Principles for Gaze-Accessible Web Development

Last updated: April 6, 2026

Preamble

While building this ALS palliative care resource website, we found that although general web accessibility guidelines exist, there is relatively little web design guidance that specifically addresses the needs of people with ALS and other neuromuscular conditions who may rely on eye-tracking (eye gaze) communication methods.

The purpose of this document is to share a set of technical design principles (our rationale) we used to build an accessible website for eye-tracking. These principles reflect practical considerations related to eye-gaze interaction, keyboard emulation, focus management, and cognitive and physical fatigue.

Disclaimer

> This document is a work in progress. The principles presented here are based on our own research, general WCAG guidelines, and cross-checked with generous feedback from the web team at Bridging Voice, a nonprofit specializing in accessible communication solutions for individuals with ALS. As this work continues, we hope to refine these principles with stronger evidence from systematic studies.

1 How Eye-Gaze Technology Works

For users with limited mobility, web navigation can be performed using eye-gaze technology, which relies on specialized hardware to track eye movements and translate them into on-screen actions. Instead of using a mouse or physical keyboard, users control the interface using where they look and how long they look at it. Common eye-gaze interaction patterns include:

- **Direct (mouse emulation):**
  - The cursor moves to wherever the user looks on the screen.
  - Holding gaze steady for a short period ("dwell time") triggers a click.
- **Indirect (keyboard-style navigation):**
  - The interface highlights one interactive element at a time (similar to pressing the `Tab` key).
  - The user looks at on-screen controls such as "Next," "Previous," or "Select."
  - Looking at these controls moves focus or activates the selected item.

2 Our Six Gaze-Accessible Web Design Principles

2.1 Large and Defined Buttons for Clickable Elements

- **Large, clearly defined targets.** Where possible, major clickable elements should be embedded in large buttons rather than inline text links, creating target areas that are more tolerant of gaze drift and visually distinct from static content.

- **Button size.** WCAG 2.1 recommends target areas of at least **44 $\times$ 44 CSS pixels**[1]. We treated this as a minimum and increased button size whenever possible, since eye-tracking systems often have larger spatial error than traditional input methods; studies have reported error rates of approximately **0.65 degrees** (roughly 30 pixels) for devices such as the Tobii Eye Tracker 5[2]. We also ensured spacing between consecutive targets of at least **20 pixels**.
- To allow for a greater margin of error, the clickable area of a button should be slightly larger than the displayed button borders (a suggestion by the Bridging Voice team)
- **Visual styling.** Buttons are visually differentiated using color contrast and borders against a clean white background to ensure recognizability.
- **Semantic markup for screen readers.** Interactive elements are implemented using semantic HTML (e.g., `<button>`, `<a>`) whenever possible, ensuring screen readers can understand an item is clickable. ARIA roles are used to communicate the purpose and role of custom elements.

2.2 Hover and Click Feedback

Because eye-gaze interaction relies on tracking gaze position rather than direct physical input, clear visual feedback is essential to help users understand where focus is currently placed and when an action has occurred. To support this, our interface provides two forms of feedback:

- **Focus feedback.** When a button or link receives focus, such as when the user's gaze rests on it, a visible highlight or outline indicates the current focus location without triggering an action.
- **Activation feedback.** When a selection is activated, a distinct visual indicator confirms that the action has been executed.
- **Preventing unintentional double clicks.** Clickable elements are disabled for a short duration after a click. Alternatively, limit how often an element can be activated (ex. max 1 click per second).

2.3 Single-Column Layouts

When displaying multiple options for selection, we implemented single vertical columns, as we believe it is preferred for eye gaze users.

- Single columns eliminate the need for column switching, keeps a consistent scan path, and thus reduces spatial cognitive burden and the likelihood of accidental selection due to jitter.
- We chose vertical layouts as they align with natural reading patterns and minimize complex head movements.
- Single-row horizontal layouts may offer potential advantages, such as distributing more content across wider displays, and representing an area for future exploration.

2.4 Dedicated Vertical Scroll Controls

- Inspired by the design of the **GazeTheWeb**[3]gaze-controlled web browser, we introduced dedicated vertical scroll buttons (up and down) to provide an alternative to continuous scrolling.

[1] https://www.w3.org/TR/WCAG21/#target-size
[2] https://link.springer.com/chapter/10.1007/978-3-030-98404-5_36
[3] https://mamem.github.io/GazeTheWeb/

- This allows users to advance through content in predictable, discrete steps rather than relying on sustained dwell or precise gaze gestures.

2.5 Keyboard Button Navigation

- All interactive elements are operable using standard keyboard navigation (`Tab`, `Shift + Tab`, `Enter`), ensuring compatibility with keyboard-emulating eye-gaze systems. All buttons on a page should be reachable using sequential Tab operations.

2.6 Static Interfaces with All Options Visible

For eye-gaze users, it is difficult to ”hover” without clicking. Thus, no displays/menus should rely on hovering interactions.

- All available actions and options are presented explicitly and remain visible on screen, without relying on hover-only interactions, hidden menus, or time-dependent content (ex. automatic scrolling).
- By making all choices available upfront, users do not need to remember where options are located, anticipate interface changes, or wait for elements to appear or disappear.
- Users can thus explore the interface at their own pace.
- If pop-ups must be used, keep them on screen until they are closed by user action.