# Biomechanically-Informed Reinforcement Learning for Fatigue-Aware VR Interface Design

**Harshitha Voleti**

**A Thesis**

**in**

**The Department**

**of**

**Computer Science and Software Engineering**

**Presented in Partial Fulfillment of the Requirements**

**for the Degree of**

**Master of Computer Science at**

**Concordia University**

**Montréal, Québec, Canada**

**March 2026**

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Harshitha Voleti**

Entitled: **Biomechanically-Informed Reinforcement Learning for Fatigue-Aware VR Interface Design**

and submitted in partial fulfillment of the requirements for the degree of

**Master of Computer Science**

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

______________________ Chair
*Dr. Anil Ufuk Batmaz*

______________________ Examiner
*Dr. Leila Kosseim*

______________________ Supervisor
*Dr. Charalambos Poullis*

Approved by ______________________
Dr. Joey Paquet, Chair
Department of Computer Science and Software Engineering

__________ 2026 ______________________
Dr. Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

# Abstract


Biomechanically-Informed Reinforcement Learning for Fatigue-Aware VR Interface Design

Harshitha Voleti

Prolonged mid-air interaction in virtual reality (VR) can lead to physical discomfort and arm fatigue, commonly referred to as the gorilla arm effect. This phenomenon negatively affects user performance and experience, highlighting the need for ergonomically informed interaction design in immersive systems.

Incorporating ergonomic considerations into VR user interface (UI) design typically requires extensive human-in-the-loop data collection to assess physical effort and fatigue. Such evaluations are costly, time-consuming, and difficult to scale, limiting their usefulness during early-stage design exploration. As a result, there is a need for alternative methods that can support ergonomic evaluation without relying exclusively on repeated human studies.

Biomechanical user models offer a promising direction, as they represent complex, muscle-actuated human motion and can estimate physical effort and fatigue during interaction. While these models have been used to simulate human movement and behavior in human–computer interaction (HCI) research, their potential role as surrogate evaluators for ergonomic VR UI design remains largely unexplored.

This work presents a framework that leverages biomechanical user models as surrogate users to evaluate and optimize VR interfaces for mid-air interaction. A biomechanical motion agent is trained using reinforcement learning (RL) to perform sequential button-press tasks in VR, generating realistic movement strategies and estimating muscle-level fatigue using a validated three-compartment control with recovery (3CC-r) fatigue model. The simulated fatigue estimates are then used as a feedback signal for an interface optimization agent that learns spatial UI layouts through RL, with the objective of minimizing cumulative user fatigue.

The approach is evaluated by comparing RL-optimized layouts against a manually designed centered baseline and a Bayesian optimized baseline. Results show that fatigue trends predicted by the biomechanical model align with those observed in a human user study, and that interfaces optimized using RL lead to lower perceived fatigue. The framework's extensibility is further demonstrated through a use case involving longer sequential tasks with non-uniform interaction frequencies.

# Acknowledgments

I would like to express my sincere gratitude to my supervisor, Charalambos Poullis, for the opportunity to join his lab and for his constant guidance and support throughout this project. I am especially grateful for his patience during the many debugging challenges this work involved and for his steady encouragement at every stage. His mentorship has been invaluable, and I could not have asked for a better mentor.

I am also grateful to my committee members, Dr. Anil Ufuk Batmaz and Dr. Leila Kosseim, for their thoughtful discussions and valuable feedback on my work.

To the members of the ICT Lab, thank you for the camaraderie, insightful discussions, and encouragement during long days of research. Special thanks to Ken for his help with technical issues whenever needed and to Mucahit for generously answering my HCI-related questions. I am also sincerely grateful to the user participants whose time and cooperation made this thesis possible.

To my parents, Jyothi and Anand, for their love and unwavering support, I am eternally grateful. To my brother, Abhi, thank you for always pushing me to become a better version of myself. And to my grandmother, Lalitha, thank you for raising me with unconditional love.

To my friends and extended family, thank you for your constant encouragement and for keeping me grounded throughout this journey. Your presence and support meant more than I can express.

To everyone who, directly or indirectly, played a role in this journey, I extend my sincere thanks. I also gratefully acknowledge the financial and institutional support provided by NSERC and Concordia University.

*This thesis is dedicated to my grandparents.*

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Virtual reality (VR) is commonly described as an interactive and immersive experience in which users are situated within a responsive virtual environment [32]. With advances in technology, VR has been widely adopted across domains such as entertainment, gaming, training, and robotics, offering increasingly natural and engaging interaction paradigms. Unlike traditional interfaces, VR interaction extends beyond screens into spatial environments, where users directly manipulate virtual content through modalities such as hand tracking, eye gaze, and voice input.

Despite the availability of these interaction techniques, many VR applications continue to rely on mid-air arm movements, either through direct interaction or controller-based input, due to their perceived naturalness and intuitiveness. However, prolonged use of unsupported arm movements where the user's arms are held without any physical support can lead to physical discomfort and fatigue. This phenomenon is commonly referred to as the *gorilla arm effect* [33]. As a result, ergonomics plays a critical role in the design of effective and sustainable VR interfaces.

Despite its importance, interface design in VR has largely prioritized visual appeal and user engagement over ergonomic considerations. Understanding user behavior in immersive environments remains a complex challenge, as it involves analyzing evolving arm postures, movement patterns, and interaction strategies over time. Unlike traditional interfaces, interactions in VR are inherently dynamic and sequential, where user actions unfold over extended periods and are influenced by prior states.

While certain design heuristics (e.g., placing frequently used elements within comfortable reach) provide useful guidelines, they often fail to generalize to long-duration interactions or complex task sequences. A layout that appears ergonomically feasible for isolated interactions may lead to accumulated fatigue when used repeatedly over time. Evaluating such effects typically requires iterative human-in-the-loop studies, which are time-consuming, costly, and difficult to scale. Consequently, ergonomic assessment is often deferred to later stages of development or overlooked entirely in favor of rapid design iteration.

This gap highlights the need for computational methods that can provide early-stage insights into user behavior and ergonomic outcomes, enabling designers to anticipate and mitigate discomfort before deployment. One possible direction is to leverage data-driven approaches, where AI models learn interaction patterns and inform interface design based on observed user behavior. However, such approaches rely on large-scale, high-quality datasets that capture diverse and long-duration interactions, which are often expensive and impractical to collect in VR settings.

An alternative is to replace reliance on real user data with computational user models that simulate human behavior. These models can generate interaction patterns without the need for extensive data collection, enabling scalable and controlled evaluation. Computational user models have been explored in domains such as gaming and animation to produce realistic motion and interaction. A particularly promising class of these models is biomechanical user models, which represent the human body as a muscle-actuated system capable of producing physically plausible movements [4].

When combined with reinforcement learning (RL), biomechanical models can learn control policies that generate realistic and task-oriented behaviors, and have been validated across a range of interactive tasks. Importantly, these models also incorporate physiological fatigue formulations, such as Consumed Endurance (CE) [21] and the 3CC-r muscle fatigue model [38], enabling the estimation of muscle fatigue during interaction.

While prior work has primarily used such models to study motor control and movement realism, their potential for evaluating and informing the ergonomic quality of user interfaces remains largely unexplored, particularly in the context of sequential, long-horizon VR interactions.

## 1.1 Goal

The goal of this thesis is to develop a computational framework that enables fatigue-aware optimization of VR interface layouts using biomechanical user models as surrogate evaluators. Specifically, this work aims to demonstrate that simulated interaction and fatigue feedback can be used to guide interface design decisions without relying on large-scale human data collection.

## 1.2 Methodology

This work introduces a hierarchical learning framework that integrates biomechanical simulation with reinforcement learning to optimize fatigue-aware interfaces.

A low-level agent is trained to control a biomechanical user model to perform interaction tasks within a VR environment. The resulting motion is used to estimate muscle fatigue through an underlying physiological model. A high-level agent then uses these fatigue estimates as feedback to iteratively optimize the spatial configuration of interface elements, to minimize predicted user fatigue over time.

By treating the biomechanical model as a black-box evaluator, the framework enables systematic exploration of interface layouts without relying on handcrafted ergonomic rules or extensive human-in-the-loop evaluation.

The proposed approach is evaluated in simulation by comparing fatigue-aware layouts against two other proposed baselines, demonstrating its potential as a scalable tool for early-stage ergonomic assessment in VR interface design.

## 1.3 Contributions

The main contributions of this thesis are as follows:

- A computational framework for VR interface design that leverages biomechanical user models as surrogate users to estimate interaction-induced muscle fatigue during simulated use.

- A hierarchical, simulation-driven optimization formulation that adapts interface layouts by minimizing predicted cumulative fatigue, eliminating the need for heuristic or rule-based ergonomic guidelines.

- An empirical evaluation, including a controlled user study, demonstrating that layouts optimized using simulated fatigue lead to reduced perceived physical effort and workload.
- An analysis of fatigue-aware interface design in sequential tasks with non-uniform interaction patterns, highlighting how task structure and usage frequency influence optimal layout configurations.

## 1.4 Thesis Structure

The remainder of this thesis is organized as follows. Chapter 2 provides an overview of reinforcement learning, biomechanical modeling, and fatigue modeling. Chapter 3 presents the core contribution of this thesis, corresponding to the paper titled "Designing Fatigue-Aware VR Interfaces via Biomechanical Models," which is currently under review[1]. This chapter includes the methodology, evaluation, and discussion. Finally, Chapter 4 concludes the thesis and outlines directions for future research.

[1] Preprint available at: https://arxiv.org/abs/2603.26031

# Chapter 2

# Background

## 2.1 Introduction

This chapter presents the theoretical foundations underlying this thesis, covering sequential decision-making, learning-based optimization, and biomechanical modeling of human interaction.

It begins by introducing sequential decision-making and its formalization through Markov Decision Processes (MDPs) and Partially Observable Markov Decision Processes (POMDPs) (Section 2.2). Building on this, Reinforcement Learning and Bayesian Optimization are discussed as computational approaches for solving such problems (Sections 2.3 and 2.4).

Finally, the chapter covers Biomechanical Modeling and Fatigue Modeling techniques (Sections 2.5 and 2.6), which provide the foundation for estimating physical effort during interaction and form the basis of the proposed framework.

## 2.2 Sequential Decision Making

Sequential decision making refers to problems in which an agent must choose actions over multiple time steps while interacting with an environment whose state evolves as a consequence of those actions. Unlike single-shot decision problems, many real-world tasks require a sequence of interdependent decisions, where earlier actions influence future observations, available actions, and outcomes. Examples include human–computer interaction (HCI), robotic manipulation, and adaptive control, where agents must continually react to feedback from the environment.

Modeling such problems is challenging due to the presence of uncertainty, delayed consequences of actions, and the need to reason about long-term objectives. To address these challenges, formal frameworks have been developed to model sequential decision-making problems and to compute policies that maximize long-term reward. In the following, we describe the Markov Decision Process (MDP) formulation, which provides a foundational model for sequential decision making under full observability.

### 2.2.1 Markov Decision Processes

A Markov Decision Process (MDP) is defined as a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ is the state space representing the possible configurations of the environment, $\mathcal{A}$ is the action space representing the set of actions available to the agent, $P(s' \mid s, a)$ is the transition probability distribution describing the likelihood of transitioning from state $s$ to state $s'$ after taking action $a$, $R(s, a)$ is the reward function specifying the immediate reward obtained by executing action $a$ in state $s$, and $\gamma \in [0, 1)$ is the discount factor that determines the relative importance of future rewards.

At each timestep $t$, the agent observes the current state $s_t \in \mathcal{S}$, selects an action $a_t \in \mathcal{A}$ according to a policy $\pi(a \mid s)$, receives a scalar reward $r_t = R(s_t, a_t)$, and transitions to the next state $s_{t+1} \sim P(\cdot \mid s_t, a_t)$.

The goal is to learn a parameterized policy $\pi_\theta$ that maximizes the expected discounted return:

$$J(\theta) = E_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right], \tag{1}$$

where $\tau = (s_0, a_0, r_0, s_1, \dots)$ denotes a trajectory induced by the policy.

This objective reflects the goal of selecting actions that lead to desirable long-term outcomes rather than short-sighted gains. The return $\sum_{t=0}^{\infty} \gamma^t r_t$ captures the cumulative reward over time, while the discount factor $\gamma$ ensures that immediate rewards are prioritized but future consequences are still taken into account. Maximizing the expected return therefore encourages the agent to learn behaviors that balance immediate benefits with long-term performance, which is essential in sequential decision-making problems where actions may have delayed effects.

Although widely used, the MDP formulation assumes that the agent has full access to the true Markovian state. In practice, this assumption rarely holds in embodied or HCI scenarios. For instance, in VR environments, internal user states such as muscle fatigue, cognitive load, or intent are not directly observable; however, they influence interaction behavior and are only indirectly reflected through observable signals such as controller positions, headset motion, or interaction events.

This limitation motivates the use of partially observable formulations, discussed in the following section.

### 2.2.2 Partially Observable Markov Decision Processes

Many real-world environments are partially observable: the agent cannot directly access the complete system state and instead receives observations that provide incomplete or noisy information. Such environments are more accurately modeled as Partially Observable Markov Decision Processes (POMDPs), defined as the tuple $(\mathcal{S}, \mathcal{A}, P, R, \Omega, \mathcal{O}, \gamma)$. A POMDP extends an MDP by introducing an observation space $\Omega$, representing the set of observations available to the agent, and an observation model $\mathcal{O}(o \mid s)$, which defines the probability of observing $o \in \Omega$ given that the environment is in state $s \in \mathcal{S}$.

At each timestep $t$, the environment is in a latent state $s_t$, while the agent receives an observation $o_t \in \Omega$ sampled according to $o_t \sim \mathcal{O}(\cdot \mid s_t)$ . Because the true state is not directly observable, the agent must base decisions on the history of observations and actions. In principle, optimal policies operate on belief states, which represent posterior distributions over latent states. In practice, many RL methods approximate this dependence using observation-conditioned or recurrent policies.

## 2.3 Reinforcement Learning

Reinforcement learning (RL) addresses sequential decision-making problems in which actions influence future states, and cumulative reward must be optimized through interaction. While the theoretical formulation of RL is commonly based on MDPs, many real-world applications, including VR interaction scenarios, are inherently partially observable. As a result, RL methods are often applied in POMDP settings, where policies are learned directly from observations or histories rather than fully observed states. The environment is typically modeled as an MDP for theoretical analysis, and the objective is to learn a policy that maximizes expected return.

RL methods can be broadly categorized into value-based and policy-based approaches, based on how the decision-making strategy is represented and learned. Value-based methods estimate a value function, such as the state-value function $V(s)$ or the action-value function $Q(s, a)$, which quantifies the expected return of states or state-action pairs. A policy is then implicitly derived by selecting actions that maximize these estimated values. While effective in discrete action spaces, value-based methods can become challenging to apply in high-dimensional or continuous action domains, where selecting the optimal action requires solving a potentially complex optimization problem. Examples include Q-learning [62] and its variants.

In contrast, policy-based methods directly parameterize the policy $\pi_\theta(a \mid s)$ and optimize it without explicitly deriving actions from a value function. This enables more flexible representations of stochastic policies and makes such methods particularly suitable for continuous control and high-dimensional decision spaces. Methods such as REINFORCE [63] belong to this class and optimize the policy using gradient ascent on expected return.

Despite their effectiveness, classical RL methods often rely on tabular representations, hand-crafted features, or simple function approximators, which limit their scalability in complex environments. To address this limitation, these approaches have been extended to deep reinforcement learning (DRL), where neural networks are used to approximate value functions, policies, or both. This enables agents to operate directly on high-dimensional inputs such as images or raw sensor data. Some examples include Deep Q-Networks (DQN) in the value-based setting [42], and methods such as Soft Actor-Critic (SAC) [18], and Proximal Policy Optimization (PPO) [55] in the policy-based setting. Within this framework, policy-based methods are commonly optimized using policy gradient techniques, where policy parameters are adjusted directly to improve expected return.

### 2.3.1 Policy Gradient Methods under Partial Observability

Many RL algorithms operate directly on observations without explicitly maintaining belief states. Policy gradient methods are particularly suitable in such settings, as they directly optimize policies and can learn effective behaviors from high-dimensional inputs.

Rather than estimating value functions and deriving actions indirectly, policy gradient methods optimize the policy by adjusting its parameters in a direction that increases the likelihood of actions that lead to higher returns. Intuitively, the objective is to reinforce actions that resulted in better-than-expected outcomes, while discouraging those that performed poorly.

This idea can be formalized by computing gradients of the expected return with respect to the policy parameters:

$$\nabla_\theta J(\theta) = E_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t \mid o_t) \hat{A}_t\right] \tag{2}$$

This expression represents how the policy should be updated to improve performance. The term $\nabla_\theta \log \pi_\theta(a_t \mid o_t)$ increases the probability of selecting action $a_t$ in observation $o_t$, while the advantage term $\hat{A}_t$ determines the direction and magnitude of the update. When $\hat{A}_t$ is positive, the action performed better than expected, and its probability is increased; when it is negative, the probability is decreased. In this way, the policy gradually shifts toward actions that yield higher long-term returns.

The advantage function $\hat{A}_t$ serves as a measure of relative performance, comparing the observed return to a baseline estimate of expected value. This helps reduce variance in the updates by focusing on how much better or worse an action is compared to typical outcomes.

Nevertheless, pure policy gradient methods still rely on returns computed from sampled trajectories, which can vary significantly due to the stochastic nature of the environment and the limited number of samples used to estimate them. Actor-critic methods address this issue by introducing a value function estimator, referred to as the critic, that evaluates the current policy. The policy itself is referred to as the actor. The critic provides an estimate of the value function, which is used to compute the advantage and guide the actor's updates. This interaction between actor and critic improves stability and sample efficiency, particularly in continuous control and partially observable environments.

However, even with these improvements, actor–critic methods can still suffer from unstable training when policy updates are too large, causing performance to collapse. This motivates the use of algorithms that explicitly constrain how much the policy is allowed to change at each update.

### 2.3.2 Proximal Policy Optimization

Proximal Policy Optimization (PPO) is an actor–critic policy gradient algorithm designed to improve training stability by limiting the deviation between successive policy updates. Instead of performing unconstrained updates, PPO optimizes a surrogate objective that penalizes large changes to the policy.

$$L^{\mathrm{PPO}}(\theta) = E_t \left[ \min \left( r_t(\theta)\hat{A}_t,\ \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right] \tag{3}$$

where

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid o_t)}{\pi_{\theta_{\mathrm{old}}}(a_t \mid o_t)} \tag{4}$$

is the probability ratio between the updated policy and the previous policy.

The ratio $r_t(\theta)$ measures how much the new policy deviates from the old policy for a given action. The clipping operation constrains this ratio to remain within a predefined range $[1-\epsilon, 1+\epsilon]$, preventing excessively large updates. The objective takes the minimum between the unclipped and clipped terms, effectively discouraging updates that would push the policy too far from its previous behavior. When the advantage $\hat{A}_t$ is positive, the objective encourages increasing the probability of the action, but only up to the clipping threshold. Similarly, when $\hat{A}_t$ is negative, it discourages the action while limiting the magnitude of the decrease.

In this way, PPO achieves a balance between improving the policy and maintaining stability, leading to more reliable and efficient training in practice.

## 2.4 Bayesian Optimization

Bayesian optimization (BO) is a sequential optimization framework designed to find the maximum of expensive, black-box objective functions using as few evaluations as possible.

In some decision problems, actions are taken primarily to gather information about an unknown objective rather than to influence evolving system dynamics. In such cases, the environment state remains fixed, while the agent's internal belief evolves over time. This setting is commonly referred to as a bandit problem. When optimizing an unknown objective function $R(\mathbf{x})$ that is expensive to evaluate, the agent can query the function at different inputs $\mathbf{x}$ and observe noisy outputs. The goal is to identify the maximizer of $R(\mathbf{x})$ using as few evaluations as possible. In this setting, the agent's state can be interpreted as a belief distribution over the unknown function, $s_t = p(R \mid h_t)$, where $h_t$ denotes the history of observations collected up to timestep $t$.

A common way to represent this belief distribution is through Gaussian Process (GP) regression, which provides both a predictive mean and uncertainty estimate over the objective. Given this posterior belief, an acquisition function is used to determine the next evaluation point: $\mathbf{x}_t = \pi(s_t)$. Common acquisition strategies include Expected Improvement (EI), Knowledge Gradient (KG), and Thompson Sampling.

Bayesian optimization is particularly useful when each evaluation of the objective function is computationally expensive, time-consuming, or resource-intensive, and when only a limited number of evaluations can be performed. In this thesis, bayesian optimization is used as a baseline method for optimizing interface layouts.

## 2.5 Biomechanical Modeling

Musculoskeletal models are computational simulation frameworks used to analyze and predict human movement based on biomechanical principles. These models represent the skeletal system through articulated rigid bodies connected by joints, which constrain kinematic motion. Muscle tendon units are incorporated to generate forces and joint torques, enabling movement. The activation of these muscles is driven by control inputs, often referred to as muscle activation signals, which determine the timing and magnitude of force production [6].

### 2.5.1 Inverse Simulation

Biomechanical models are predominantly used within inverse simulation pipelines. These approaches begin with motion capture data collection, where physical markers are mapped to virtual model markers. The model is then scaled to match the participant's anthropometric dimensions.

Inverse kinematics is used to estimate joint angles from the recorded motion by finding the set of joint configurations that best reproduce the observed marker positions. This is followed by inverse dynamics, which computes the joint torques required to produce the estimated motion based on the system's dynamics. Finally, static optimization or computed muscle control methods estimate muscle activations corresponding to each pose [1].

However, acquiring and processing motion capture data is labor-intensive and requires extensive calibration, marker placement, and post-processing. For dynamic and interactive HCI tasks, continuously tracking real users can be impractical and limit scalability. This makes motion-driven inverse simulation less suitable for iterative interface design, where rapid evaluation across many conditions is required.

### 2.5.2 Forward Simulation

In contrast to inverse simulation, forward simulation begins with control inputs applied to the model's actuators and computes the resulting motion through system dynamics.

#### Torque-Based Models

Early works in computer graphics and animation relied on torque-based models for motion synthesis, often using trajectory optimization frameworks such as spacetime constraints [45]. With the advancement of deep learning, RL techniques were subsequently leveraged to learn motor skills directly from interaction with the simulated environment [25]. Torque-based models offer relatively simple dynamics and computational efficiency.

The extension of torque-based models to HCI applications was later explored in works such as [4], which integrated fatigue modeling to estimate arm fatigue during mid-air interaction (as shown in Figure 2.1). Similarly, [11] demonstrated that RL-controlled torque-based models can reproduce established laws of human movement, including Fitts' Law [13] and the $2/3$ Power Law [29].

Despite these advantages, torque-based models lack explicit physiological grounding. Torque limits are typically specified independently for each joint and do not account for the coupled behavior of muscle-tendon units spanning multiple joints. Consequently, while such models can generate visually realistic motion, they may produce torque patterns that are not fully physiologically plausible.



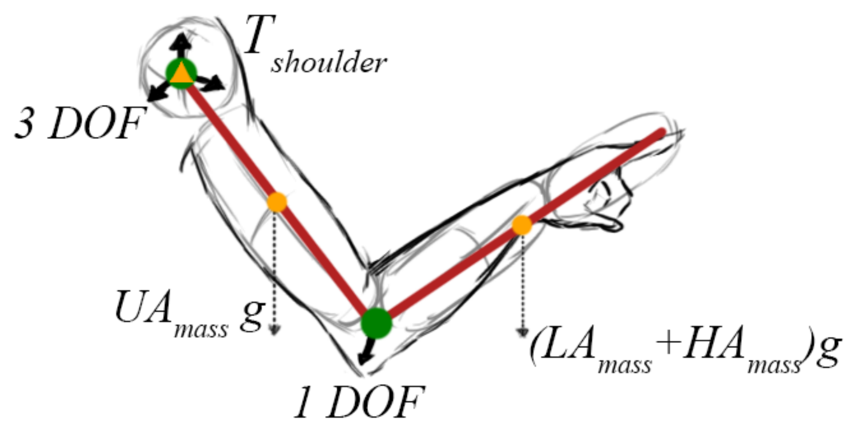
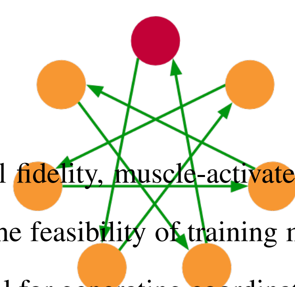



**Figure 3. Forces acting on our biomechanical upper limb model. The limbs are modeled as rigid bodies (red) connected with joints (green). The degrees of freedom (DOF) of each joint are denoted by the arrows at the respective joint.**

Figure 2: Torque-based upper-limb model. The arm is modeled as a 4-DOF serial chain with three shoulder joints and one elbow joint. Each segment is actuated by joint torques, with gravitational forces acting on each segment. Adapted from [4].

### Muscle-Activated Models



To achieve greater physiological fidelity, muscle-activated models have been increasingly explored. Early works demonstrated the feasibility of training muscle-driven biomechanical systems using RL, highlighting their potential for generating coordinated multi-joint movements [27].

**Figure 4. ISO 9241-9 reciprocal pointing task with 7 targets. Agents point to highlighted (red) target. Targets highlight in the pattern indicated by the arrows.**

Within the context of HCI, [1] laid foundational work by introducing motion-captured biomechanical simulation to validate interaction paradigms and analyze physical effort in mid-air tasks. These approaches emphasized the importance of physiologically grounded modeling for evaluating interaction design.

$\vec{T}L(\vec{T})$ at the joint and the desired muscle activation $\vec{M}_A = [M_{A_1}^+, M_{A_1}^-, M_{A_2}^+, M_{A_2}^-, M_{A_3}^+, M_{A_3}^-]^\top$ given by the 3CC-$r$ model:

However, many existing muscle-activated models rely primarily on joint-level state representations in the observation space and do not explicitly incorporate perceptual inputs. More recent

$$Effort_C(\vec{T}) = \left\| \frac{\vec{M}_A}{100} - \vec{T}L(\vec{T}) \right\| \quad (8)$$



The advantage of this over using $M_F$ directly is that the cumu-

United Kingdom
Finland

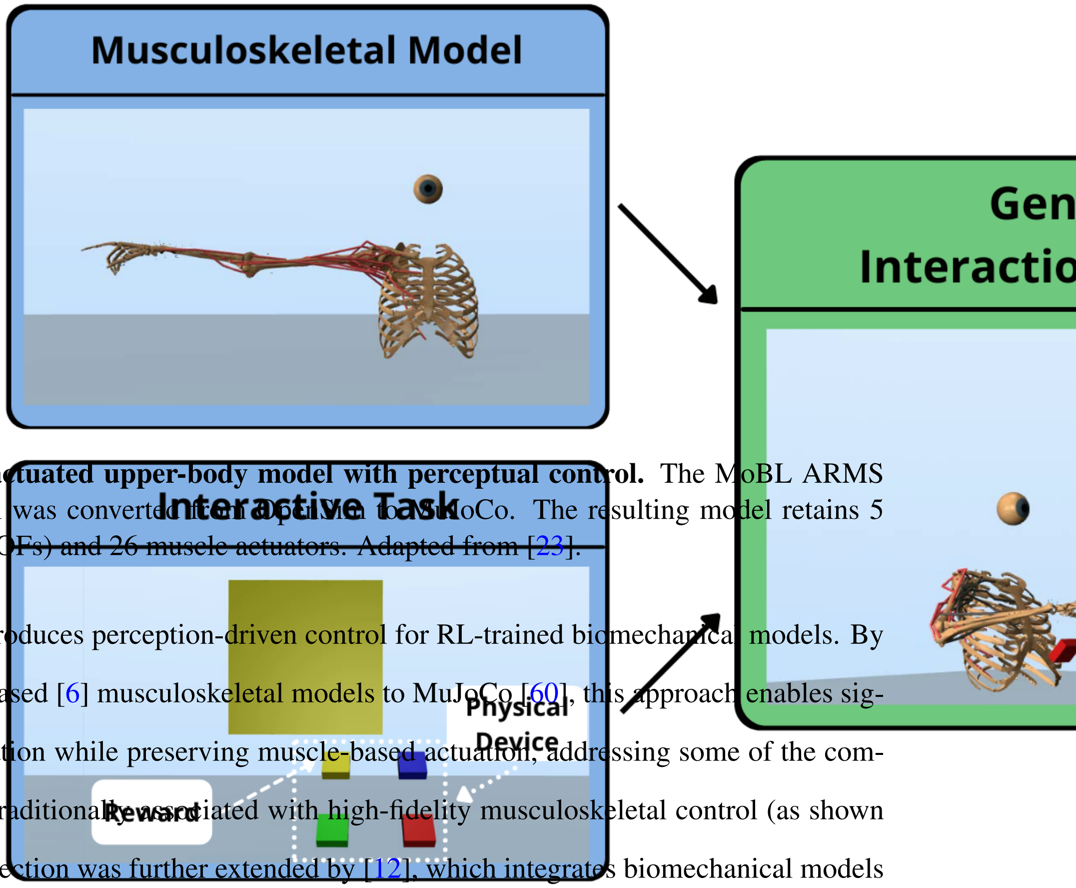



Figure 2.2: **Muscle-actuated upper-body model with perceptual control.** The MoBL ARMS musculoskeletal model was converted from OpenSim to MuJoCo. The resulting model retains 5 degrees of freedom (DOFs) and 26 muscle actuators. Adapted from [23].

work, such as [23], introduces perception-driven control for RL-trained biomechanical models. By converting OpenSim-based [6] musculoskeletal models to MuJoCo [60], this approach enables significantly faster simulation while preserving muscle-based actuation, addressing some of the computational challenges traditionally associated with high-fidelity musculoskeletal control (as shown in Figure 2.2). This direction was further extended by [12], which integrates biomechanical models with Unity-based VR applications to enable automated and scalable testing across diverse virtual environments.

Because these models explicitly represent muscle activations, they can also be used to estimate muscle fatigue during interaction.

## 2.6 Fatigue Modeling

Quantifying physical effort and fatigue is essential for evaluating and optimizing mid-air interaction techniques in virtual reality. Several fatigue models have been proposed in the literature, each with different assumptions and limitations.

### 2.6.1 Consumed Endurance

Consumed Endurance (CE) [21] is one of the earliest objective models proposed to quantify arm fatigue during mid-air interaction. The model is derived from Rohmert's endurance time (ET)



**Figure 1: We present an approach for generative simulation o interacting with physical devices. The users are modelled wi perception models, and we use deep reinforcement learning a showcase, we apply a state-of-the-art upper body model to reaction, and parking a remote control car via joystick.**


## ABSTRACT

Forward biomechanical simulation in HCI holds great promise as a tool for evaluation, design, and engineering of user interfaces. Although reinforcement learning (RL) has been used to simulate biomechanics in interaction, prior work has relied on unrealistic assumptions about the control problem involved, which limits the


*Also with University of Bergen.



*UIST '22, October 29-November 2, 2022, Bend, OR, USA*

curve [51], which relates sustained muscular load to the maximum time a muscle can maintain that load before exhaustion.

CE first estimates the instantaneous shoulder torque generated during an interaction. This torque is accumulated and normalized by a maximum shoulder torque (Max_Torque) derived from literature, yielding a relative exertion level. Rohmert's empirical endurance function is then used to map this normalized exertion to a predicted maximum holding time, given by:

$$ET_{\text{Rohmert}} = \frac{1236.5}{\left(\frac{\text{Torque}}{\text{Max\_Torque}} \times 100 - 15\right)^{0.618}} - 72.5 \tag{5}$$

The CE score is subsequently computed as the percentage of endurance used during the interaction:

$$CE = \frac{\text{TotalTime}}{ET_{\text{Rohmert}}} \times 100 \tag{6}$$

However, CE is fundamentally based on Rohmert's curve, which assumes static load conditions. It does not explicitly model recovery during rest periods nor account for dynamically varying torque levels over time. Consequently, CE provides limited accuracy when applied to realistic VR interaction scenarios involving continuous movement and fluctuating biomechanical loads.

### 2.6.2 Three-Compartment Controller (3CC-r) Model

The next commonly used fatigue model is the 3CC-r model [38] as shown in Figure 2.3. This motor-unit-based model assumes that muscles can exist in one of three states:

- **Active** ($M_A$): motor units currently involved in the task,
- **Fatigued** ($M_F$): motor units that are temporarily unable to generate force,
- **Resting** ($M_R$): inactive motor units available for recruitment.

Each compartment is expressed as a percentage of maximum voluntary contraction (%MVC), with the constraint:

$$M_A + M_F + M_R = 100. \tag{7}$$

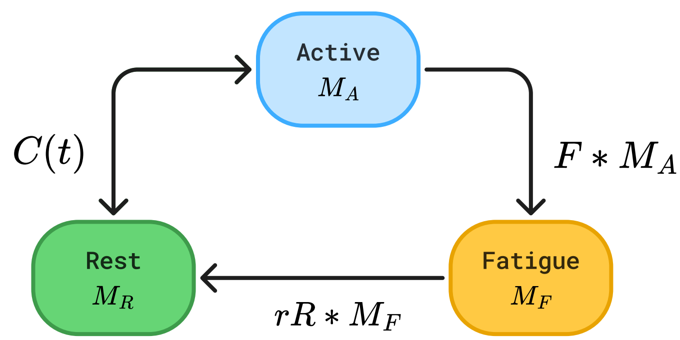



Figure 2.3: **Three-compartment controller model.** Depicts the structure and flows between the compartments, described mathematically by Eq. (8).

Once a motor unit (MU) is activated, its force production gradually decays over time due to fatigue. The current approach combines compartment theory with control theory to rigorously define system behavior in accordance with muscle physiology. This behavior is mathematically described in Eq. 8:

$$\begin{aligned} \frac{dM_R}{dt} &= -C(t) + rR \cdot M_F \\ \frac{dM_A}{dt} &= C(t) - F \cdot M_A \\ \frac{dM_F}{dt} &= F \cdot M_A - rR \cdot M_F \end{aligned} \tag{8}$$

where $C(t)$ represents the time-varying muscle activation–deactivation drive. The parameters $F$ and $rR$ define the rates at which motor units fatigue and recover, respectively. The rest multiplier $r$ was set to 7.5 based on a sensitivity analysis performed in [4], while $F = 0.0146$ and $R = 0.0022$ were adopted from [24].

The 3CC-r model extends the original 3CC model by including a rest-recovery factor $r$, which enhances recovery when the target load ($T_L$) is zero. This modification better represents perceived fatigue estimates from user studies [38]:

$$rR = \begin{cases} r \cdot R, & \text{if } M_A \geq T_L \\ R, & \text{otherwise} \end{cases} \tag{9}$$

The activation function $C(t)$ produces the target load $T_L$ (in percent) by controlling the size of $M_A$ and the availability of $M_R$. It is defined as:

$$C(t) = \begin{cases} L_D \cdot (T_L - M_A), & \text{if } M_A < T_L \text{ and } M_R > T_L - M_A \\ L_D \cdot M_R, & \text{if } M_A < T_L \text{ and } M_R \leq T_L - M_A \\ L_R \cdot (T_L - M_A), & \text{if } M_A \geq T_L \end{cases} \tag{10}$$

The target load $T_L$ is defined as:

$$T_L = \left( \frac{T_{\text{current}}}{T_{\text{max}}} \right) \times 100. \tag{11}$$

Here, $L_D$ controls the rate of force development and $L_R$ controls relaxation. Following the sensitivity analysis by [64], both parameters are set to $10$.

Due to its ability to model both fatigue and recovery dynamics, the 3CC-r model has been widely adopted for estimating muscle fatigue during dynamic interaction tasks.

## 2.7 Summary

This chapter reviewed the key foundations required for this thesis. Sequential decision-making was formulated as an MDP and extended to POMDP to account for partial observability in real-world settings such as VR interaction.

RL was presented as a framework for learning policies from interactions, with policy-gradient and actor–critic methods enabling stable learning in high-dimensional, continuous domains. PPO was discussed as a practical approach for improving training stability. BO was introduced as a complementary method for optimizing expensive black-box objectives with limited evaluations.

Finally, biomechanical modeling was described as a means of simulating human movement and estimating physical effort. Forward simulation with muscle-activated models enables physiologically meaningful evaluation, while fatigue models such as 3CC-r capture dynamic fatigue and recovery during interaction.

These components form the basis for the proposed fatigue-aware interface optimization framework.

# Chapter 3

# Designing Fatigue-Aware VR Interfaces via Biomechanical Models

*The following is a verbatim copy of the manuscript currently under review, titled "Designing Fatigue-Aware VR Interfaces via Biomechanical Models," authored by Harshitha Voleti and Charalambos Poullis.*

## 3.1 Abstract

Prolonged mid-air interaction in virtual reality (VR) can lead to arm fatigue and discomfort, negatively impacting user experience. Incorporating ergonomic considerations into VR user interface (UI) design typically requires extensive human-in-the-loop evaluation. While biomechanical models have been used to simulate human behavior in HCI tasks, their use as surrogate users for ergonomic VR UI design remains underexplored. We propose a framework that leverages biomechanical user models to evaluate and optimize VR interfaces for mid-air interaction. A motion agent is trained to perform button-press tasks in VR under sequential interaction conditions, executing realistic movement strategies and estimating muscle-level effort using a validated three-compartment control with recovery (3CC-r) fatigue model. The simulated fatigue output is then used as a feedback signal for an interface agent that optimizes the spatial layout of UI elements via reinforcement learning (RL), with the goal of minimizing user fatigue. We compare the RL-optimized layout

against a manually-designed centered baseline and a Bayesian optimized baseline, and show that fatigue trends predicted by the biomechanical model align with those observed in human users. Furthermore, RL-optimized layout using simulated fatigue feedback resulted in significantly lower perceived fatigue in a subsequent human user study. We further demonstrate the extensibility of our framework through a simulation-based case study involving longer sequential tasks with non-uniform interaction frequencies, examining optimization behavior under increased task complexity. To our knowledge, this is the first work to use simulated biomechanical muscle fatigue as a direct optimization signal for VR UI layout design. Our results highlight the potential of biomechanical user models as effective surrogate tools for ergonomic VR interface design, enabling more efficient early-stage design iteration with reduced reliance on extensive human participation.

## 3.2 Introduction

Designing ergonomic user interfaces for VR remains a persistent challenge. Most VR applications rely on mid-air interaction, often requiring repetitive reaching and sustained arm elevation. These interaction patterns can quickly lead to physical discomfort and muscle fatigue, commonly referred to as the "gorilla arm" effect [33]. This fatigue reduces user comfort, degrades task performance, and limits long-term usability, making physical ergonomics a critical yet frequently under-addressed aspect of VR interface design.

Accounting for physical effort during interface design is inherently difficult. Anticipating how users will move, reach, and sustain postures requires contextual knowledge of task flow and behavior that is typically unavailable during early design stages. Consequently, layouts are often optimized for visual structure or task efficiency, while physical demands are evaluated only after implementation through user studies.

This gap motivates methods that can estimate physical effort during continuous interaction and inform layout decisions prior to human evaluation. Biomechanical user models offer this potential. As muscle-actuated systems, they can simulate realistic motion and incorporate fatigue formulations such as Consumed Endurance (CE) [21] and the 3CC-r model [24]. While prior work has used such models to validate motion realism or correlate simulated outcomes with user reports, their use as proactive design tools for optimizing interface layouts remains largely unexplored—particularly in VR, where sustained mid-air interaction dominates.

In this work, we repurpose a biomechanical user model as a surrogate user for ergonomically informed interface design. The simulated user interacts with alternative sequential layouts, generating muscle-level effort signals that are aggregated into cumulative fatigue estimates. These predictions are then used directly to evaluate and optimize interface configurations before human-in-the-loop testing.

Our framework consists of two components: (1) a low-level motion agent that executes interaction tasks and estimates muscle effort, which is aggregated into cumulative fatigue, and (2) a higher-level interface agent that proposes and refines button layouts based on predicted cumulative fatigue. The biomechanical user model is treated as a black-box ergonomic evaluator, enabling systematic exploration of layout alternatives without hand-crafted ergonomic heuristics or auxiliary task-completion rewards. The optimization objective is defined solely in terms of predicted muscle fatigue, penalizing long reaches, awkward postures, and sustained activation patterns.

We evaluate RL-optimized fatigue-aware layouts against a static baseline and a Bayesian optimization baseline in a controlled user study. Results show that simulated fatigue patterns across layouts align with those observed in human participants. Furthermore, RL-optimized layouts reduce perceived physical effort and subjective workload compared to both baselines.

In summary, our contributions are:

1. A fatigue-aware VR interface design framework that leverages biomechanical user models as surrogate users to estimate physical effort during continuous interaction at design time.
2. A simulation-driven optimization approach that updates interface layouts to minimize predicted cumulative muscle fatigue without relying on heuristic design rules.
3. Empirical validation demonstrating that layouts optimized using simulated fatigue reduce perceived physical effort and overall workload in a controlled user study.
4. A use-case application of the proposed framework to longer sequential tasks with non-uniform button usage frequencies, examining how cumulative fatigue and usage

## 3.3 Related Work

### 3.3.1 Evolution of Adaptive User Interfaces

Designing adaptive user interfaces is a complex optimization problem, and the techniques used to address this challenge have evolved considerably over time. Early adaptive UI systems primarily relied on manually specified rules [46], heuristics [17], and explicit user feedback [22]. While these approaches allow designers to encode domain knowledge directly, defining and maintaining explicit rules quickly becomes tedious and difficult to scale.

To reduce manual effort, subsequent work explored optimization-based approaches for interface adaptation, including genetic algorithms [52, 49], cost-function-driven optimization [14, 53], and combinatorial optimization techniques [5, 36, 58]. Although these methods enable systematic exploration of the design space, they often require carefully engineered constraints to ensure feasibility and usability. Defining such constraints is challenging, as these constraints may conflict with one another or overly restrict the solution space.

Data-driven approaches later leveraged observed user behavior to infer preferences and adaptation strategies. Early machine learning methods [31, 40, 48, 50] enabled automatic adaptation from interaction data, and more recent work has explored deep neural networks for interface optimization [57, 7]. However, these methods often require large datasets, limiting their practicality in interactive VR settings.

To address data efficiency concerns, online learning approaches such as multi-armed bandits and Bayesian optimization have gained attention. Bandit-based methods iteratively update adaptation strategies by balancing exploration and exploitation based on observed feedback, allowing systems to adapt effectively with limited data [26, 37]. Similarly, Bayesian optimization treats interface adaptation as a black-box optimization problem, using probabilistic surrogate models to efficiently search for high-performing designs with few evaluations [8, 28]. While both approaches are effective in low-data settings, they typically rely on sparse, per-design evaluations and are sensitive to noise in the feedback signal, which can affect their ability to resolve subtle performance differences across design alternatives.

To address these limitations, RL has been explored as a mechanism for adaptive interface design, as it enables optimization over long-term interaction sequences by maximizing cumulative reward. RL-based AUIs have demonstrated the ability to adapt layouts or interaction strategies over time, accounting for temporal dependencies in user behavior [16, 59, 30, 10, 15]. While these approaches establish RL as a powerful tool for interface optimization, they primarily focus on task efficiency or user preference and do not explicitly consider physical ergonomics as an optimization objective. Crucially, physical effort and fatigue are not directly observable from interaction traces alone, making them difficult to optimize without an explicit physiological or user effort model.

### 3.3.2 Ergonomic Considerations in XR Interaction

Physical ergonomics has received increasing attention in XR interface design, but remains underexplored compared to cognitive and contextual factors, particularly as an optimization signal for adaptive layouts. Prolonged mid-air interaction in VR often involves extended reach, repetitive arm movements, and overhead gestures, which can cause discomfort and cumulative fatigue. Early ergonomic approaches aimed to retarget interface elements to more physically comfortable locations using spatial warping and optimization techniques [43]. XRgonomics [9] provided a more systematic framework, estimating ergonomic cost using metrics such as RULA scores, cumulative effort, and muscle activations derived from biomechanical models. However, these approaches primarily focus on static postures or discrete evaluation points, failing to account for fatigue accumulation during extended, continuous interactions. More recent work has begun introducing fatigue-aware interaction strategies. For example, AlphaPIG [34] continuously adjusts gesture interaction parameters based on a real-time fatigue model so that users receive progressively more assistance only as physical fatigue increases, thereby prolonging comfortable interaction without manual tuning. Such approaches enable designers to explore fatigue-aware interaction strategies systematically, but generally focus on interaction techniques rather than interface layout, leaving an opportunity to optimize UI placement based on physical strain.

### 3.3.3 Fatigue Assessment in Virtual Reality

Accurate fatigue assessment is central to ergonomics-aware design. In VR, prolonged mid-air interaction can produce significant arm fatigue. Various methods exist to quantify this, including physiological measurements such as blood flow, heart rate, and oxygen consumption, as well as subjective self-report instruments such as the Borg Rating of Perceived Exertion (RPE), NASA-TLX, and Likert-scale questionnaires. Although these methods provide valuable insights into user experience, they typically require continuous monitoring or post-hoc reporting, limiting their suitability for real-time or design-time interface adaptation. Computational models offer an alternative for fatigue estimation. One widely used approach is Consumed Endurance (CE) [21], which estimates fatigue based on Rohmert's endurance time curve and correlates with subjective exertion ratings. However, CE assumes that interactions below a fixed percentage of maximum strength do not contribute to fatigue and does not explicitly model recovery during rest, which can lead to inaccurate estimates in interactive scenarios. The 3CC fatigue model addresses these limitations by providing a fine-grained, interaction-oriented framework suitable for HCI applications [64], and has been shown to predict perceived exertion in mid-air tasks with strong correspondence to Borg CR10 ratings [24]. It was later extended to 3CC-r to better support intermittent interactions by introducing an explicit recovery multiplier, validated against joint-specific perceived fatigue [38]. More recently, hybrid models such as NICER [35] combine torque-based estimation with empirically measured muscle contraction and recovery terms, improving prediction of above-shoulder fatigue. However, these metrics are primarily evaluated as post-hoc analytical tools rather than as actionable signals for interface design.

### 3.3.4 Biomechanical Modeling in HCI

As discussed, machine learning optimization often depends on large amounts of data, which can be difficult or costly to obtain. Biomechanical simulations offer a promising alternative by generating rich, controlled datasets that capture human motion and physical effort. In HCI, such models have gained attention for their ability to objectively estimate muscle-level effort and fatigue, providing a more precise complement to subjective or physiological assessments [1]. Early biomechanical models relied heavily on motion data. This involved mapping physical to virtual markers, scaling the model to match the subject's dimensions, computing joint angles through inverse kinematics, and estimating muscle activations [2]. In contrast to these motion-driven approaches, Cheema et al. [4] demonstrated that torque-driven simulations could predict fatigue and task performance in silico, bypassing the need for human participants. Later work showed that RL could train torque-based models to reproduce human-like movement strategies that follow Fitts' Law [13] and the Two-Thirds Power Law [29] [11]. Building on these advances, Ikkala et al. [23] developed muscle-actuated biomechanical user models capable of performing a wide range of HCI tasks, enabling more realistic simulations of motor control and fatigue. These models have also been applied to diverse interaction scenarios [20, 44, 41]. Most recently, Fischer et al. [12] introduced Sim2VR, integrating biomechanical simulation with VR applications, which allows simulated users to train and interact in immersive environments.

While prior work has demonstrated that biomechanical simulations can execute and analyze interactive VR tasks, their outputs have primarily been used for validation, behavioral analysis, or performance prediction. The design implications of simulated muscle activity have remained largely unexplored. In this work, we shift the role of biomechanical modeling from analysis to design optimization. Specifically, we incorporate muscle-level fatigue estimates as the objective signal in an RL pipeline that iteratively refines UI placement. Rather than evaluating completed interfaces post hoc, the simulation directly informs layout generation, enabling ergonomic considerations to shape interface structure during the design process without requiring additional human data collection.

## 3.4 Methodology

We formulate VR interface layout optimization as a hierarchical simulation problem. A low-level motion agent, implemented as a muscle-actuated biomechanical user model, executes interaction tasks and produces muscle activation signals. These signals are aggregated into fatigue estimates, which define the objective optimized by a higher-level UI agent that updates interface layouts.

We first present the overall simulation–optimization framework and its integration with the

SIM2VR pipeline. We then describe the interaction task used to evaluate candidate layouts, followed by details of the motion agent responsible for executing the task and estimating fatigue. Finally, we formalize the UI agent and its reinforcement learning formulation for interface layout optimization.

### 3.4.1 Framework

Our framework builds on SIM2VR [12], which integrates a muscle-actuated biomechanical user model with Unity-based VR applications. In SIM2VR, the model is trained via RL in Python and communicates with the VR environment through a ZMQ interface, enabling goal-directed arm movements and interaction with virtual objects. We extend this pipeline by introducing a UI agent that proposes candidate interface layouts (Fig. 3.1). For each layout, the configuration is transmitted to the VR environment using the existing SIM2VR modules. The biomechanical motion agent then executes the task on the updated interface. During execution, muscle activations are recorded at each timestep and used to compute cumulative fatigue estimates. These estimates are returned to the UI agent as the optimization signal for subsequent layout updates.

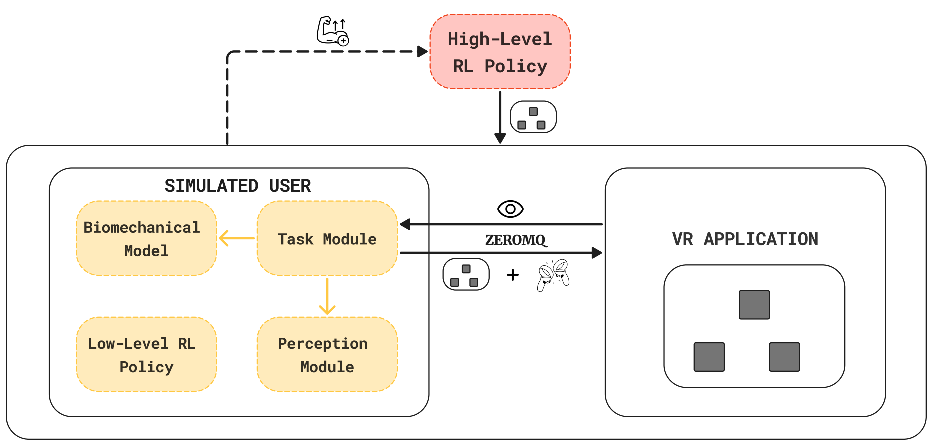



Figure 3.1: **Overview of the proposed hierarchical framework**. A high-level UI agent proposes discrete interface layouts and evaluates them using cumulative fatigue feedback from a simulated user. Each candidate layout is instantiated in the Unity-based VR application through the SIM2VR task module. Rendered observations are provided to the simulated user's perception module, which is fed into a learned control policy that outputs muscle activation signals. A low-level motion agent controls a biomechanical model to execute the interaction sequence under the proposed layout. Muscle activations are logged during execution and converted into fatigue estimates. The cumulative fatigue over the completed sequence is returned to the UI agent as a reward signal, together with the resulting layout state, to guide subsequent layout optimization.

### 3.4.2 Task Description

We define a sequential button-selection task to study repeated reaching and selection movements. A $0.64\,\mathrm{m} \times 0.36\,\mathrm{m}$ UI canvas is positioned $0.58\,\mathrm{m}$ in front of the user's head, approximating a comfortable mid-air reach for seated or standing users. The dimensions also account for the biomechanical model's lack of neck and head movement, ensuring that the UI remains within the field of view while staying within reach. Three square buttons are displayed sequentially, with the next button appearing only after the previous one has been pressed. Buttons were modeled as square targets of size $0.10\,\mathrm{m} \times 0.10\,\mathrm{m}$, consistent with prior VR fatigue and reaching studies that employ targets of comparable physical extent to elicit sustained mid-air interaction while avoiding excessive precision demands (e.g., [24]). This size reflects a common design choice in fatigue-oriented interaction tasks rather than an optimized UI parameter. Users complete the task by selecting all buttons with a VR controller. Buttons are axis-aligned, meaning their edges are parallel to the canvas axes, simplifying reach computation and ensuring consistent placement. This minimal sequential selection task isolates fatigue arising primarily from arm movements, minimizing confounding factors such as visual search or decision-making strategies.

### 3.4.3 Motion Agent

The motion agent models human behavior during the button-selection task using the upper-body, right-handed biomechanical model *MoblArmsWrist* [54], as implemented in [12]. We focus on this model because our primary interest is shoulder fatigue. *MoblArmsWrist* provides a physiologically grounded representation of arm motion, explicitly modeling muscle activations. It includes shoulder, elbow, and wrist joints, and the muscles actuating these joints, resulting in a total of 7 DOFs and 32 muscles.

#### Overview

The implementation of the motion agent follows the same underlying structure as in [12]. The agent receives an image input of the VR application as a 120×80 pixel RGB-D array along with proprioceptive state features, which are jointly encoded using a convolutional neural network (CNN).

During training, the agent is guided by a reward objective that encourages successful button selection while promoting efficient movement toward the target. The agent outputs continuous muscle control signals in the range $[-1, 1]$, which are used to actuate the biomechanical arm model. (Fig. 3.2)

Similar to SIM2VR, the motion agent is not scaled to represent a specific participant; instead, it functions as a generic, physically plausible surrogate for upper-limb motion rather than a personalized user model.

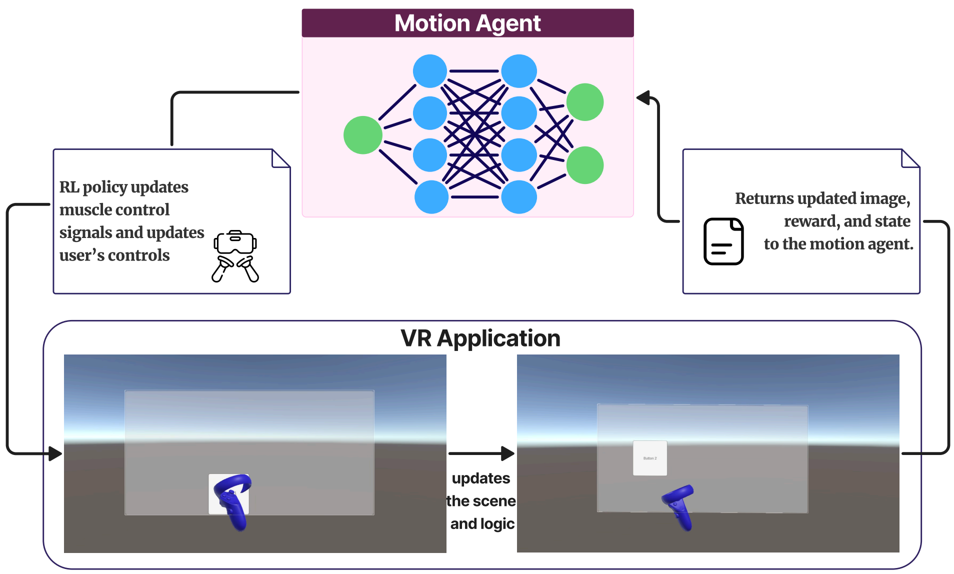



Figure 3.2: **Overview of the motion agent used to simulate interaction.** The motion agent generates muscle control signals using an RL policy and interacts with the VR application, which updates the scene and returns rendered observations, reward, and state information. Muscle activations produced during task execution are logged and later converted into fatigue estimates, which are used to evaluate interface layouts.

### Reward Design

Prior work by [56] provides guidance on reward design for biomechanical simulations. Accordingly, the reward function is structured to include three components: a task-specific reward, a distance-based term, and an effort regularization term.

- **Task-specific reward:** A sparse reward is provided only when the task is successfully completed, i.e., when a button is pressed. Both the controller and the button are equipped with colliders, and a button press is registered when the controller collider intersects the button collider within a predefined proximity. The button collider is placed at the center of the button, while the controller collider is placed on the trigger of the controller, ensuring that successful interaction requires alignment between the controller and button centers.

  During motion agent training, episodes terminate after a fixed time limit of 60 seconds, and buttons are spawned at random positions on the UI canvas. The agent is tasked with pressing as many buttons as possible within the allotted time, encouraging generalizable reaching behavior across the full interface space. The episode duration was chosen empirically to provide sufficient interaction time for learning diverse reaching behaviors without leading to excessively long training episodes.

  Each successful button press yields a reward of 5, chosen empirically to balance the task reward with shaping and effort terms.

$$r_{\text{task}} = \begin{cases} 5, & \text{if the button is pressed,} \\ 0, & \text{otherwise.} \end{cases}$$

- **Distance term:** A dense reward is computed at each timestep based on the Euclidean distance between the controller and the target ($d$). Given the fixed 0.58 m distance to the UI canvas, distances remain small, thus an exponential formulation is used to provide stronger gradients near the target and a smoother shaping signal during reaching:

$$r_{\text{d}} = e^{-\text{d}} - 1.$$

  Fig. 3.3 illustrates the shape of the distance-based shaping term used during training.

- **Effort regularization:** The effort term $\delta t$ is subtracted directly from the total reward to penalize higher physical effort, thereby discouraging unnecessary or inefficient movements [23, 12]. It represents the instantaneous effort cost $C_{\text{eff}}$ computed by the biomechanical model

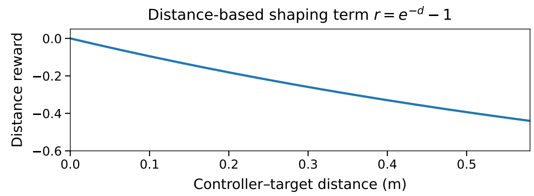



Figure 3.3: Distance-based shaping term used for reward design.

and is multiplied by a constant to scale its impact:

$$\delta t = 0.01 \cdot C_{\text{eff}}$$

where $C_{\text{eff}}$ is the instantaneous effort cost produced by the biomechanical fatigue model.

The total reward of the motion agent for each timestep is:

$$r = r_{\text{task}} + r_{\text{d}} - \delta t$$

### Training Setup

We train the motion agent using Proximal Policy Optimization (PPO), following the training configuration reported in SIM2VR [12] to ensure consistency with prior work. The policy is a multi-input actor–critic network with two fully connected layers of 256 units each and LeakyReLU activations. Continuous muscle activations are output with Tanh scaling, and the feature extractor processes both RGB-D visual input and state variables.

### Fatigue Calculation

SIM2VR provides several models for estimating effort and fatigue in biomechanical simulations, including CE [21] and variants of the 3CC model. In this work, we use the Three Compartment Controller with recovery (3CC-r) model, which explicitly accounts for both fatigue accumulation and recovery over time. The 3CC-r model maintains internal muscle states corresponding to active, resting, and fatigued capacity, and computes a scalar $C_{\text{eff}}$ based on the difference between target muscle activations and currently available capacity. In our framework, $C_{\text{eff}}$ serves as the primary fatigue signal for UI optimization, capturing the physical demand experienced during interaction. Although the model also outputs a muscle fatigue index $M_F$, prior work has shown that $M_F$ can saturate under prolonged or near-static loading [4]. Because mid-air VR interaction often involves sustained reaches and postures, we rely on $C_{\text{eff}}$ as a more sensitive measure of fatigue for layout comparison. The choice of $C_{\text{eff}}$ as the fatigue signal allows us to compare interface layouts based on predicted user fatigue without requiring absolute calibration to human fatigue levels. Its role in guiding layout optimization is described in Section 3.4.4.

#### Motion Agent Validation

The trained motion agent reliably completes the button-selection task across most of the UI canvas, making it suitable for use as a fatigue evaluator during layout optimization. Failures primarily occur when buttons are placed at the extreme left edge of the canvas, which is consistent with the reach limitations of the right-handed biomechanical model used in this work.

Since the UI agent relies on relative fatigue comparisons between candidate layouts rather than absolute task success rates, these limitations define the effective design space explored during optimization. Layouts that require repeated interaction in regions beyond the model's reachable workspace naturally incur higher predicted fatigue or task failure and are therefore disfavored by the UI agent.

### 3.4.4 UI Agent

#### Role and Problem Formulation

The UI agent operates at the interface design level and is responsible for proposing button layouts that minimize predicted user fatigue during interaction. Unlike the motion agent, which interacts with the VR environment through embodied movement and physics-based control, the UI agent interacts with the VR system by parametrically modifying the interface layout. Specifically, it proposes button placements that are instantiated in the Unity-based VR application, after which the simulated user perceives and interacts with the updated layout. The UI agent itself does not execute physical actions, but treats the simulated user as a black-box evaluator that returns fatigue estimates for a given layout. The overall UI agent workflow is illustrated in Fig. 3.4.

We formulate UI layout optimization as a discrete episodic RL problem. Each episode corresponds to evaluating a single UI layout for the fixed sequential button-selection task described in Section 3.4.2. The episode terminates once the simulated user has attempted the full button sequence, and the cumulative fatigue serves as the sole performance signal. This formulation enables automated exploration of interface design alternatives based purely on predicted physical comfort, without relying on heuristic ergonomic rules.

Although the layout space is discrete and low-dimensional, we adopt RL to handle stochastic, simulation-based evaluation in which fatigue estimates vary due to biomechanical dynamics and recovery effects. This formulation allows the UI agent to learn under noisy ergonomic feedback while incorporating soft constraints via reward penalties for infeasible layouts (e.g., unreachable or overlapping button placements), rather than explicitly restricting the action space.

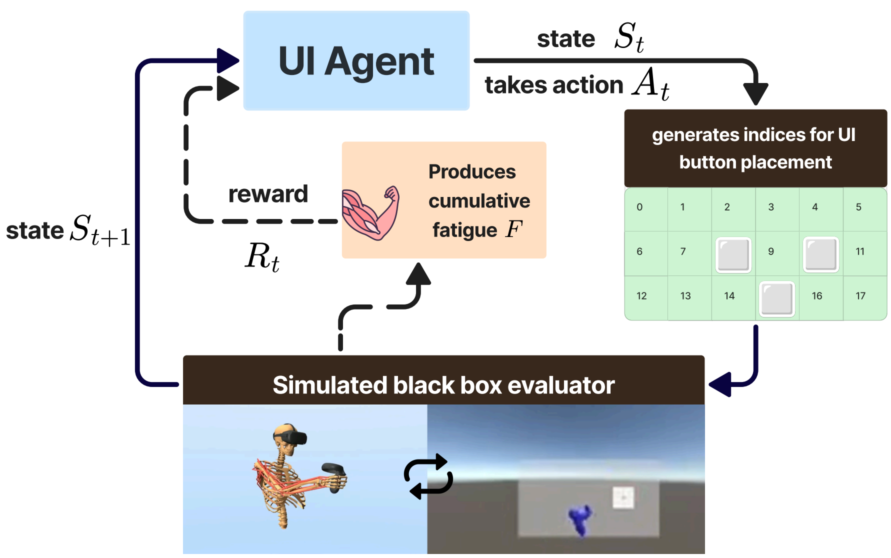



Figure 3.4: **Overview of the UI agent responsible for interface layout optimization.** The agent proposes discrete grid-based button layouts, which are instantiated in the VR environment and evaluated by the motion agent. Aggregated fatigue is returned as a reward signal, along with the next state of the UI agent defined by the grid coordinates of all button positions.

### Action Space

The UI canvas has dimensions $0.64\,\mathrm{m} \times 0.36\,\mathrm{m}$ and contains three square buttons of size $0.10\,\mathrm{m} \times 0.10\,\mathrm{m}$. To keep the optimization problem tractable while ensuring physically valid and visually distinct placements, and to reflect realistic UI layouts commonly used in VR applications, we discretize the canvas into a $3 \times 6$ grid, yielding 18 non-overlapping cells. Each cell corresponds to a valid button placement that lies fully within the canvas boundaries.

We adopt a discrete action space for two primary reasons. First, interface layout optimization is fundamentally a combinatorial design problem rather than a continuous control task. Each action represents the selection of grid-based positions for multiple UI elements, and evaluating any proposed layout requires executing a full biomechanical simulation, making reward signals both computationally expensive and stochastic. By discretizing the canvas into semantically meaningful grid cells, we restrict the search to structurally distinct configurations, thereby improving sample efficiency under costly rollouts. Second, discretization avoids degenerate solutions common in continuous optimization, such as near-overlapping buttons or infinitesimal spatial adjustments that produce negligible ergonomic differences but introduce instability in policy learning.

The UI agent's action is a discrete vector

$$\mathbf{a} = (a_1, a_2, a_3),$$

where each $a_i \in \{0, \ldots, 17\}$ specifies the grid cell index for button $i$. All three button positions are proposed simultaneously at the beginning of an episode. Layouts that assign multiple buttons to the same grid cell are penalized through the reward function, discouraging overlapping or ambiguous configurations without imposing hard constraints on the action space.

### Observation Space

The observation provided to the UI agent consists of the current layout configuration expressed as the $(x, y)$ grid coordinates of each button. These coordinates are normalized to $[0, 1]$ based on the canvas dimensions. The UI agent does not receive any direct information about the user's kinematics, task state, or workspace geometry. This deliberately minimal observation design reflects realistic interface design settings, where designers must reason about ergonomic quality without access to detailed kinematic measurements of the user, and instead rely on aggregate comfort-related feedback.

### Reward Function

The objective of the UI agent is to minimize cumulative physical effort experienced by the simulated user over the entire interaction sequence. During task execution, the motion agent produces a scalar instantaneous effort cost $C_{\text{eff}}(t)$ at each timestep using the 3CC-r fatigue model (Section 3.4.3). These values are accumulated over time to compute sequence-level fatigue. For the three-button sequential task, we define the cumulative effort associated with selecting button $i$ as

$$F_i = \sum_{t \in \mathcal{T}_i} C_{\text{eff}}(t),$$

where $\mathcal{T}_i$ denotes the set of timesteps corresponding to the reach or transition to button $i$ (from the resting posture for $i = 1$ and from the previously selected button for $i > 1$).

The episode-level reward is then defined as

$$r = -\sum_{i=1}^{3} F_i.$$

Each episode begins from a fixed initial posture, and the motion agent's internal fatigue state is reset between episodes so that rewards reflect fatigue accumulated within a single interaction sequence. This formulation directly aligns the optimization objective with ergonomic comfort: layouts that require longer reaches, awkward postures, or sustained muscle activation receive lower rewards. No additional task-completion or heuristics are included in the reward, allowing the UI agent to discover fatigue-efficient layouts autonomously.

To ensure robustness, we discourage infeasible layouts through reward penalties rather than explicitly restricting the action space. Instead of masking invalid actions, we allow the UI agent to explore the full layout space and assign large penalties to unreachable or ill-formed configurations. If the motion agent fails to press a button within the first 15 seconds of an episode, a large fatigue penalty is assigned and the episode terminates early. While episodes are already bounded by a fixed time limit, this additional early termination is introduced to avoid uninformative trajectories and improve training efficiency when the agent is unable to make meaningful progress. Similarly, layouts that assign multiple buttons to the same grid cell incur a penalty and trigger early termination before deployment in VR. This reward-based constraint formulation encourages the UI agent to autonomously learn reachable, well-formed layouts while preserving fatigue minimization as the primary optimization objective.

#### Training Setup

We train the UI agent using PPO. The policy network is a lightweight fully connected actor–critic model with two hidden layers of 128 units each and LeakyReLU activations. A custom feature extractor first normalizes the button coordinate inputs using layer normalization and then projects them into a 64-dimensional latent representation via a linear layer with LeakyReLU activation, before policy and value prediction. Training is performed until policy convergence using multiple parallel environments. The network architecture and hyperparameters were selected empirically to provide stable training, and were not extensively tuned across alternative configurations.

This setup enables the UI agent to iteratively refine layout proposals based on predicted physical effort, providing a principled mechanism for exploring fatigue-aware interface designs. Because layout proposals and fatigue estimates are evaluated at the level of individual button placements, the learned policies can be post hoc analyzed to identify spatial patterns associated with lower predicted physical demand, offering opportunities for design insight beyond black-box optimization.

## 3.5 Evaluation

Two baseline UI configurations were considered for comparison. The first is a static, hand-designed layout with all buttons positioned at the center of the UI canvas, reflecting a conventional and intuitive design. The second is a Bayesian-optimized layout, generated using Bayesian Optimization (BO) to minimize predicted user fatigue. BO treats the mapping from candidate UI layouts to predicted fatigue as a black-box objective and is commonly used for data-efficient interface adaptation when evaluations are expensive or noisy, providing a natural optimization-based baseline for this study. (Section 3.3).

BO is implemented using Ax [47]. The optimization is initialized with 15 Sobol-sampled layouts, followed by 250 BO iterations. Each candidate layout is evaluated by the motion agent, which executes the sequential button-selection task and returns the sequence-level accumulated effort

$$F = \sum_{t=1}^{T} C_{\text{eff}}(t),$$

i.e., the sum of the instantaneous effort cost $C_{\text{eff}}(t)$ produced by the 3CC-r model over all timesteps in the sequence. This value is used as feedback to the optimizer. This setup enables systematic comparison between conventional central placement and layouts discovered through simulation-driven fatigue-aware optimization.

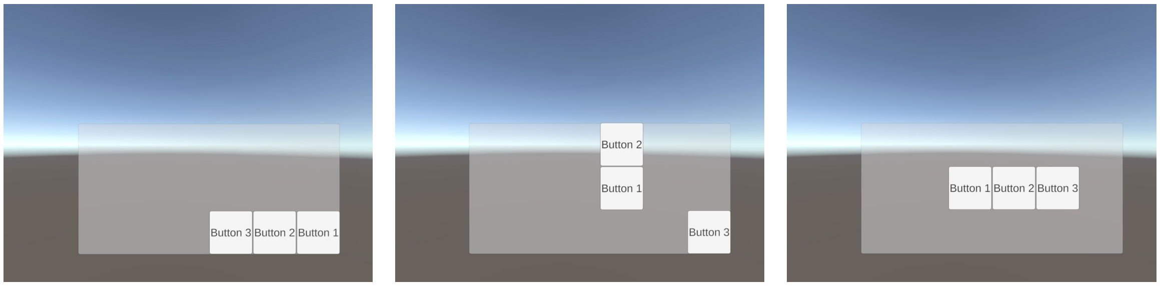
### 3.5.1 Simulation Results

We compare layouts produced by the RL agent against the static baseline and the BO-based baseline. All layouts are defined over a discrete $3 \times 6$ grid on a $0.64\,\mathrm{m} \times 0.36\,\mathrm{m}$ UI canvas. The resulting RL, BO, and static layouts are shown in Fig. 3.5.

**RL Agent** After training, the RL agent consistently selects grid cells 17, 16, and 15 for the sequential button task. These positions correspond to the rightmost section of the grid, which the biomechanical simulation predicts to yield lower sequence-level accumulated effort ($F$) over the interaction sequence. The selected layouts cluster buttons within a contiguous, reachable region of the workspace, resulting in lower $F$ across sequential button transitions.

**BO** After performing optimization, we selected the layout recommended by Ax as the best-performing configuration, based on the observed objective values across evaluations. This resulted in a layout placing buttons at grid cells 9, 3, and 17, with a mean accumulated effort of 25.26 (variance 21.09), which we used as the BO baseline in the user study.

**Static Layout** The static baseline places buttons at grid cells 8, 9, and 10, forming a centrally aligned horizontal configuration on the UI canvas. This layout reflects a common default design




(a) RL (b) BO (c) Static

Figure 3.5: **UI configurations evaluated in the study.** (a) RL-based layout optimized using biomechanical fatigue feedback, (b) Bayesian optimization (BO) layout minimizing accumulated effort, and (c) static layout with centrally placed buttons.

choice in VR interfaces, where elements are positioned near the center of the user's field of view. Because it does not account for user biomechanics or the cost of transitions between successive button selections, it serves as a naive reference for evaluating fatigue-aware layout optimization.

### 3.5.2 User Study

#### Setup

**Participants** Eighteen participants were recruited to interact with the three UI configurations. All participants were students enrolled in institutions of higher education. The sample included 11 male and 7 female participants, with a mean age of 26.5 years (SD = 2.25). All participants were right-handed, as the simulated user employed in our pipeline is based on a right-handed biomechanical model. Twelve participants reported prior experience with VR, while six had little or no prior exposure.

**Apparatus** We used a Meta Quest 2 HMD, powered by a computer running on an Intel Core 11th Gen CPU and equipped with an NVIDIA GeForce RTX 3080 Ti.

**Procedure** Each UI configuration was treated as a separate condition and consisted of 30 sequential button-interaction tasks. Participants interacted with each condition three times, resulting in a total of 90 interaction sequences per participant. The order of UI conditions was counterbalanced using a Latin square design in a within-subject experiment. Participants were given a short break of

30 seconds between interaction sequences and a longer break of one minute after completing each condition, with additional rest available upon request.

During the study, participants were seated on a chair in an office environment and were instructed to maintain an upright posture while minimizing torso movement, in order to match the fixed-torso assumption of the simulated user model. After completing each condition, participants provided subjective feedback by reporting their perceived effort using the Borg CR10 scale as well as workload ratings using the NASA Task Load Index (NASA-TLX). On average, participants required between 45 and 50 minutes to complete the study.

### 3.5.3 Results

We evaluated the three UI configurations using both simulated effort estimates and subjective fatigue reported by human participants. Data were analyzed using SPSS 31. Distributions were considered approximately normal when Skewness (S) and Kurtosis (K) were within $\pm 1$ [19, 39]; otherwise, a log transformation was applied.

#### Predicted Effort and Perceived Exertion

**Simulated User** We analyzed the accumulated effort predicted by the biomechanical model across the three UI layouts. Since the UI agent is optimized at the sequence level, we report sequence-level accumulated effort ($F$), defined as the sum of the instantaneous effort cost $C_{\text{eff}}(t)$ over all timesteps in a sequence. Each configuration was evaluated over 30 trials using identical sequences, and mean $F$ values were reported. In the simulation, muscle states were reset between sequences to isolate the effort required for a single sequence. In contrast, participants completed sequences consecutively and reported perceived exertion after each condition. Although these measures capture different timescales, layouts with lower predicted $F$ per sequence are expected to result in lower perceived exertion when repeated. A Friedman test revealed a significant effect of UI configuration on predicted accumulated effort ($\chi^2(2) = 48.26$, $p < 0.001$). Post-hoc comparisons indicated that the RL-based UI yielded significantly lower $F$ than both the Static and BO-based UIs, while the Static UI yielded lower $F$ than the BO-based UI.

| User | Fatigue Type | UI | Mean | Std. | Comparison | Wilcoxon Signed Rank | |
|---|---|---|---|---|---|---|---|
| | | | | | | Z | p |
| Simulated User | Accumulated effort ($F$) | UI_Static | 30.4513 | 4.0708 | $UI_{Static} < UI_{BO}$ | -2.910 | $< 0.01$ |
| | | UI_BO | 33.0482 | 1.3801 | $UI_{RL} < UI_{BO}$ | -4.782 | $< 0.001$ |
| | | UI_RL | 22.8955 | 0.7652 | $UI_{RL} < UI_{Static}$ | -4.782 | $< 0.001$ |
| User Study | Borg CR10 | UI_Static | 3.639 | 2.2016 | $UI_{Static} < UI_{BO}$ | -2.875 | $< 0.01$ |
| | | UI_BO | 5.94 | 1.662 | $UI_{RL} < UI_{BO}$ | -3.598 | $< 0.001$ |
| | | UI_RL | 1.889 | 1.6230 | $UI_{RL} < UI_{Static}$ | -3.210 | $= 0.001$ |

Table 3.1: Descriptive statistics and Wilcoxon signed-rank post-hoc comparisons of fatigue across UI conditions.

**User Study** For the human participants (N = 18), we applied the same non-parametric analysis to perceived exertion measured using the Borg CR10 scale. A Friedman test showed a significant effect of UI configuration on perceived exertion ($\chi^2(2) = 22.90$, $p < 0.001$). Post-hoc comparisons revealed the same ordering across UI conditions: the RL-based UI produced the lowest fatigue, followed by the Static UI, with the BO-based UI yielding the highest fatigue.

Descriptive statistics and post-hoc pairwise Wilcoxon signed-rank comparisons are summarized in Table 3.1.

Overall, the relative ordering of UI configurations was consistent across simulated effort estimates and subjective fatigue reports, suggesting alignment between the biomechanical model's predictions and human-perceived exertion.

### Task Completion Time

We measure task completion time to examine whether UI layout influences user performance. Since the next button appears only after the current button is successfully pressed, accuracy and error rates are not analyzed separately; errors are implicitly reflected as increased completion time.

A repeated-measures ANOVA was performed on log-transformed task completion time (S = 0.54, K = 0.31). Results revealed a significant effect of UI condition on completion time ($F(2, 34) = 5.92$, $p = 0.006$, $\eta_p^2 = 0.26$). Post-hoc comparisons with Bonferroni correction indicated that UI_BO resulted in significantly longer completion times than both UI_Static ($p < 0.05$) and UI_RL

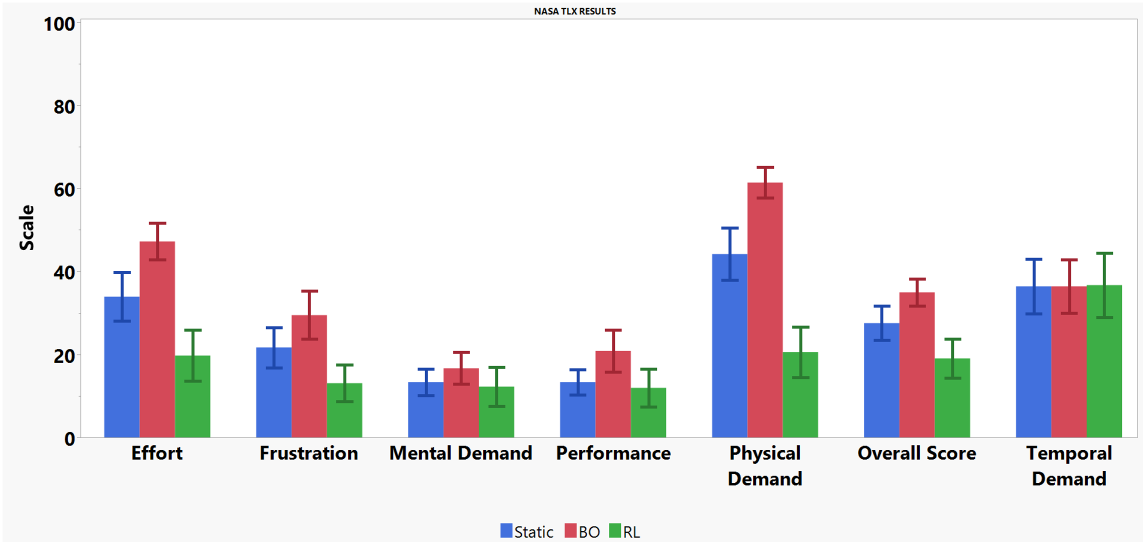



Figure 3.6: **NASA-TLX workload ratings across the three UI configurations.** Bars show mean scores for each NASA-TLX subscale, aggregated across participants. Lower values indicate lower perceived workload.

($p < 0.05$), while no significant difference was observed between UI_Static and UI_RL.

### NASA-TLX

We evaluated user workload across the three UI configurations using the NASA-TLX questionnaire (Fig. 3.6). A Friedman test on the overall NASA-TLX scores revealed a significant effect of UI configuration ($\chi^2(2) = 15.672$, $p < 0.001$).

Follow-up analyses of individual NASA-TLX subscales showed significant effects of UI configuration for *Effort* ($\chi^2(2) = 12.94$, $p < 0.01$), *Frustration* ($\chi^2(2) = 7.73$, $p < 0.05$), *Performance* ($\chi^2(2) = 9.18$, $p = 0.01$), and *Physical Demand* ($\chi^2(2) = 20.55$, $p < 0.001$).

Post-hoc pairwise comparisons were conducted using Wilcoxon signed-rank tests to identify differences between UI configurations. As shown in Table 3.2, significant pairwise differences were most consistently observed between the RL-based and BO-based UIs, particularly for the Physical Demand subscale. Comparisons between the RL-based and Static UIs yielded smaller effects and were not consistently significant across subscales, with the exception of Physical Demand, where the RL-based UI resulted in significantly lower scores.

| Subscale | Comparison | Z | p-value |
|---|---|---|---|
| Overall Scores | $UI_{RL}$–$UI_{Static}$ | -2.172 | $< 0.05$ |
| | $UI_{RL}$–$UI_{BO}$ | -2.461 | $< 0.05$ |
| | $UI_{BO}$–$UI_{Static}$ | -2.344 | $< 0.05$ |
| Effort | $UI_{RL}$–$UI_{Static}$ | -1.998 | $< 0.05$ |
| | $UI_{RL}$–$UI_{BO}$ | -2.594 | $< 0.01$ |
| | $UI_{BO}$–$UI_{Static}$ | -1.827 | n.s |
| Frustration | $UI_{RL}$–$UI_{Static}$ | -1.542 | n.s |
| | $UI_{RL}$–$UI_{BO}$ | -2.028 | ¡ 0.05 |
| | $UI_{BO}$–$UI_{Static}$ | -1.494 | n.s |
| Performance | $UI_{RL}$–$UI_{Static}$ | -0.910 | n.s |
| | $UI_{RL}$–$UI_{BO}$ | -1.724 | n.s |
| | $UI_{BO}$–$UI_{Static}$ | -2.405 | $< 0.05$ |
| Physical Demand | $UI_{RL}$–$UI_{Static}$ | -3.007 | $< 0.01$ |
| | $UI_{RL}$–$UI_{BO}$ | -3.36 | $< 0.001$ |
| | $UI_{BO}$–$UI_{Static}$ | -2.421 | $< 0.05$ |

Table 3.2: Post-hoc Wilcoxon signed-rank test results for NASA-TLX subscales.

## 3.6 Discussion

This work investigates whether biomechanical simulation can serve as a reliable driver for fatigue-aware VR UI layout optimization and why RL is particularly well-suited for this problem. While biomechanical models should not be interpreted as predictors of absolute human fatigue, they provide consistent and actionable signals for *comparative* interface evaluation. The results also highlight the limitations of layout selection based on sparse, per-layout evaluations under noisy ergonomic feedback, motivating learning-based approaches that optimize expected cumulative interaction cost rather than isolated interface elements.

### 3.6.1 Role of Biomechanical Models in UI Design

A key question is whether biomechanical user models can be meaningfully integrated into UI design. The results indicate that these models are well-suited for *relative* evaluation of interface layouts, even without detailed kinematic measurements or human-in-the-loop fatigue data. Simulated fatigue values may not correspond directly to perceived exertion for individual users, but the ordinal ranking of interface layouts was preserved consistently in the user study. This demonstrates that biomechanical simulation can function as an effective early-stage design tool, allowing designers to identify ergonomically unfavorable layouts and explore alternatives before conducting costly and time-consuming user studies. The value of the approach lies in guiding attention toward layouts that are likely to impose lower physical demand, rather than in providing precise fatigue predictions.

### 3.6.2 Limitations of Bayesian Optimization under Noisy Fatigue Signals

Despite explicitly optimizing predicted fatigue, the BO baseline underperformed both the RL-based layout and the heuristic central placement. This outcome can be attributed to the stochastic characteristics of the fatigue signal: even under fixed interaction sequences, predicted fatigue exhibits variability due to movement stochasticity and recovery dynamics in the biomechanical model. While individual measurements are noisy, the relative ordering of layouts remains stable across repeated evaluations. BO relies on a limited number of noisy evaluations to infer improvements, making it difficult to distinguish weak but systematic ergonomic advantages.

In contrast, the RL agent optimized expected cumulative fatigue through repeated rollouts, effectively learning from distributions of outcomes rather than individual evaluations. This enabled the agent to exploit consistent ergonomic patterns such as reduced cumulative reach and compact spatial arrangements, even under stochastic per-episode feedback. The static layout benefited from a robust central-placement heuristic, while the RL agent refined beyond this baseline by explicitly optimizing cumulative interaction cost.

### 3.6.3 Implications for VR UI Design

The learned layouts provide concrete insights for fatigue-aware VR interface design. The RL agent consistently positioned all three buttons in the lower region of the canvas (cells 15–17), resulting in lower predicted and perceived fatigue compared to the centrally clustered static baseline (cells 8–10) and the more dispersed BO layout (cells 3, 9, and 17). From a biomechanical perspective, these placements reduce sustained shoulder elevation and upper-arm abduction across successive interactions, lowering activation of fatigue-prone muscle groups. This demonstrates that cumulative arm effort can be meaningfully reduced by leveraging row-level placement preferences rather than relying solely on small local adjustments around a central position. Importantly, these fatigue reductions were achieved without increasing task completion time, suggesting that ergonomic improvements can coexist with interaction efficiency.

The framework is designed to evaluate, compare, and refine layouts under task and sequence-specific constraints, rather than to discover a single optimal layout. Observed fatigue reductions emerge from cumulative interaction effects rather than isolated reach distances, making them difficult to anticipate through manual design alone. Training the motion and UI agents incurs upfront computation, but this cost is amortized across multiple design iterations, enabling rapid exploration of layout alternatives without repeated user studies, which are logistically more expensive and limited in coverage.

Finally, the optimization can incorporate designer-imposed constraints such as restricted regions, symmetry, accessibility requirements, or aesthetic guidelines, allowing refinement within realistic design boundaries. This demonstrates that biomechanical simulation can support iterative, informed UI design while respecting practical constraints.

## 3.7 Application: Extending to Longer Sequential Tasks

The results in Section 3.5 suggest that biomechanical simulation provides a reliable *comparative* signal for evaluating interface layouts, and that RL is particularly effective when ergonomic differences emerge through cumulative interaction effects rather than isolated actions. To investigate how our framework scales to more complex interactions, the three-button setup is extended to longer sequential tasks involving multiple interface elements. This setup enables evaluation of layouts under more realistic interaction patterns, where repeated selections and variable button usage affect cumulative fatigue.

### 3.7.1 Task Definition

The interface consists of five buttons displayed simultaneously on the UI canvas. For a given layout, the simulated user executes three button-selection sequences sampled from a predefined sequence generator, resulting in a total of nine button presses per episode. The layout remains fixed throughout each episode to reflect realistic interface behavior, as dynamically repositioning elements between sequences would be disruptive to users.

#### Optimization Objective

At a higher level, optimizing button placement for high-frequency usage might initially seem like a simple assignment problem. However, fatigue is cumulative and path-dependent, making it more complex than a static spatial optimization. By framing this as an RL problem, the interface agent is enabled to adapt to frequency-weighted fatigue patterns over time. In sequential interaction tasks, repeated reaching motions lead to cumulative fatigue, and elements that are used more frequently contribute significantly more to the total physical effort than less frequently used ones. The goal is to optimize interface layouts not only for spatial efficiency but also for frequency-weighted fatigue. To achieve this, the empirical button usage is tracked during each episode and incorporated into the interface agent's reward structure. This ensures that high-frequency buttons are placed in regions that minimize physical effort, extending the optimization beyond simple spatial considerations, as seen in the three-button setup.

### 3.7.2 RL Approach

#### Motion Agent

To train the motion agent for multiple buttons simultaneously, we propose the following approach. At each step of the sequence, a small target sphere is placed at the center of the currently active button. The agent is trained to reach and select this target. After each successful interaction, the target advances to the next button in the sequence, enabling multi-step trajectories while preserving the original reward structure and motion dynamics. Following the same reward structure as in Section 3.4.3, the reward includes a task component $r_{\text{task}}$, a distance component $r_{\text{d}}$, and an effort regularization term $\delta t$. A reward of 5 is given for pressing a button, and an additional reward of 15 is provided for completing the sequence. The distance reward $r_{\text{d}}$ is computed as $e^{-\text{d}} - 1$, where distance is measured between the controller and the center sphere. The effort regularization term remains unchanged.

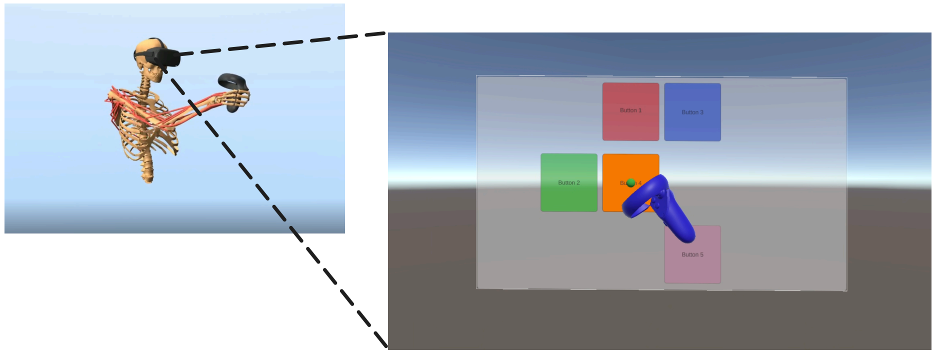


Figure 3.7: **Sequential five-button interaction task.** All five buttons are displayed simultaneously, and the agent performs selections in the order specified by the visual marker.

**Sequential Interaction Modeling in VR** In the original single-button task, reward shaping based on Euclidean distance was sufficient because only one target was visible at a time. When extending the task to multiple simultaneously visible buttons, distance-based rewards no longer reliably indicate which button in the sequence should be selected next, leading to ambiguous credit assignment. This issue is amplified by image-based observations, where multiple potential targets appear in the visual input without an explicit indication of priority. Motivated by [20], we introduce an explicit target representation as a small sphere placed at the center of the currently active button. Unlike [20], where the agent receives a low-dimensional state vector explicitly encoding target location and end-effector position, our motion agent operates on image-based observations and outputs muscle-level control signals, making target identity non-trivial to infer from the input alone. Alternative strategies, including directly embedding button positions or visually differentiating buttons, did not reliably disambiguate the active target. The explicit spatial target, therefore, provides a clear, visually grounded goal for sequential interactions while preserving the original reward structure (Fig. 3.7).

**Interface Agent**

The episode is defined as follows: At each timestep, the interface agent takes an action. For the same action (i.e., the same layout), three button sequences are executed, leading to nine button interactions. This design ensures that the layout does not change every timestep, avoiding confusion caused by dynamic layout changes.

Over an episode, we record the empirical usage frequency of each button and the average fatigue cost incurred per button press. The observation space is defined as the concatenation of button usage frequencies and their current grid positions. The action space corresponds to assigning each of the five buttons to a unique cell in the discretized UI grid.

Let $n_i^{(t-1)}$ denote the number of times button $i$ was selected in the previous episode, and define the empirical usage frequency as

$$\pi_i^{(t-1)} = \frac{n_i^{(t-1)}}{\sum_{j=1}^{5} n_j^{(t-1)}}.$$

Empirical usage frequencies from the previous episode $\pi_i^{(t-1)}$ are used rather than those from the current rollout. Because usage statistics are only available after the episode completes, computing the reward with $\pi_i^{(t)}$ would allow the layout to influence the frequency weights used to evaluate itself, introducing circularity into the reward signal. Using $\pi_i^{(t-1)}$ instead provides a fixed estimate of expected interaction frequencies, ensuring that layout evaluation remains independent of the rollout used to measure fatigue. This design stabilizes training and avoids biased credit assignment.

Let $F_i^{(t)}$ denote the cumulative effort incurred for button $i$ during the current episode, computed as the sum of instantaneous effort costs $C_{\text{eff}}(t)$ over all transitions to that button.

The reward at episode $t$ is then defined as

$$R_t = e^{-\sum_{i=1}^{5} \pi_i^{(t-1)} F_i^{(t)}}.$$

An exponential scaling is applied to increase the sensitivity of the reward to fatigue incurred by frequently used buttons. This ensures that layouts placing high-frequency buttons in moderately high-cost regions are penalized more strongly, providing a clearer optimization signal for the UI agent. In contrast to the previous three-button setup, where linear scaling of cost was sufficient, the current design involves nine button interactions per episode. Using a linear combination of


Figure 3.8: **Five-button layout optimized under frequency-weighted fatigue.** Frequently used buttons are placed in lower-cost regions of the canvas, reflecting the agent's adaptation to cumulative biomechanical effort across sequential interactions.

frequency and cost would produce relatively small differences in reward values, which could hinder PPO's learning by generating weak gradient signals. The exponential formulation amplifies these differences, allowing the agent to more effectively distinguish between layouts and accelerate convergence during training.

To avoid overlapping buttons, a large penalty is applied whenever multiple buttons occupy the same grid cell. This ensures that invalid UI placements are clearly discouraged.

### Results

Across episodes, the interface agent consistently converged to a single layout configuration, even when button usage frequencies varied (Fig. 3.8). This behavior is a consequence of the optimization formulation rather than a limitation of frequency-aware learning. In the reward formulation, each button's fatigue cost is scaled by how often it is used, so frequently pressed buttons contribute more to the overall penalty. However, the relative difficulty of positions on the canvas remains largely stable, meaning that frequency weighting amplifies penalties but does not change which locations are easiest or hardest to reach. As a result, the agent learns a layout that minimizes expected fatigue across all episodes, rather than adjusting button placements to per-episode usage fluctuations. This illustrates that policy-based optimization finds a globally efficient layout under the expected interaction statistics, highlighting the value of incorporating frequency into the fatigue signal. This demonstrates that the framework proposed produces ergonomically sensible layouts at scale and validates the utility of frequency-weighted fatigue as a design signal, even when per-episode variations are small.

## 3.8 Limitations and Future Work

This extension to longer sequential tasks illustrates the role of cumulative fatigue as an effective optimization signal and shows that the proposed framework remains well-behaved under more dynamic and realistic interaction scenarios. The focus of this analysis is on scalability and optimization behavior as task complexity increases.

The results should therefore be interpreted at the level of the simulated model. While the predicted fatigue signals were effective for comparative evaluation of interface layouts, they are not intended to serve as absolute predictors of perceived exertion for specific users. Biomechanical models provide a reasonable baseline by mimicking human movement strategies, but they lack cognitive processes and decision-making, and cannot fully capture adaptive or nuanced human behaviors. In addition, individual differences in anthropometry, posture, strength, and movement strategy were not explicitly modeled. This abstraction enables controlled analysis of physical effort, but real-world VR fatigue may also be influenced by cognitive load, visual strain, or prior physical activity—factors not captured by the current simulation.

Future work could address these limitations by introducing variability in anthropometric parameters, such as sampling different arm lengths during execution, to better approximate user diversity. In this work, we focused on a right-handed upper-body model; extending the framework to full-body or bimanual biomechanical models would allow investigation of fatigue patterns in larger or more complex interfaces involving both controllers [3, 61]. Such extensions would also support the study of multi-step or multi-task interaction scenarios in which fatigue accumulates across broader task contexts.

From a systems perspective, the architecture follows a hierarchical RL structure in which the UI agent depends on a pretrained biomechanical motion agent. While the motion agent generalizes to moderate variations in the VR task (e.g., button size or position), its ability to adapt to more substantial changes in interaction mechanics or interface design remains limited. Future work could explore curriculum learning, structured randomization of task and interface parameters (e.g., target size, position, and interaction dynamics), or fine-tuning strategies to improve generalization across interface variations without requiring full retraining.

Finally, while we demonstrated the framework using PPO, the proposed formulation is not tied to a specific learning algorithm. Alternative optimization methods or hybrid approaches could be explored depending on design constraints and interaction context. The framework could also be extended to support more complex or adaptive UI elements, such as dynamically adjusting layouts, variable button sizes, or compound widgets (e.g., drop-down menus), enabling investigation of scalability in more flexible and user-responsive interface designs.

## 3.9 Conclusion

We present a framework that systematically integrates biomechanical simulation into the VR UI design process to support fatigue-aware layout optimization. Although simulated fatigue should not be treated as an absolute predictor of human effort, our results demonstrate that biomechanical models provide consistent *comparative* signals for evaluating and refining interface layouts. By framing layout selection as a sequence-level optimization problem under noisy ergonomic feedback, we show that RL is well-suited to optimizing layouts under cumulative fatigue signals that are difficult to address with static or single-shot approaches. Rather than identifying a single universally optimal layout, the framework enables early-stage exploration of ergonomically informed design alternatives, reducing reliance on costly user studies and facilitating the development of more physically considerate VR interfaces.


## Acknowledgement

This research was undertaken, in part, based on support from the Natural Sciences and Engineering Research Council of Canada Grant RGPIN-2021-03479 (NSERC DG).

# Chapter 4

# Conclusion and Future Work

## 4.1 Contributions

This thesis presents a simulation-driven framework for fatigue-aware VR interface design that leverages biomechanical user models as surrogate evaluators of physical effort. By formulating interface layout selection as a sequence-level optimization problem under cumulative fatigue, the framework enables ergonomic considerations to be incorporated directly into the design process without relying on hand-crafted heuristics or extensive human data collection.

Rather than identifying a single optimal layout, the approach supports systematic exploration of design alternatives using physically grounded cost signals. The results demonstrate that relative fatigue trends predicted by biomechanical simulation align with human-reported exertion, highlighting the potential of simulated users as effective early-stage design tools.

## 4.2 Limitations

Despite the thesis's contribution towards a framework that demonstrates the potential of biomechanical simulation for fatigue-aware interface design, it has limitations.

Although the simulation provides meaningful relative comparisons between interface layouts, it does not fully capture subjective or user-specific perceptions of fatigue. Factors such as cognitive load, visual strain, and prior physical state are not modeled.

Furthermore, the current framework focuses on simplified interaction tasks and a right-handed upper-body model. Individual differences in anthropometry, strength, and movement strategy are not explicitly considered. While this enables controlled analysis, it limits the direct applicability of the findings to more complex interaction scenarios and diverse user populations.

## 4.3 Future Work

Future work can extend the proposed framework to more complex and realistic VR interfaces beyond grid-based button layouts. This includes structured interface designs such as panels, toolbars, and hierarchical menus, where interaction cost depends not only on spatial placement but also on navigation depth, interface transitions, and task structure.

Another direction is to incorporate greater variability in user characteristics. Extending the framework to left-handed and bimanual models [3], as well as introducing variation in anthropometric parameters such as arm length and reach, would enable analysis across more diverse user populations. Together, these directions aim to improve the generalizability and applicability of fatigue-aware interface optimization in real-world VR systems.

# Appendix A

# Hyperparameters for the RL Agents

## A.1 UI Agent

| **Parameter** | **Value** |
|---|---|
| Algorithm | PPO |
| Policy network | 2 layers, 128 units, LeakyReLU |
| Learning rate schedule | Linear $10^{-4} \rightarrow 10^{-6}$ |
| Discount factor $\gamma$ | 0.0 |
| GAE parameter $\lambda$ | 0.0 |
| Total timesteps | $10^6$ |
| Parallel workers | 12 |
| Rollout steps per update | 320 |
| Batch size | 960 |
| Target KL | 0.05 |

Table A.1: Training hyperparameters for the UI agent.

## A.2 Motion Agent

| Parameter | Value |
|---|---|
| Algorithm | PPO |
| Policy network | 2 layers, 256 units, LeakyReLU |
| Action activation | Tanh |
| Learning rate schedule | Linear $5 \times 10^{-5} \rightarrow 1 \times 10^{-7}$ |
| Total timesteps | $10^8$ |
| Parallel workers | 10 |
| Rollout steps per update | 4000 |
| Batch size | 1000 |
| Target KL | 1.0 |

Table A.2: Training hyperparameters for the motion agent.

### A.2.1 Motion Agent Pipeline

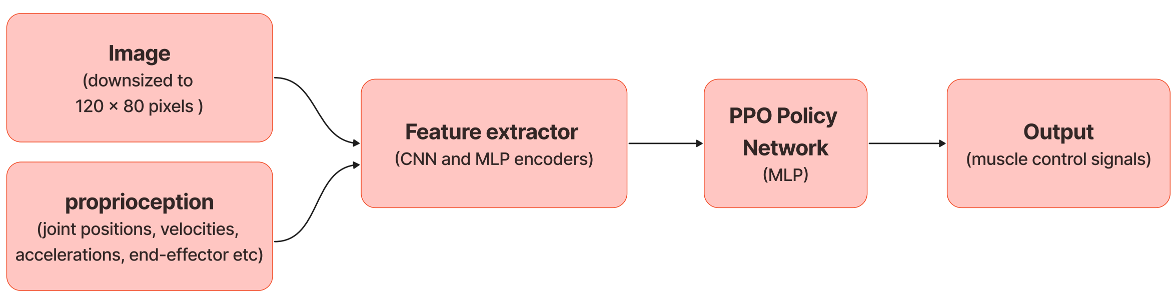



Figure A.1: **Motion agent pipeline.** The agent receives multimodal observations consisting of visual input and proprioceptive features. These inputs are then processed through a feature extractor composed of convolutional and fully connected encoders, producing a latent representation used by a PPO-based policy network. The policy outputs continuous muscle control signals, which are applied to the biomechanical model, creating motion.

# Appendix B

# User Study: Designing Fatigue Aware VR Interfaces via Biomechanical Models

## B.1 Ethics Approval Form

UNIVERSITÉ Concordia UNIVERSITY

CERTIFICATION OF ETHICAL ACCEPTABILITY
FOR RESEARCH INVOLVING HUMAN SUBJECTS

| | |
|---|---|
| Name of Applicant: | Harshitha Voleti |
| Department: | Gina Cody School of Engineering and Computer Science\Computer Science and Software Engineering |
| Agency: | N/A |
| Title of Project: | Optimizing User Interfaces in VR Applications Using Reinforcement Learning |
| Certification Number: | 30021562 |

Valid From: March 12, 2025 To: March 11, 2026

The members of the University Human Research Ethics Committee have examined the application for a grant to support the above-named project, and consider the experimental procedures, as outlined by the applicant, to be acceptable on ethical grounds for research involving human subjects.

Richard DeMont

Dr. Richard DeMont, Chair, University Human Research Ethics Committee

## B.2 Borg CR10 Rating

| Score | Definition | Note |
|---|---|---|
| 0 | Nothing at all | No arm fatigue |
| 0.5 | Very, very weak | Just noticeable |
| 1 | Very weak | As taking a short walk |
| 2 | Weak | Light |
| 3 | Moderate | Somewhat but not hard to go on |
| 4 | Somewhat heavy | |
| 5 | Heavy | Tiring, not terribly hard to go on |
| 6 | | |
| 7 | Very strong | Strenuous |
| 8 | | |
| 9 | | |
| 10 | Extremely strong | Extremely strenuous |

Table B.1: Borg CR10 Scale for Perceived Exertion

# Bibliography


[1] M. Bachynskyi, A. Oulasvirta, G. Palmas, and T. Weinkauf. Is motion capture-based biomechanical simulation valid for hci studies? study and implications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3215–3224, 2014.

[2] M. Bachynskyi, G. Palmas, A. Oulasvirta, and T. Weinkauf. Informing the design of novel input methods with muscle coactivation clustering. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 21(6):1–25, 2015.

[3] V. Caggiano, H. Wang, G. Durandau, M. Sartori, and V. Kumar. Myosuite–a contact-rich simulation suite for musculoskeletal motor control. *arXiv preprint arXiv:2205.13600*, 2022.

[4] N. Cheema, L. A. Frey-Law, K. Naderi, J. Lehtinen, P. Slusallek, and P. Hämäläinen. Predicting mid-air interaction movements and fatigue using deep reinforcement learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[5] Y. F. Cheng, C. Gebhardt, and C. Holz. Interactionadapt: Interaction-driven workspace adaptation for situated virtual reality environments. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–14, 2023.

[6] S. L. Delp, F. C. Anderson, A. S. Arnold, P. Loan, A. Habib, C. T. John, E. Guendelman, and D. G. Thelen. Opensim: open-source software to create and analyze dynamic simulations of movement. *IEEE Transactions on Biomedical Engineering*, 54(11):1940–1950, 2007.

[7] P. Duan, C. Wierzynski, and L. Nachman. Optimizing user interface layouts via gradient descent. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2020.

[8] J. J. Dudley, J. T. Jacques, and P. O. Kristensson. Crowdsourcing interface feature design with bayesian optimization. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2019.

[9] J. M. Evangelista Belo, A. M. Feit, T. Feuchtner, and K. Grønbæk. Xrgonomics: Facilitating the creation of ergonomic 3d interfaces. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–11, 2021.

[10] J. M. Evangelista Belo, M. N. Lystbæk, A. M. Feit, K. Pfeuffer, P. Kán, A. Oulasvirta, and K. Grønbæk. Auit–the adaptive user interfaces toolkit for designing xr applications. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, pages 1–16, 2022.

[11] F. Fischer, M. Bachinski, M. Klar, A. Fleig, and J. Müller. Reinforcement learning control of a biomechanical model of the upper extremity. *Scientific Reports*, 11(1):14445, 2021.

[12] F. Fischer, A. Ikkala, M. Klar, A. Fleig, M. Bachinski, R. Murray-Smith, P. Hämäläinen, A. Oulasvirta, and J. Müller. Sim2vr: Towards automated biomechanical testing in vr. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, pages 1–15, 2024.

[13] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6):381, 1954.

[14] K. Gajos and D. S. Weld. Supple: automatically generating user interfaces. In *Proceedings of the 9th International Conference on Intelligent User Interfaces*, pages 93–100, 2004.

[15] D. Gaspar-Figueiredo, M. Fernández-Diego, R. Nuredini, S. Abrahão, and E. Insfrán. Reinforcement learning-based framework for the intelligent adaptation of user interfaces. In

*Companion Proceedings of the 16th ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, pages 40–48, 2024.

[16] C. Gebhardt, B. Hecox, B. Van Opheusden, D. Wigdor, J. Hillis, O. Hilliges, and H. Benko. Learning cooperative personalized policies from gaze data. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, pages 197–208, 2019.

[17] C. Gobert, K. Todi, G. Bailly, and A. Oulasvirta. Sam: a modular framework for self-adapting web menus. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*, pages 481–484, 2019.

[18] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. Pmlr, 2018.

[19] J. F. Hair, W. C. Black, B. J. Babin, R. E. Anderson, R. L. Tatham, et al. Multivariate data analysis, 2019.

[20] L. Hetzel, J. Dudley, A. M. Feit, and P. O. Kristensson. Complex interaction as emergent behaviour: Simulating mid-air virtual keyboard typing using reinforcement learning. *IEEE Transactions on Visualization and Computer Graphics*, 27(11):4140–4149, 2021.

[21] J. D. Hincapié-Ramos, X. Guo, P. Moghadasian, and P. Irani. Consumed endurance: a metric to quantify arm fatigue of mid-air interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1063–1072, 2014.

[22] J. Hussain, A. Ul Hassan, H. S. Muhammad Bilal, R. Ali, M. Afzal, S. Hussain, J. Bang, O. Banos, and S. Lee. Model-based adaptive user interface based on context and user experience evaluation. *Journal on Multimodal User Interfaces*, 12(1):1–16, 2018.

[23] A. Ikkala, F. Fischer, M. Klar, M. Bachinski, A. Fleig, A. Howes, P. Hämäläinen, J. Müller, R. Murray-Smith, and A. Oulasvirta. Breathing life into biomechanical user models. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, pages 1–14, 2022.

[24] S. Jang, W. Stuerzlinger, S. Ambike, and K. Ramani. Modeling cumulative arm fatigue in mid-air interaction based on perceived exertion and kinetics of arm motion. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 3328–3339, 2017.

[25] Y. Jiang, T. Van Wouwe, F. De Groote, and C. K. Liu. Synthesis of biologically realistic human motion using joint torque actuation. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[26] I. Kangas, M. Schwoerer, and L. Bernardi. Scalable user interface optimization using combinatorial bandits. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3375–3379, 2022.

[27] Ł. Kidziński, S. P. Mohanty, C. F. Ong, J. L. Hicks, S. F. Carroll, S. Levine, M. Salathé, and S. L. Delp. Learning to run challenge: Synthesizing physiologically accurate motion using deep reinforcement learning. In *The NIPS'17 Competition: Building Intelligent Systems*, pages 101–120. Springer, 2018.

[28] Y. Koyama, D. Sakamoto, and T. Igarashi. Crowd-powered parameter analysis for visual design exploration. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, pages 65–74, 2014.

[29] F. Lacquaniti, C. Terzuolo, and P. Viviani. The law relating the kinematic and figural aspects of drawing movements. *Acta Psychologica*, 54(1-3):115–130, 1983.

[30] T. Langerak, S. Christen, M. Albaba, C. Gebhardt, C. Holz, and O. Hilliges. Marlui: multi-agent reinforcement learning for adaptive point-and-click uis. *Proceedings of the ACM on Human-Computer Interaction*, 8(EICS):1–27, 2024.

[31] P. Langley. Machine learning for adaptive user interfaces. In *Annual Conference on Artificial Intelligence*, pages 53–62. Springer, 1997.

[32] S. M. LaValle. *Virtual reality*. Cambridge University Press, 2023.

[33] J. J. LaViola Jr, E. Kruijff, R. P. McMahan, D. Bowman, and I. P. Poupyrev. *3D user interfaces: theory and practice*. Addison-Wesley Professional, 2017.

[34] Y. Li, F. Fischer, T. Dwyer, B. Ens, R. Crowther, P. O. Kristensson, and B. Tag. Alphapig: The nicest way to prolong interactive gestures in extended reality. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2025.

[35] Y. Li, B. Tag, S. Dai, R. Crowther, T. Dwyer, P. Irani, and B. Ens. Nicer: A new and improved consumed endurance and recovery metric to quantify muscle fatigue of mid-air interactions. *ACM Transactions on Graphics (TOG)*, 43(4):1–14, 2024.

[36] D. Lindlbauer, A. M. Feit, and O. Hilliges. Context-aware online adaptation of mixed reality interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, pages 147–160, 2019.

[37] J. D. Lomas, J. Forlizzi, N. Poonwala, N. Patel, S. Shodhan, K. Patel, K. Koedinger, and E. Brunskill. Interface design optimization as a multi-armed bandit problem. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 4142–4153, 2016.

[38] J. M. Looft, N. Herkert, and L. Frey-Law. Modification of a three-compartment muscle fatigue model to predict peak torque decline during intermittent tasks. *Journal of Biomechanics*, 77:16–25, 2018.

[39] P. Mallery and D. George. Spss for windows step by step, 2000.

[40] N. Mezhoudi. User interface adaptation based on user feedback and machine learning. In *Proceedings of the Companion Publication of the 2013 International Conference on Intelligent User Interfaces Companion*, pages 25–28, 2013.

[41] M. P. Miazga and P. Ebel. Increasing interaction fidelity: Training routines for biomechanical models in hci. In *Adjunct Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*, pages 1–3, 2025.

[42] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[43] R. A. Montano Murillo, S. Subramanian, and D. Martinez Plasencia. Erg-o: Ergonomic optimization of immersive virtual environments. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 759–771, 2017.

[44] H.-S. Moon, Y.-C. Liao, C. Li, B. Lee, and A. Oulasvirta. Real-time 3d target inference via biomechanical simulation. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pages 1–18, 2024.

[45] I. Mordatch, E. Todorov, and Z. Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (ToG)*, 31(4):1–8, 2012.

[46] G. Mori, F. Paterno, and C. Santoro. Design and development of multidevice user interfaces through multiple logical descriptions. *IEEE Transactions on Software Engineering*, 30(8):507–520, 2004.

[47] M. Olson, E. Santorella, L. C. Tiao, S. Cakmak, M. Garrard, S. Daulton, Z. J. Lin, S. Ament, B. Beckerman, E. Onofrey, et al. Ax: A platform for adaptive experimentation. In *AutoML 2025 ABCD Track*, 2025.

[48] V. Pejovic and M. Musolesi. Interruptme: designing intelligent prompting mechanisms for pervasive applications. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 897–908, 2014.

[49] J. C. Quiroz, S. J. Louis, A. Shankar, and S. M. Dascalu. Interactive genetic algorithms for user interface design. In *2007 IEEE Congress on Evolutionary Computation*, pages 1366–1373. IEEE, 2007.

[50] N. Rathnayake, D. Meedeniya, I. Perera, and A. Welivita. A framework for adaptive user interface generation based on user behavioural patterns. In *2019 Moratuwa Engineering Research Conference (MERCon)*, pages 698–703. IEEE, 2019.

[51] W. Rohmert. Ermittlung von erholungspausen für statische arbeit des menschen. *Internationale Zeitschrift für angewandte Physiologie einschließlich Arbeitsphysiologie*, 18(2):123–164, 1960.

[52] P. Salem. User interface optimization using genetic programming with an application to landing pages. *Proceedings of the ACM on Human-Computer Interaction*, 1(EICS):1–17, 2017.

[53] S. Sarcar, J. P. Jokinen, A. Oulasvirta, Z. Wang, C. Silpasuwanchai, and X. Ren. Ability-based optimization of touchscreen interactions. *IEEE Pervasive Computing*, 17(1):15–26, 2018.

[54] K. R. Saul, X. Hu, C. M. Goehler, M. E. Vidt, M. Daly, A. Velisar, and W. M. Murray. Benchmarking of dynamic simulation predictions in two software platforms using an upper limb musculoskeletal model. *Computer Methods in Biomechanics and Biomedical Engineering*, 18(13):1445–1458, 2015.

[55] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[56] H. Selder, F. Fischer, P. O. Kristensson, and A. Fleig. Demystifying reward design in reinforcement learning for upper extremity interaction: Practical guidelines for biomechanical simulations in hci. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*, pages 1–17, 2025.

[57] H. Soh, S. Sanner, M. White, and G. Jamieson. Deep sequential recommendation for personalized adaptive user interfaces. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, pages 589–593, 2017.

[58] Y. Song, C. Gebhardt, Y.-C. Liao, and C. Holz. Preference-guided multi-objective ui adaptation. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*, pages 1–13, 2025.

[59] K. Todi, G. Bailly, L. Leiva, and A. Oulasvirta. Adapting user interfaces with model-based reinforcement learning. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2021.

[60] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[61] H. Wang, V. Caggiano, G. Durandau, M. Sartori, and V. Kumar. Myosim: Fast and physiologically realistic mujoco models for musculoskeletal and exoskeletal studies. In *2022 International Conference on Robotics and Automation*, pages 8104–8111. IEEE, 2022.

[62] C. J. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.

[63] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.

[64] T. Xia and L. A. F. Law. A theoretical approach for modeling peripheral muscle fatigue and recovery. *Journal of Biomechanics*, 41(14):3046–3052, 2008.