

# Why Reducing Search Fatigue is Important

## 1. What – Domain Research

### 1.1 Application Domain

In this project context, a Tamagotchi-style virtual pet (“Calsifer”) is primarily influenced by live chat interactions during streaming sessions. The streamer (Desca) functions as an operator/moderator who triggers special events and modifies system parameters when necessary (e.g., balancing, moderation interventions, recovery actions, or content pacing).

Operator actions occur under live conditions with continuous attention demands (presentation, chat monitoring, stream timing). The operator interface must support fast, accurate intervention with minimal cognitive disruption.

#### **Problem in context:**

When operator controls are distributed across multiple tabs, panels, or tools, the operator must repeatedly search for the correct control and confirm outcomes. This repeated searching creates search fatigue, which can degrade performance over time.

### 1.2 Available Work – Existing Knowledge (Library Evidence)

#### **Search fatigue as a real, modeled phenomenon**

Academic literature models search fatigue as an increase in the effective cost of continued search, affecting behavior such as stopping, pausing, or reducing effort. Carlin’s NBER working paper explicitly frames “search fatigue” as a mechanism with intertemporal effects in search behavior.

[\*nber\*](#)

Ursu, Zhang, and Honka document that people take “search gaps” (breaks) during search and develop/estimate a model in which fatigue rationalizes such breaks; the work connects fatigue to measurable changes in search decisions and effort.

[\*anderson.ucla.edu\*](#)

#### **Search fatigue in operational UI terms (interface searching)**

Although much “search fatigue” literature originates in consumer/product search, the underlying mechanism (effort accumulation; increased marginal cost of continued search) transfers to interface operation whenever repeated “finding” actions are required: locating controls, scanning menus, reorienting after switching contexts, and confirming outcomes.

#### **Task switching and reorientation costs (a key driver of interface search fatigue)**

Research in cognitive control shows that switching tasks imposes measurable “switch costs” in time and errors due to goal shifting and rule activation processes.

[\*arxiv\*](#)

Window management research demonstrates that switching among windows is time-intensive and composed of multiple overhead actions; this supports the claim that distributing controls across windows/tabs increases operational overhead.

[arxiv](#)

### **1.3 Innovation Domain (Contribution in this project context)**

This applied research does not claim a finished product. The innovation contribution is the translation of “search fatigue” literature into concrete operator-interface requirements for a live-stream moderation/control scenario, where:

- Search occurs as locating controls and states under time pressure.
- Fatigue manifests as slower interventions, more errors, and higher cognitive load across long sessions.
- Minimizing between-patch movement (tab/window switching) becomes a primary design objective.

## **2. Why – Purpose & Trade-offso**

### **2.1 Research Purpose**

The purpose is to justify, using existing literature, why reducing search fatigue is important for a live operator/moderator interface and to derive evidence-based principles that reduce search cost and fatigue risk.

Importance is defined in operational terms:

- Faster correct intervention.
- Fewer operator errors.
- Reduced cumulative mental strain during long live sessions.
- Improved reliability and flow of live moderation actions.

Search fatigue models show that fatigue changes behavior and motivates breaks/interruptions; in live operation, such interruptions translate to delayed or missed interventions.

[anderson.ucla.edu](#)

### **2.2 Trade-offs**

Reducing search fatigue by consolidating controls creates potential counter-pressures

#### **1. Clutter vs speed**

Centralizing controls can reduce between-patch search cost, but excessive visible options can increase within-patch visual search and decision time.

#### **2. Flexibility vs consistency**

Highly configurable control panels can introduce inconsistency across sessions, potentially increasing search time if control locations change.

#### **3. Safety vs rapid action**

Adding confirmations reduces accidental triggering but can slow down time-critical actions.

This trade-off framing aligns with DOT-style applied design reasoning: optimizing for the context while preserving usability and safety.

## 3. How – Research Strategies & Methods (DOT)

### 3.1 Library Research

#### Method

A targeted literature review was conducted across:

- Academic search-fatigue modeling (economics/behavior)
- Cognitive task switching and window management research.
- HCI theory explaining search cost in interfaces (information foraging).

#### Core sources used

- Carlin (NBER): formal “Search Fatigue” modeling.

[\*nber\*](#)

- Ursu, Zhang, Honka: empirical evidence and model of fatigue causing search gaps.

[\*anderson.ucla.edu\*](#)

- Jeuris et al.: quantification of window management as time-intensive overhead.

[\*arxiv\*](#)

#### Findings (synthesized from sources)

- Search fatigue increases the effective cost of continued search and leads to behavior changes such as pausing/breaking search.
- Fragmented interfaces raise search costs through switching overhead and reorientation time.
- In live operations, increased search cost plausibly translates to slower interventions and higher error risk (applied inference supported by switching-cost evidence).

## 4. Findings – Why Reducing Search Fatigue Matters

### 4.1 Search fatigue predicts degraded performance over time

Search fatigue research supports the idea that continued searching becomes progressively more “costly,” altering behavior through pauses (“search gaps”) or reduced search depth. In a live operator environment, those same mechanisms map to:

- Hesitation.
- Delays in triggering interventions,
- Reduced willingness to search for “the best” control path.
- And increased likelihood of stopping early.

### 4.2 Interface fragmentation amplifies fatigue through switching overhead

When controls are spread across tabs/windows, the operator must repeatedly:

- Switch contexts.
- Reorient visually.
- Locate the correct control.
- Return to monitoring.

Window management research provides direct evidence that task switching is time-intensive and composed of overhead actions, supporting the claim that distributed controls raise operational “search cost.”

### **4.3 Reduced “between-patch” cost is a primary design lever**

Information foraging theory predicts that reducing the cost of moving between information patches reduces overall search effort. Translating to UI: keeping high-frequency controls in one panel reduces between-patch transitions (tabs/windows), lowering search cost.

## **5. Guideline – Evidence-Based Interface Requirements to Reduce Search Fatigue**

Based on the library findings, an operator control panel intended to reduce search fatigue should implement:

### **1. Consolidation of high-frequency controls in one persistent panel**

Reduces between-patch movement, aligning with information foraging predictions and reducing switching overhead.

### **2. Minimization of window/tab switching during interventions**

Directly targets measured overhead from window management and the cognitive costs associated with switching.

### **3. Persistent visibility of critical state feedback**

Reduces verification searches (searching for confirmation/state elsewhere), decreasing total search episodes and preventing fatigue accumulation inferred from search-fatigue models.

### **4. Structured grouping to prevent within-panel search overload**

Consolidation must not create excessive scanning. Controls should be grouped by function/frequency to lower within-patch search time (trade-off management consistent with DOT applied reasoning).

### **5. Consistency across sessions**

Stable placement supports faster reacquisition of controls and reduces repeated searching caused by layout variability (derived from switching and search-cost logic).

## **6. Conclusion**

Using the DOT framework structure demonstrated in the provided example, this library-only applied research establishes that reducing search fatigue is important in live operator interfaces because:

- Search fatigue increases the effective cost of continued searching and changes behavior (pausing, reducing effort), which is incompatible with time-critical live operation.
- Fragmentation of controls across windows/tabs increases overhead via measurable window management costs.
- HCI theory predicts that reducing between-patch navigation reduces search effort, supporting consolidation as a primary design direction.

This provides an evidence-based foundation for an operator panel design whose central objective is lowering search effort and reducing fatigue risk during long live streaming sessions.