted that, without this, they had to rely on their host device's screen reader to hear their typed output, forcing them to divide attention between two separate audio streams. Adding speech synthesis directly to BrailleX would let it operate more independently of the host device's accessibility features (Alnfiai & Sampalli, 2017).

Comparison With Existing Braille Input Devices

Table 1 summarises how BrailleX sits alongside three well known commercial Braille input and notetaking devices.

Looking at the current assistive technology market, we observed a substantial price gap. Most high-end devices are "notetakers" that attempt to integrate a full suite of screens, applications, and keyboards into one unit, often at a price tag near $5,000, putting them out of reach for many users (Southern et al., 2012). More mobile alternatives, such as the Mantis Q40 or Focus 14 Blue, still rely on expensive electromechanical pins for their refreshable Braille displays (Frey et al., 2011).

BrailleX takes a different approach: our goal was to remove the expensive display hardware entirely and provide a lightweight, high-speed wireless input "bridge" (Begmatov et al., 2023), delegating screen-reading and processing to a smartphone the user already owns. This simpli-fies the underlying hardware substantially, making the device considerably cheaper and more pocketable than a large, fragile commercial display (Begmatov et al., 2023).

Manufacturing Cost and Scalability

A preliminary bill-of-materials estimate for the current prototype – covering the microcon-troller, Bluetooth module, tactile switches, battery, power management IC, buzzer, and enclo-sure – puts the per-unit component cost at approximately Rs. 4,000 (roughly US$48) at low-volume, hobbyist-scale

Table 1 Comparative Analysis of Braille Keyboards

Device	Type	Main Conn.	Key Advantages	Cost Tier	Primary Drawback
BrailleSense 6	Notetaker	Wi-Fi / BT	Full Android suite, 32-cell display	Very High	Bulky and extremely expensive
Mantis Q40	Hybrid	USB / BT	40-cell display with QWERTY keys	High	Large foot-print, high cost
Orbit Writer	Portable Input	USB / BT	Perkins-style keys, navigation Mode	Moderate	Limited standalone logic
BrailleX (Proposed)	**Tactile Input**	**Bluetooth**	**Event-driven, audio feedback, pocket-sized**	**Low**	**Current prototype stage**

sourcing. This is substantially below the cost of any refreshable-display device, since BrailleX avoids the electromechanical pin arrays that dominate the bill of materi-als in those products. At moderate production volumes (on the order of hundreds to low thou-sands of units), component costs would be expected to fall further due to bulk purchasing, and injection-molded enclosures would replace the current 3D-printed housing, reducing per-unit assembly cost and improving durability. Because the design avoids custom precision mechanical parts — the primary cost driver in refreshable Braille displays — BrailleX is inherently more amenable to scaling through standard PCB assembly (PCBA) services than pin-based de-vices are. We see this as one of the strongest practical arguments for the input-only approach: cost reduction here is a matter of conventional electronics manufacturing scale-up rather than solving a hard mechanical engineering problem.

DISCUSSION

The comparison makes one thing clear: existing manufacturers assume a user is only fully served if a display is integrated into the device, which drives cost into the thousands of dollars. By addressing only the input side of the problem, BrailleX avoids this cost driver entirely. By offloading text-to-speech processing and computing to a user's existing smartphone or laptop, we retain a high level of usability while substantially lowering hardware cost.

At the same time, its compact, event-driven architecture gives BrailleX a level of portability that "all-in-one" devices cannot match. It is designed for the modern user who already owns a capable mobile device but still needs a silent, tactile way to interface with it (Alnfiai & Sampalli, 2017).

CONCLUSION

BrailleX grew out of a basic belief that building assistive technology on top of skills people already have is more respectful and practical than requiring them to learn new skills (National Federation of the Blind, 2009). Braille literacy is not an obscure skill — it is a prerequisite for reading and writing for many blind people, and a device that accepts Braille input should feel natural to anyone who already knows Braille (Azenkot et al., 2012). That was our point of departure, and the design followed from it. The physical button layout, chord detection approach, Bluetooth HID profile, and audio-only feedback system (Alnfiai & Sampalli, 2017) were all focused on realizing this starting point as cleanly as possible.

This prototype demonstrates that the approach works at an engineering level. Input accuracy was strong for our internal testers, Bluetooth communication was consistently reliable and low-latency across all tested host platforms, and even testers without prior Braille fluency could use the device productively after a short orientation period (Kane et al., 2011). Device state was communicated effectively through the audio feedback system, and the physical design was comfortable to hold and easy to use by touch (Azenkot et al., 2012).

What still needs improvement is well understood. A central limitation of the present work is that testing

to date has been an internal, preliminary evaluation carried out by 2 members of the project team under blindfolded and sighted conditions, rather than an independent study with visually impaired, Braille-literate participants. The encouraging results reported above should therefore be read as evidence that the hardware and firmware function as intended, not as a val-idated measure of usability, accuracy, or learnability for the device's intended user population, whose prior Braille fluency, tactile sensitivity, and typing habits may differ substantially from our internal testers'. Addressing this is our highest priority next step: we plan to conduct a for-mal study with a larger and more diverse cohort of blind and visually impaired, Braille-literate participants, recruited through channels such as schools for the visually impaired, disability support offices, or assistive-technology user groups, with appropriate informed consent and ethics review. That study should span a wider range of ages and Braille proficiency levels, and evaluate the device over longer, real-world use periods (weeks rather than single sessions) in varied settings such as classrooms, workplaces, and independent daily use, rather than only in a controlled internal setting.

Beyond this core limitation, calibration of the chord detection window should be revisited once a larger and more representative participant pool is available (Kane et al., 2011). Text-to-speech feedback, the feature most consistently requested during internal testing, will be a priority for the next hardware/ firmware iteration (Azenkot et al., 2012). Predictive text or word completion could also improve typing speed for less fluent Braille-chording users (World Health Organization, 2026). Together, the planned user study and these firmware refinements would provide the kind of evidence needed to support adoption of BrailleX in educational or clinical settings (Kane et al., 2011).

None of this is a small undertaking, but none of it is out of reach either. Most notably, this proto-type demonstrates that the underlying architecture is sound: a compact, affordable, Bluetooth-based Braille input device that requires no special host software is a viable thing to build (Begmatov et al., 2023). What remains is the work of turning a viable prototype into a deployable, rigorously validated tool.

Author Contributions

Mitul Sudhirkumar Nagar: Project supervision, research guidance, and manuscript review.

Snehalatha N.: Project supervision, research guidance, and manuscript review.

Arjun Mehta: Conceptualization, hardware design, firmware development, prototype fabrication, PCB design, testing, and lead writing.

Adarsh Jaiswal: System architecture design, Bluetooth communica-tion module development, firmware development, results analysis, manuscript preparation, and lead writing.

Funding: This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Declarations

Conflict of interest: The authors declare no conflicts of interest.

REFERENCES

Alnfiai, M., & Sampalli, S. (2017). BrailleEnter: a touch screen braille text entry method for the blind. *Procedia Computer Science*, *109*, 257-264.

Azenkot, S., Wobbrock, J. O., Prasain, S., & Ladner, R. E. (2012). Input finger detection for nonvisual touch screen text entry in Perkinput. In *Proceedings of graphics interface 2012* (pp. 121-129).

Begmatov, S., Arabboev, M., Vakhkhobov, S., Rikhsivoev, M., Gaziev, K., & Nosirov, K. (2023). Design and development of a folding type braille keyboard. *Central Asian Journal of Theoretical and Applied Science*, *4*(9), 1-9.

Frey, B., Southern, C., & Romero, M. (2011, July). Brailletouch: mobile texting for the visually impaired. In *International Conference on Universal Access in Human-Computer Interaction* (pp. 19-25). Berlin, Heidelberg: Springer Berlin Heidelberg

Kane, S. K., Wobbrock, J. O., & Ladner, R. E. (2011, May). Usable gestures for blind people: understanding preference and performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 413-422).

Liu, W., Yu, W., Li, K., Zhou, S., Wang, Q., & Yu, H. (2023). Enhancing blind-dumb assistance through a self-powered tactile sensor-based Braille typing system. *Nano Energy*, *116*, 108795.

Mascetti, S., Bernareggi, C., & Belotti, M. (2012, July). TypeInBraille: quick eyes-free typing on smartphones. In *International Conference on Computers for Handicapped Persons* (pp. 615-622). Berlin, Heidelberg: Springer Berlin Heidelberg.

National Federation of the Blind. (2009). *The Braille literacy crisis in America*. https://nfb.org/images/nfb/documents/pdf/braille_literacy_report_web.pdf

Oliveira, J., Guerreiro, T., Nicolau, H., Jorge, J., & Gonçalves, D. (2011, October). Blind people and mobile touch-based text-entry: acknowledging the need for different flavors. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility* (pp. 179-186).

Ramos-García, O. I., Vuelvas-Alvarado, A. A., Osorio-Pérez, N. A., Ruiz-Torres, M. Á., Estrada-González, F., Gaytan-Lugo, L. S., ... & Santana-Mancilla, P. C. (2022). An iot braille display towards assisting visually impaired students in mexico. *Engineering Proceedings*, *27*(1), 11.

Southern, C., Clawson, J., Frey, B., Abowd, G., & Romero, M. (2012, September). An evaluation of BrailleTouch: mobile touchscreen text entry for the visually impaired. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services* (pp. 317-326).

World Health Organization. (2026). *Blindness and vision impairment* [Fact sheet]. https:// www.who.int/news-room/fact-sheets/detail/ blindness-and-visual-impairment