VIRTUAL REALITY: THEORY AND PRACTICE

LESSON 3: INTERACTIONS AND USER INTERFACE IN VR

GIUSEPPE TURINI - FRI 16 FEB 2024 - UNIVERSITY OF PISA

TABLE OF CONTENTS

PART 1: INTRODUCTION TO HCI

Human-computer interaction (HCI) is *"...a discipline that is concerned with the design, evaluation, and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them..."*

HEWETT ET AL. ACM SIGCHI CURRICULA FOR HUMAN-COMPUTER INTERACTION. 2014.

HCI starting from the **understanding of behaviors, needs, and preferences of users**
aims to **design interfaces and interactions that are intuitive, efficient, safe, and enjoyable**
by **considering factors such as usability, accessibility, and user experience**
with the goals of enhancing human capabilities.

BASICS: AN HCI EXAMPLE

See: www.youtube.com/watch?v=N0rApIB27x4

BASICS: SIGNIFICANCE OF HCI

HCI is always present in our everyday life in the modern world: from withdrawing cash using an ATM, to buying a snack using a food dispenser.

So, HCI is important because all devices should have easy-to-use and effective interfaces.

HCI is important to create products that are more successful, safe, functional, etc.

So, HCI should be involved at all stages of design/development of any product.

HCI is important to design/develop user-friendly products that anyone can use.

So, HCI integrates common sense and human/world understanding into products.

HCI is important to use familiar concepts to design easy-to-use products.

So, HCI uses metaphors related to user experiences (folders, books, typewriters, etc.).

BASICS: HCI GLOSSARY

User Interaction: A user interaction is how the user acts on a device (and viceversa).

UI: A user interface (UI) is the space where interactions user-device occur.

UX: The user experience (UX) is how a user interacts with and experiences a product/service (including: utility, ease of use, efficiency, etc.).

HCI: A human-computer interface (HCI) is a UI that allows a human (user) to control a computer (device). In general, a machine is designed to apply some sort of mechanical force, whereas a computer is designed to process and store data; however, both require input from a human using the proper UI to perform the computer functions.

HMI: A human-machine interface (HMI) is a UI that allows a human (user) to control a machine (device). Depending on the application field, an HMI could include different software modules and physical controls, and its complexity could vary: from simple HMIs including levers and buttons, to complex HMIs providing voice/touch controls.

BASICS: HCI ELEMENTS AND PRINCIPLES

HCI Main Elements: HCI includes 3 main elements:

- The users (cognitive capabilities, experiences, etc.).
- The tools (computer, mobile phone, website, etc.).
- The context (environment, limitations, etc.).

HCI Principles: HCI is based on 7 principles:

- Use both knowledge in the world and knowledge in the hand.
- Simplify the structure of tasks.
- Make everything clearly visible.
- Make the mapping correct.
- Convert limitations into benefits.
- Design for errors.
- Standardize if everything else fails.

BASICS: HCI DESIGN PROCESS

HCI Design Process:

The HCI design process includes 3 general phases:

- **Research:** Before design/development starts, target audience and their needs (and related issues) must be studied. The results of this user investigation direct the HCI design process.
- **Prototyping:** Providing answers to the needs/issues resulted in the previous stage. This phase includes both physical design (materials, colors, interactions) and conceptual design (how a device operates).
- **Review:** After the finished product is ready, its design must undergo an evaluation involving experts and users to ensure that it complies with specifications and HCI design principles (this could lead to a re-design).

BASICS: HISTORY OF HCI

This is a brief history of human-computer interaction:

- 1941-44: early computing machines (**research-grade computers**).
- 1963-64: prototypes of point-and-click interfaces (sketchpad, mouse).

At this time, HCI was: function-centered, for specialists (not consumers), mainly mechanical.

- 1983-90: desktop computers and OS (**consumer-grade computers**).
- 1990: hypertexts (**Internet**, websites, webpages).

Before 2000, HCI was: function-centered, with few/no visuals, and with steep learning curve.

- 1999-2000: robot companions (social robotics).
- 2005-2006: **video game** consoles (widespread controllers, GUIs, etc.).
- 2007: mobile OS and **smartphones** (ubiquitous devices/interfaces).
- 2012: head-mounted displays (consumer-grade **VR-AR** interfaces).

Nowadays, HCI is: user-centered, mostly graphics-based, and easy-to-use.

BASICS: HCI TODAY

HCI emerged in the early 1980s, with the advent of personal computers, as a subfield of computer science initially focused on improving the usability of desktop computers.

With the invention of computer networks and smartphones, HCI shifted its focus from desktop environment to mobile environment; and over time it has expanded to incorporate other disciplines: computer science, cognitive science, and human-factors engineering.

"...it no longer makes sense to regard HCI as a specialty of computer science... HCI expanded from its initial focus on individual and generic user behavior to include social and organizational computing, accessibility for the elderly, the cognitively and physically impaired, and for all people... It expanded from desktop office applications to include games, learning and education, commerce, health and medical applications, emergency planning and response... It expanded from early graphical user interfaces to include myriad interaction techniques and devices, multi-modal interactions, tool support for model-based user interface specification, and a host of emerging ubiquitous, handheld and context-aware interactions."

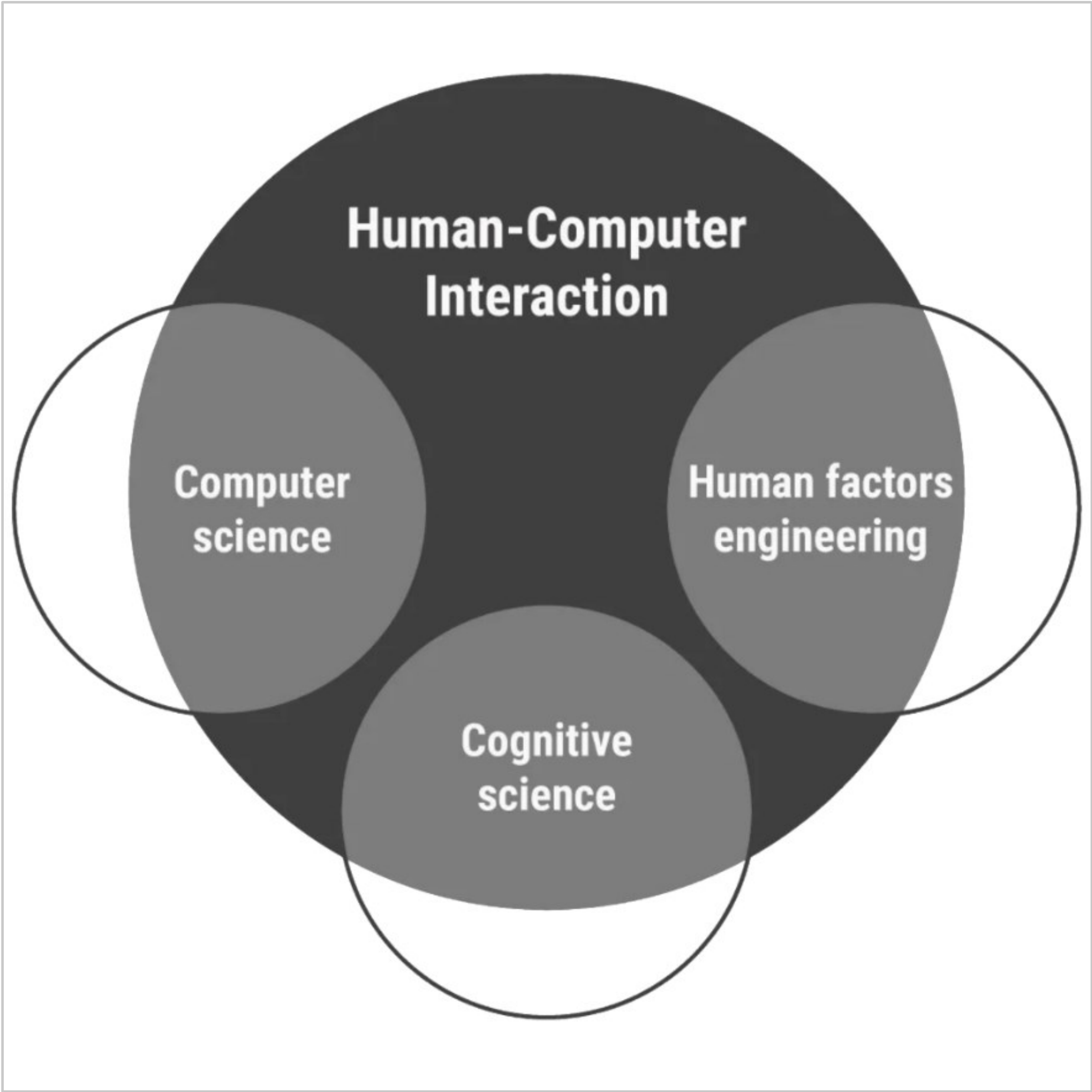
JOHN M. CARROLL. THE ENCYCLOPEDIA OF HUMAN-COMPUTER INTERACTION (2ND ED.) 2019.

BASICS: HCI TODAY (2)

HCI involves a multidisciplinary approach, drawing from computer science, cognitive science, and human factors engineering.

Additionally, HCI overlaps with many other fields of study: user-centered design, user-interface (UI) design, user experience (UX) design, and others (*e.g.*, ubiquitous computing, in-vehicle systems, ambient intelligence, etc.).

Today, we use devices that are more and more complex, and the cognitive load of HCI has constantly increased together with the operating error rate.



BASICS: HCI SUPPORTING TECHNOLOGY

Modern HCI involves a multitude of technologies:

- **Internet of Things (IoT):** Devices that can communicate with each other and with users by combining digital systems with sensors, actuators, and other devices.
- **Motion Tracking:** Technologies that can track in real-time the user/device motion. The outputs can be used to implement novel interactions and interfaces.
- **Speech Recognition:** Technology that can recognize human speech and that can be used to implement voice-based interfaces designed for natural language.
- **VR-AR:** Technologies that allow unconstrained immersive virtual environment and use specialized interface devices useful to design novel HDI.
- **Spatial Computing:** Technologies that use user actions (motions, speech) and physical environment as inputs for interactive systems that use the perceived surrounding environment as the *"canvas"* for multimedia outputs (video, audio, haptic).

BASICS: HCI EXAMPLES FOR TYPING

This section shows different interfaces for the same function: **text typing**. In particular:

- A **traditional typewriter** as an example of a function-centered HMI.
- A **modern keyboard** as an example of an HCI still using the typewriter metaphor.
- An **ergonomic keyboard** as an example of an HCI more user-centered.
- Innovative keyboard interfaces considering: different user needs, innovative technologies, different contexts.
    - A **one-handed gaming keyboard**.
    - A **laser projection mobile keyboard**.
    - An **AI-enabled mobile phone keypad**.
    - A **virtual keyboard** for VR-AR.

BASICS: HCI EXAMPLES FOR TYPING (2)

The Underwood Model 5 (right), commercialized in 1899, became the most successful office machine of the time, and set the modern standard of how a typewriter worked and looked like.

This HMI (the typewriter) is: function-centered (designed considering mechanical limitations and language properties), with minimal visuals, and definitely not easy-to-use.

BASICS: HCI EXAMPLES FOR TYPING (3)

In the early 1970s, computer keyboards started to look like what we use today.

In the late 1970s, personal computers started being commercialized including their own keyboards (one of the first was the IBM PC and its Model F keyboard, below).

This HCI (the keyboard) is: using the typewriter metaphor (same layout of keys), is still function-centered (key layout not considering user-hand workspace), is still using minimal visuals, but is now easy-to-use (because of the typewriter metaphor).

BASICS: HCI EXAMPLES FOR TYPING (4)

The limitations of function-centered keyboards lead alternative solutions.

Starting in 1926, ergonomics keyboards were proposed to solve the issues of standard keyboards. One of the first commercially-successful ergonomic keyboards was the Natural Keyboard by Microsoft (below the 2019 Microsoft Surface Ergonomic Keyboard 4000).

This HCI (the keyboard) is: using only in part the typewriter metaphor (same keys layout, but no grid layout), and is now more user-centered (considering hands orientation).

BASICS: HCI EXAMPLES FOR TYPING (5)

An example of an innovative user-centered HCI: a one-handed gaming keyboard designed for video gamers, allowing at the same time the typing with the left hand and the pointing-and-clicking with the right hand.

BASICS: HCI EXAMPLES FOR TYPING (6)

An example of an innovative user-centered HCI: a laser projection mobile keyboard designed for mobile phone users, facilitating typing by increasing the size of the keyboard surface relying on laser projection and infrared finger tracking.

BASICS: HCI EXAMPLES FOR TYPING (7)

An example of an innovative user-centered HCI: an AI-enabled mobile phone keypad, the Nokia 7110 was the first mobile phone to integrate the T9 predictive text method to facilitate the composition of SMS text messages using a 9-keys keyboard.

BASICS: HCI EXAMPLES FOR TYPING (8)

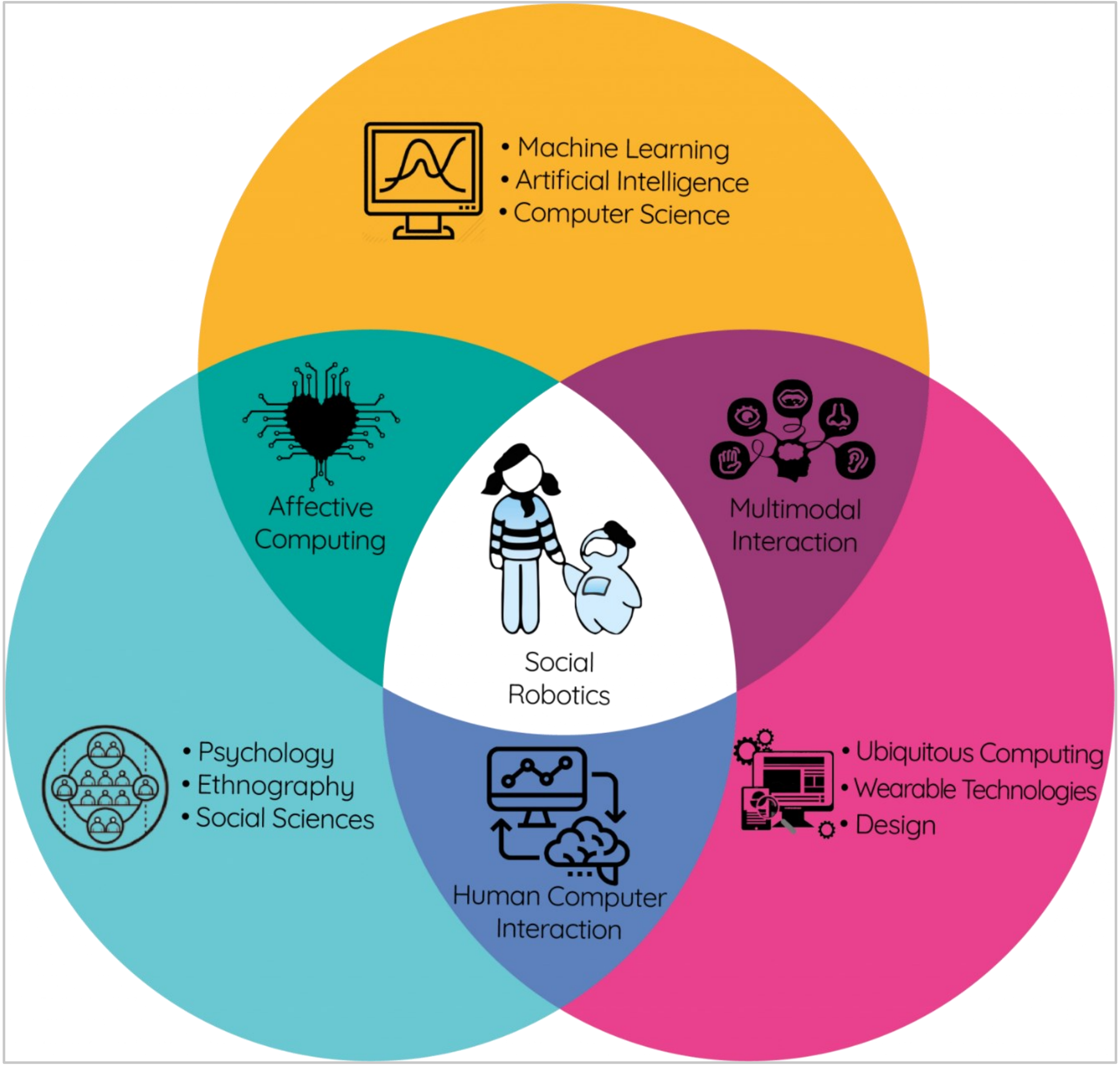
An example of an innovative user-centered HCI: a virtual keyboard for VR-AR, allowing text typing by using controllers and replacing the lack of haptic feedback with visuals and sounds.

BASICS: HCI AND ARTIFICIAL INTELLIGENCE

Nowadays, the cognitive load of human-computer interactions constantly increases together with the operating error rate.

This has promoted the integration of AI techniques into HCI interfaces, opening new possibilities for humans and computers to interact.

AI algorithms have been developed to use user feedback with the goal of automating/facilitating a variety of HCI tasks: from UX design, to accessibility, and human-AI collaboration.



PART 2: VR INTERACTION-UI DESIGN

This section briefly explores the key considerations for VR interaction design:

- Hardware requirements, and adverse health effects.
- User comfort and safety.
- Immersive audio.
- User interface (UI) design.
- Interaction design.
- Scale and proportion.
- Performance optimization.
- Navigation and wayfinding, storytelling and narrative, and content creation.
- Accessibility and inclusivity.
- Testing and feedback.
- Performance feedback.

BASICS: COMFORT

Physiological Comfort: Body senses do not feel conflict in sensory stimulation.

Lack of physiological comfort results in fatigue, nausea, etc.

Environmental Comfort: Discomfort depending on the environment.

Lack of environmental comfort results in claustrophobia, vertigo, etc.

BASICS: VR SICKNESS

Motion Sickness: Discomfort that occurs due to a difference between actual/real and expected/perceived motion.

Common symptoms are: nausea, cold sweat, headache, sleepiness, etc.

VR Sickness: (aka Simulator Sickness) It is the discomfort caused by sensory conflicts experienced by a VR user, and generated by a VR experience.

Common symptoms are: nausea, cold sweat, headache, sleepiness, etc.

Ergonomics: The application of psychological and physiological principles to the engineering and design of products, processes, and systems. The goals are to reduce human error, increase productivity, enhance safety and comfort, with a specific focus on human-X interactions.

BASICS: VR SICKNESS (2)

These factors can cause VR sickness:

- **Acceleration** (minimize any acceleration).
- **Control** (ensure that the user has a good degree of control in VR).
- **Session Duration** (allow breaks during VR sessions or design short sessions).
- **Visual Flow** (avoid strong visual flows).
- **Binocular Disparity** (remember that not everyone is capable of fusing stereo pairs).
- **FOV** (decreasing the FOV could reduce comfort).
- **Latency** (minimize it because *"lags"* cause discomfort in VR).

Remember that VR developers/users become resilient to VR sickness!

BASICS: VR AVATAR

VR Avatar: A VR avatar is a virtual representation of the user in the VR environment, and it facilitates both locomotion and interactions.

A VR avatar can range from a basic 3D shape (e.g., a capsule-like body) to an animated 3D model (e.g., a player character).

See: en.wikipedia.org/wiki/Avatar_(computing)

Codec Avatar: Invented at Facebook Reality Labs (FRL), Pixel Codec Avatars (PiCA) are AI-models of 3D human faces optimized for reconstruction/computation.

See: research.facebook.com/publications/pixel-codec-avatars/

See: www.youtube.com/watch?v=TKIxw0vh9X0

BASICS: HAPTICS

Force Feedback: Also called kinesthetic feedback, it refers to the feelings provided by sensors in your muscles, joints, etc. Humans use this feedback to estimate properties (sizes, weights, etc.) of objects we touch.

Tactile Feedback: It refers to the feelings provided by the sensors in your skin tissue. Humans use this feedback to estimate vibration, texture, etc.

Haptics: Any technology that can create an experience of touch by applying forces, vibrations, or motions to the user. Simple haptic device examples are: game controllers integrating vibrations, steering wheels with torque feedback, etc. Haptic feedback includes both force/kinesthetic feedback and tactile feedback.

HARDWARE: VR CONTROLLERS

Standard input devices (mouse, keyboard etc.) are not user-friendly during a VR session. So, VR controllers are the standard input devices used to interact with the virtual world.

This is a short list of the most common VR controllers:

- **Wand:** a hand-held joystick (see Nintendo Wii), often including a tracker.
- **VR Gloves (Data Gloves):** special gloves able to track hand-finger movements, often including basic tactile feedback (vibrations).
- **Hand-Finger Trackers:** trackers designed to track hand-finger movements (see Leap Motion), often integrated on HMDs to implement touch-less gestural interactions.
- **VR Controllers:** a pair of hand-held joysticks (see Oculus Touch Controllers), usually integrating both 6-DOF tracking and basic tactile feedback.
- **Haptic Stylus:** a stylus-like joystick attached to a simple robotic arm (see Sensable Phantom Omni), always integrating both 6-DOF tracking and haptic feedback.

HARDWARE: VR CONTROLLERS (2)

VR Controllers

A pair of hand-held joysticks, usually integrating both 6-DOF tracking and basic tactile feedback.

Oculus Touch Controllers

These controllers consist in a pair of handheld units, each featuring: 1 analog stick, 3 buttons, and 2 triggers.

Some versions also include: a dedicated thumbrest, and a system for detecting finger gestures.

The controller ring contains infrared LEDs to allow 6-DOF tracking.

HARDWARE: VR TRACKING

Pose Tracking: In VR a pose tracking system detects the precise pose (3D position and orientation) of HMDs, VR controllers, and other objects or body parts.

Pose tracking is often referred to as 6-DOF tracking, for the six degrees of freedom in which the pose is often tracked.

Note: Pose tracking is sometimes referred to as positional tracking, but it is an error!
Pose tracking tracks both 3D position and 3D orientation of a target.
Positional tracking only tracks the 3D position of the target (and no 3D orientation).

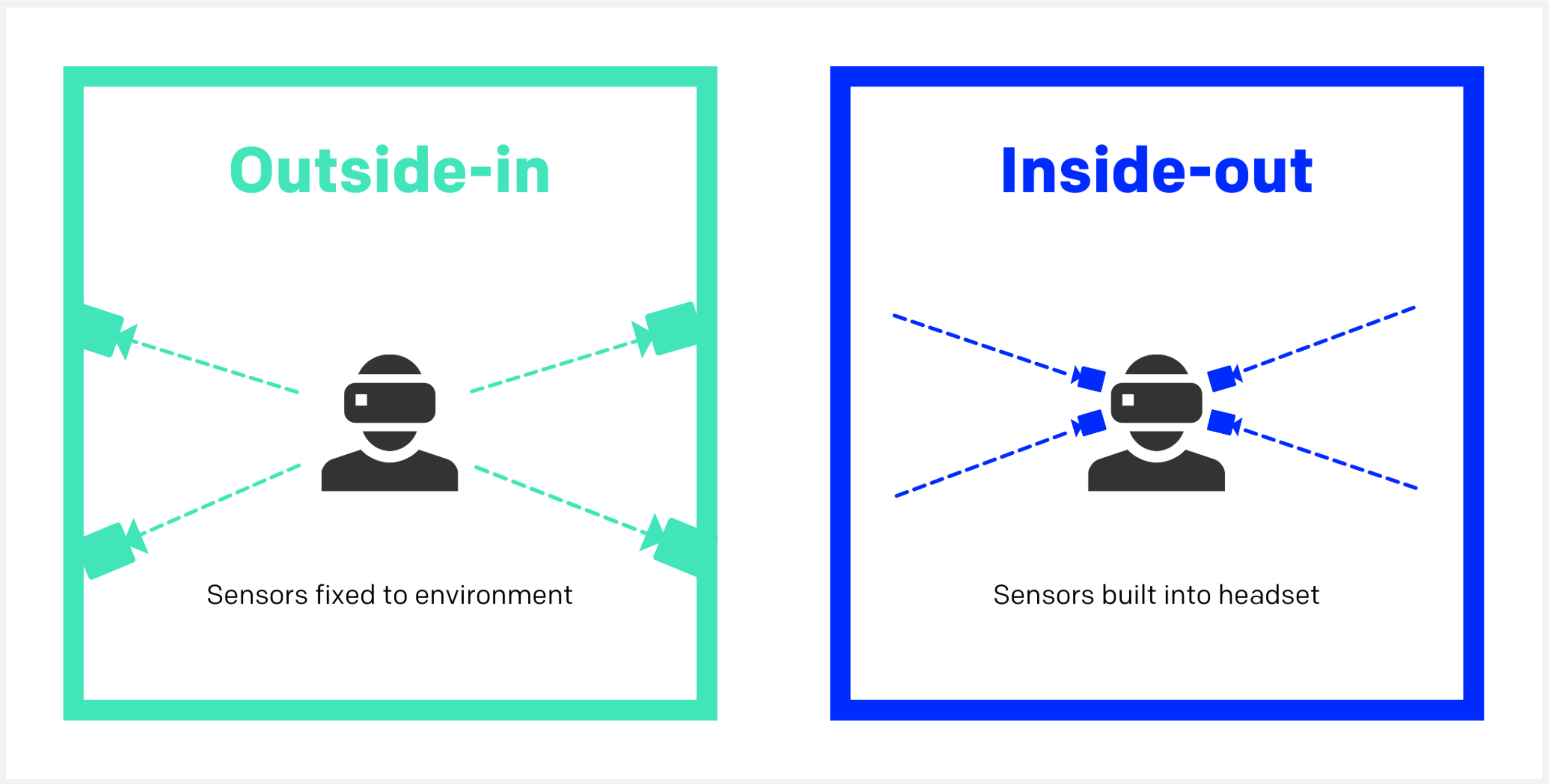
Outside-In Tracking: Tracking cameras are placed in static locations to track the position of markers on the target (HMD, VR controllers, etc.).

Inside-out tracking: Tracking cameras are placed on the target (HMD) and look outward to determine its location considering the surrounding environment.

See: en.wikipedia.org/wiki/Pose_tracking

HARDWARE: VR TRACKING (2)

Comparison between outside-in and inside-out tracking methods for VR.



See: *"Inside a VR Headset: Outside-In Tracking"* on YouTube.

See: *"Inside a VR Headset: Inside-Out Tracking"* on YouTube.

DESIGN: KEY CONSIDERATIONS

Interactions and user interfaces (UIs) in VR present unique challenges and considerations compared to real-world interactions and traditional 2D UIs design:

- **User Comfort and Safety:** Prevent motion sickness, eye strain and other discomfort with high FPS, low latency, and comfort features (e.g., teleportation).
- **Immersive Audio:** Use 3D spatial audio to enhance the sense of presence and immersion in VR.
- **User Interface (UI) Design:** Design intuitive and easily navigable UIs that are seamlessly integrated into the VR environment.
- **Interaction Design:** Implement natural and intuitive interaction techniques (hand tracking, gesture recognition, controller-based, etc.) to enhance user engagement.
- **Scale and Proportion:** Ensure that objects and spaces in VR maintain realistic scale and proportion to create a convincing sense of presence.

THEORY: KEY CONSIDERATIONS (2)

Interactions and user interfaces (UIs) in VR present unique challenges and considerations compared to real-world interactions and traditional 2D UIs design **(continued)**:

- **Accessibility and Inclusivity:** Consider diverse audiences and accessibility features (adjustable font sizes, voice commands, controller settings, etc.).
- **Testing and Feedback:** Continuously test and gather user feedback to identify and address design flaws, comfort issues, and usability concerns.
- **Performance Feedback:** Provide users with feedback on their interactions (haptic feedback, visual cues, audio responses, etc.) to enhance the feeling of agency.

DESIGN: ACCESSIBILITY IN VR

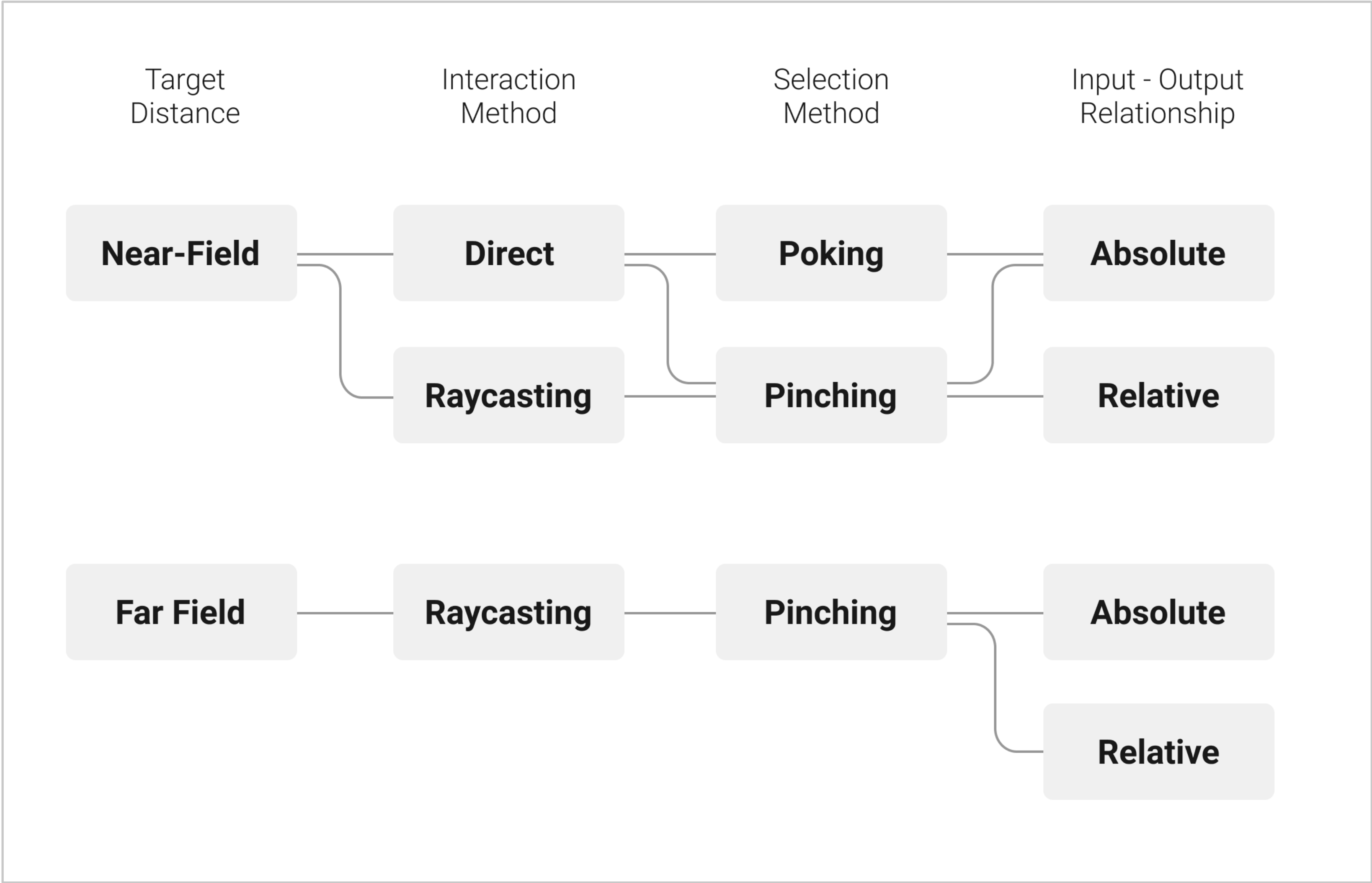
This diagram illustrates the main factors to consider in achieving accessibility in VR.

DESIGN: VR INTERACTION DESIGN EXAMPLE

Interaction design in VR involves creating intuitive and engaging ways for users to interact with a 3D virtual environment.

Key considerations: 3D spatial interactions, affordances, signifiers, feedback, safety and comfort.

The diagram (right) shows **best practices for hand/finger interactions.**



THEORY: VR INTERACTION DESIGN CONCEPTS

Every interactable object (UI element) should *"hint"* at how to interact with it.
Moreover, in VR (usually) there is no/limited tactile feedback to confirm user actions.

These two issues are usually solved by reinforcing on all interactions the communication of affordances through clear signifiers and continuous feedback.

Affordances: What you can do with an object.

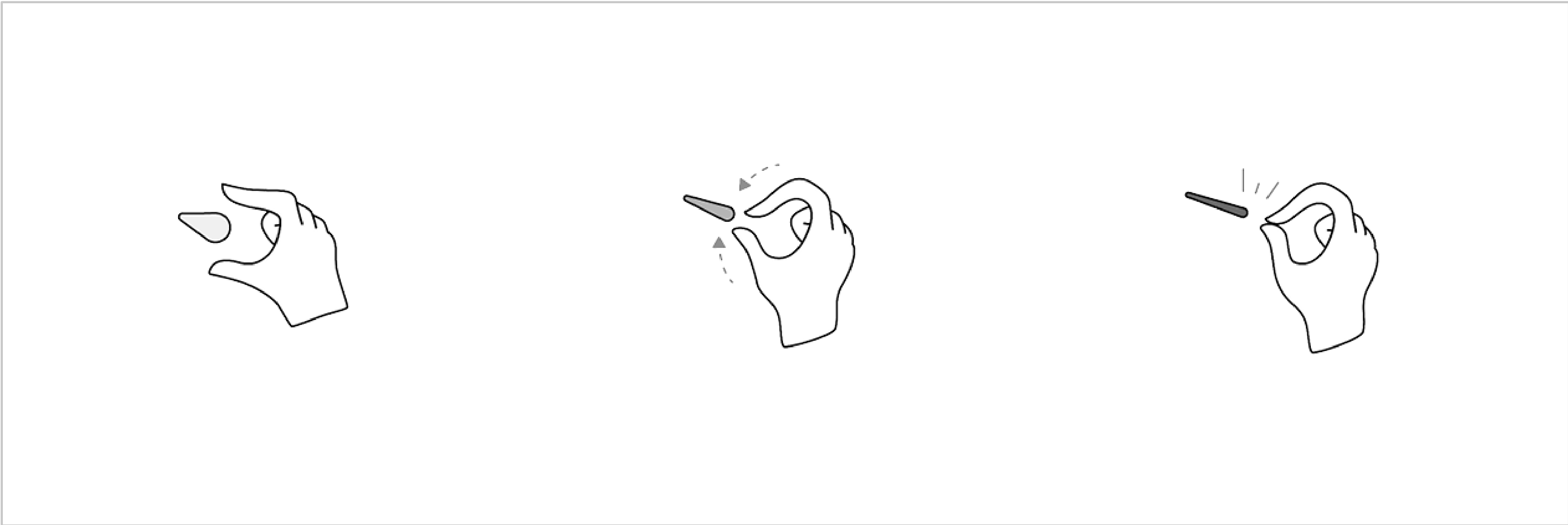
Signifiers: Communicate the relative affordances to the user.

Feedback: Confirm to the user the interaction/user state throughout the interaction.

Example: Seeing a door, the user assumes it affords opening (affordance); the handle (signifier) communicates this affordance to the user; then, during the interaction, the door/handle opening/turning (interaction) is continuously confirmed to the user just by looking at them (continuous feedback).

DESIGN: VR POINTING INTERACTION EXAMPLE

VR Example: When a user wants to interact with the VR system via pointing (affordance), the user can achieve a *"poiting state"* (left) that activates the visualization of a pointer (signifier, left). The pointer is squishable so that the user understands that **a *"click interaction"* is achieved by squishing the pointer enough.** Continuous feedback is provided by the squishing/color animation of the pointer through the point-and-click interaction.

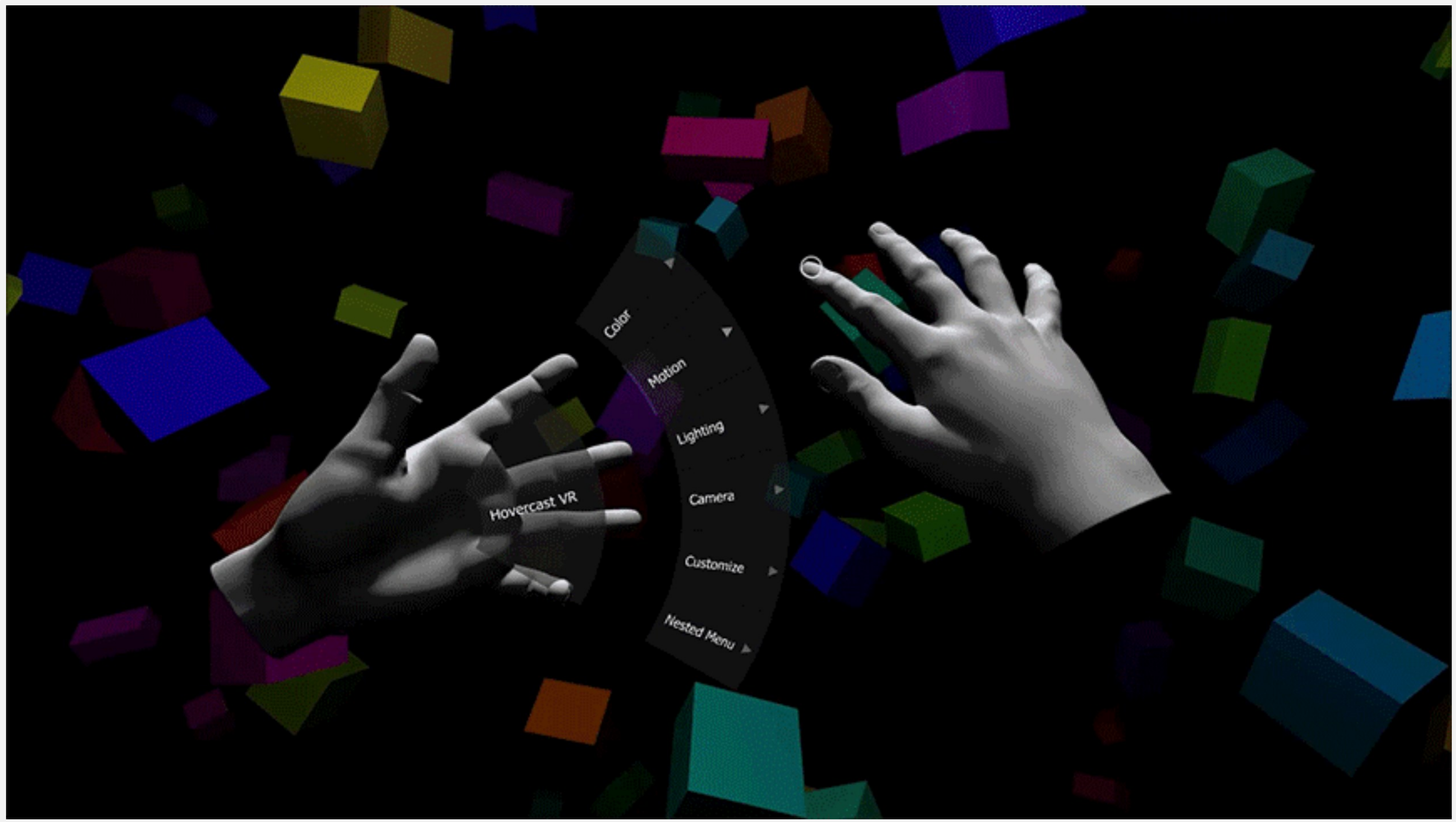


DESIGN: VR UI EXAMPLES

A traditional non-diegetic UI (left), and a VR UI (right).

DESIGN: VR UI EXAMPLES (2)

Another example of a VR UI (Hovercast VR Menu by Leap Motion).



DESIGN: VR UI ELEMENTS

The design and implementation of user interfaces (UIs) for VR includes:

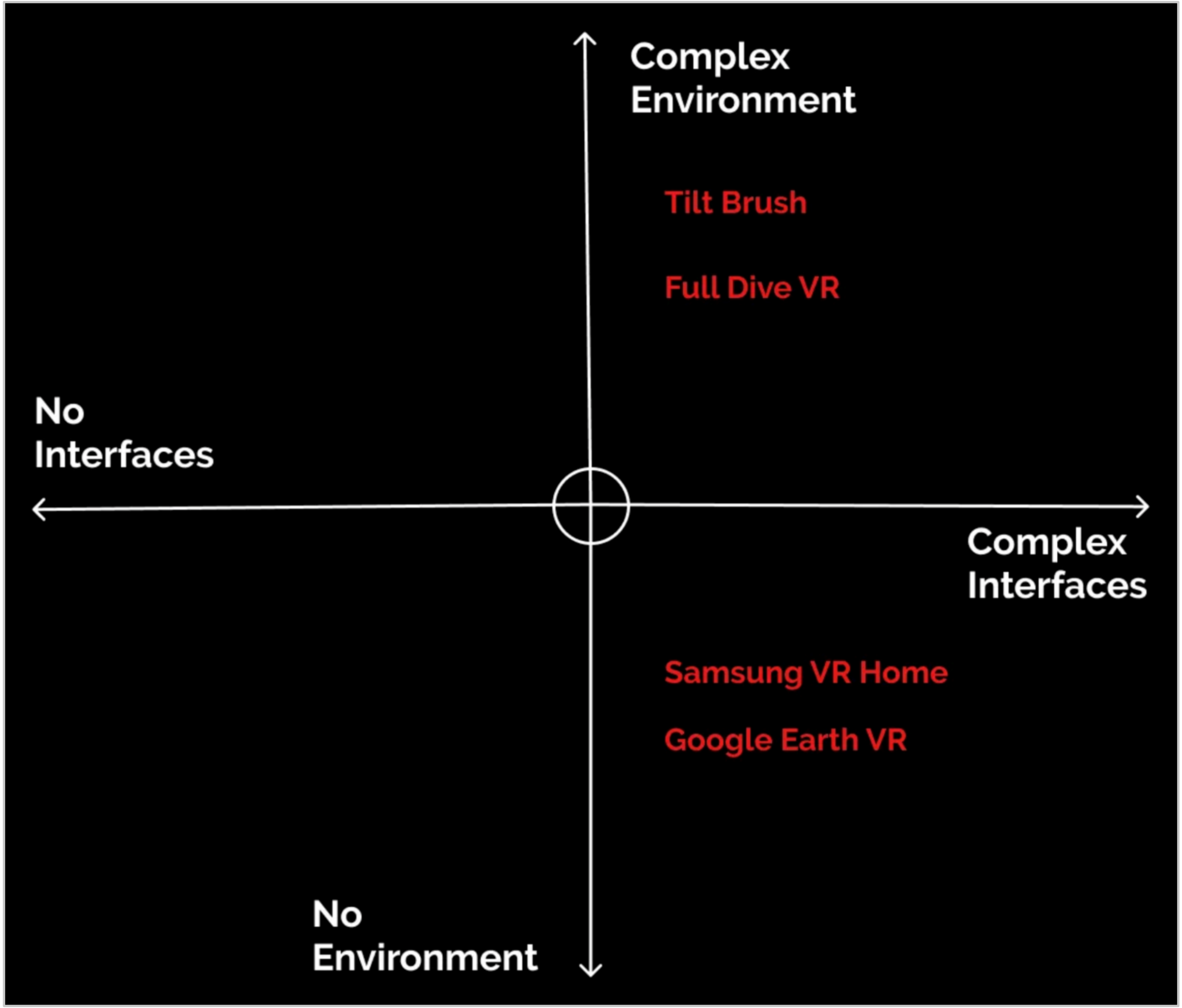
- **Menus/Panels:** for specific views, in-game, integrated in 3D (diegetic), etc.
- **Heads-Up Displays (HUDs):** minimaps, timers, etc.
- **Controls/Interactions:** keyboard-mouse, point-and-click, joypad, etc.
- **Interaction Feedback:** visual (highlighting), audio (clicks), tactile (vibration), etc.

Designing UI for VR means:
determining which information/interactions should be available to the user,
and how these will be implemented
(usually maximizing comfort and usability).

DESIGN: VR UI INTERFACE VS ENVIRONMENT

A VR user interface (UI) is the set of elements that users interact with to control the VR environment.

A VR app can be categorized depending to the complexity of its interface and enviroment.



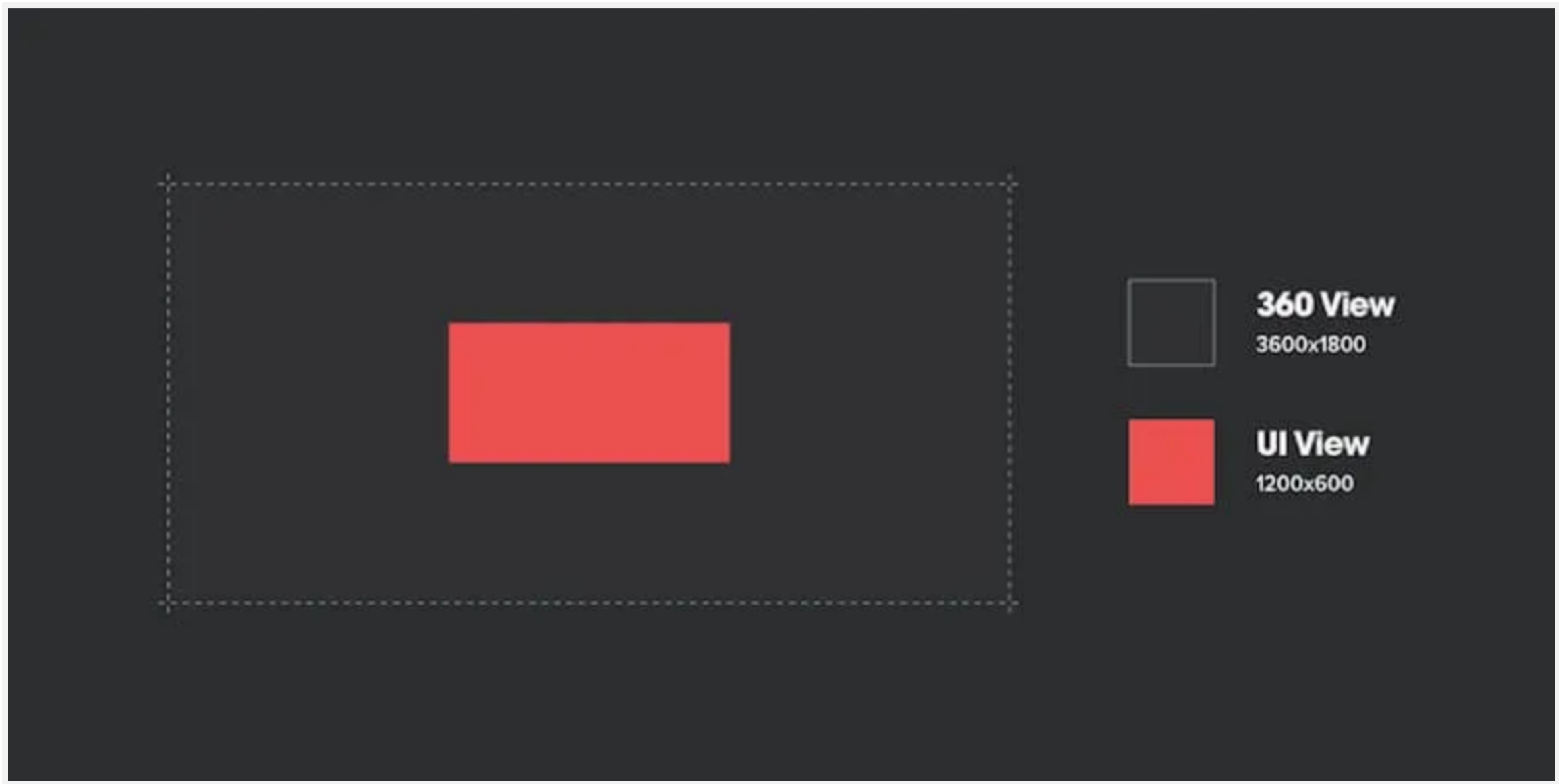
DESIGN: VR UI CANVAS

Initially, to design a VR UI, the canvas size has to be determined.

Equirectangular Projection: A 360-degree view of the environment, flattened as a 2D rectangular image.

DESIGN: VR UI CANVAS (2)

Following the guidelines by Mike Alger on comfortable viewing areas, we can define the part of the canvas that will accommodate the VR UI.



DESIGN: VR UI CANVAS (3)

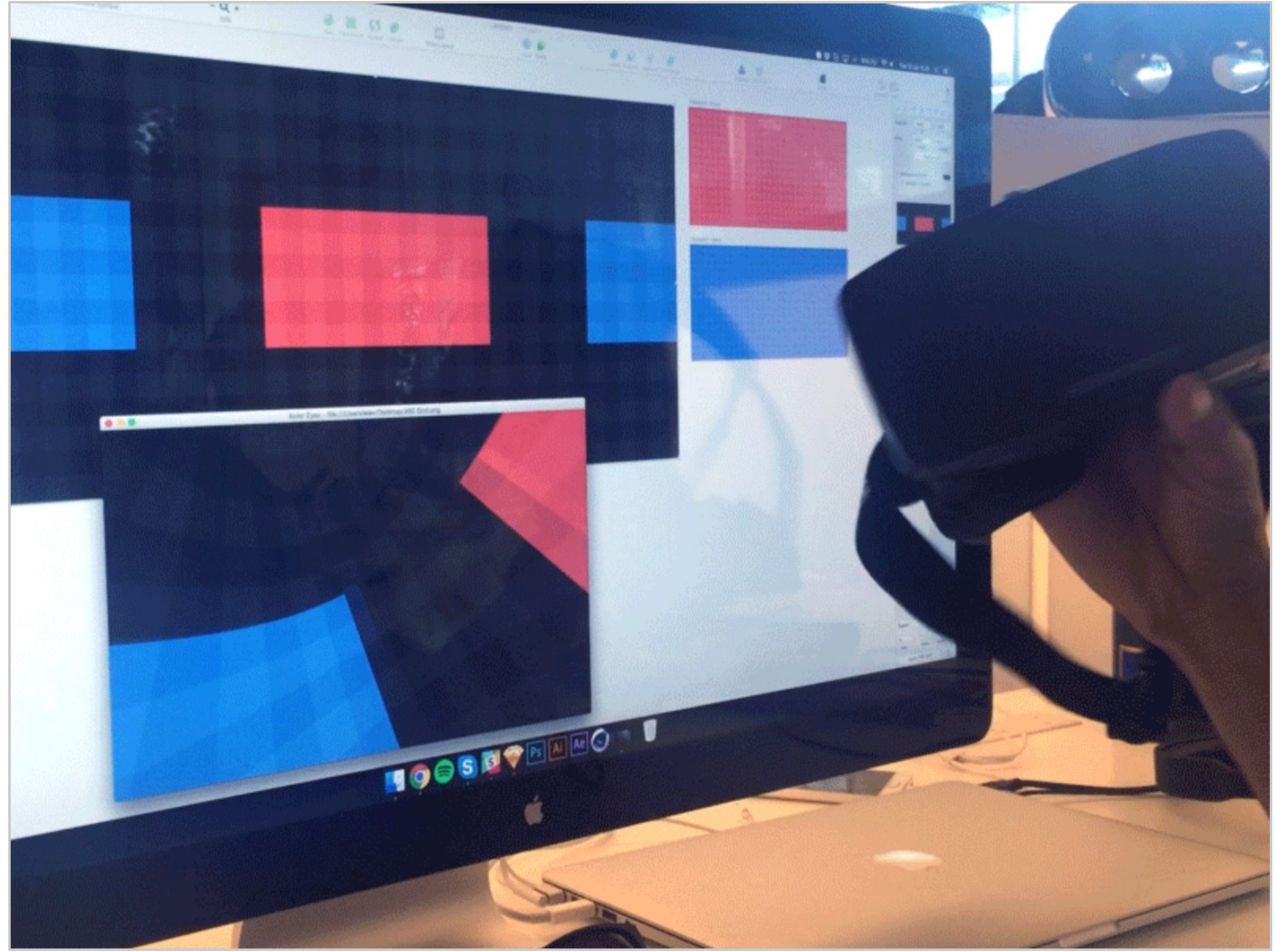
Following the guidelines by Mike Alger on comfortable viewing areas, we can define the part of the canvas that will accommodate the VR UI.

DESIGN: VR UI CANVAS (4)

Using 2 canvases (*"UI View"* and *"360 View"*) for a single screen allows testing the UI.

The ***"UI View"* canvas** helps to keep our focus on the interface we are designing.

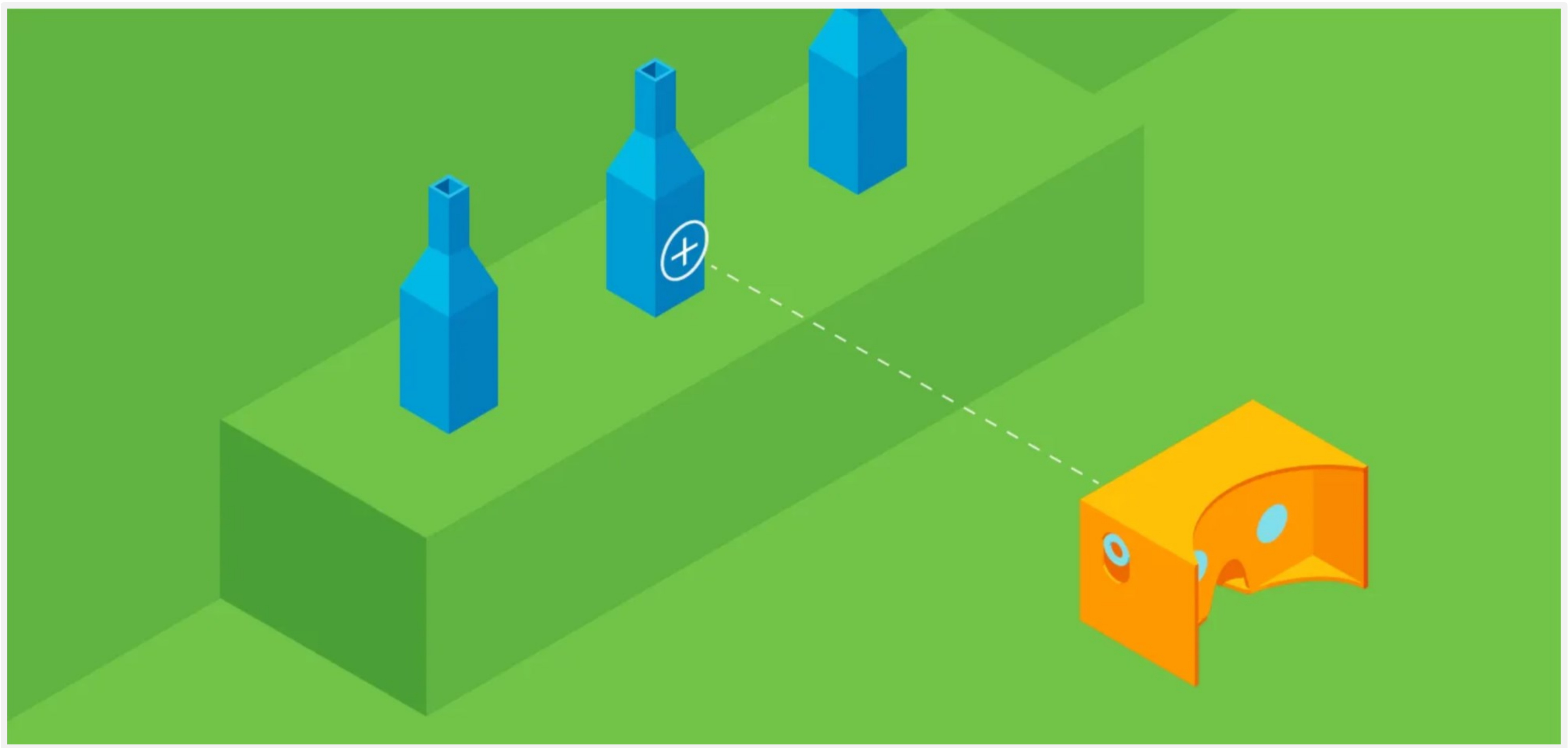
The ***"360 View"* canvas** is used to preview the interface in VR (checking proportions, and testing the UI using a VR headset).



See: www.smashingmagazine.com/2017/02/getting-started-with-vr-interface-design

DESIGN: VR UI RETICLE

Reticle: A visual aid for the user to target objects within a VR environment. It can be used to facilitate a targeting task controlled by different means (gaze, controllers, etc.), and its visual style can vary.



DESIGN: USER VIEWING-INTERACTION ZONES

To design a comfortable VR UI, avoiding user fatigue in longer VR sessions, comfortable range of motion zones must be considered.

Careful with Distance: Avoid making the VR user look back and forth between something close and something far away (switching focus often).

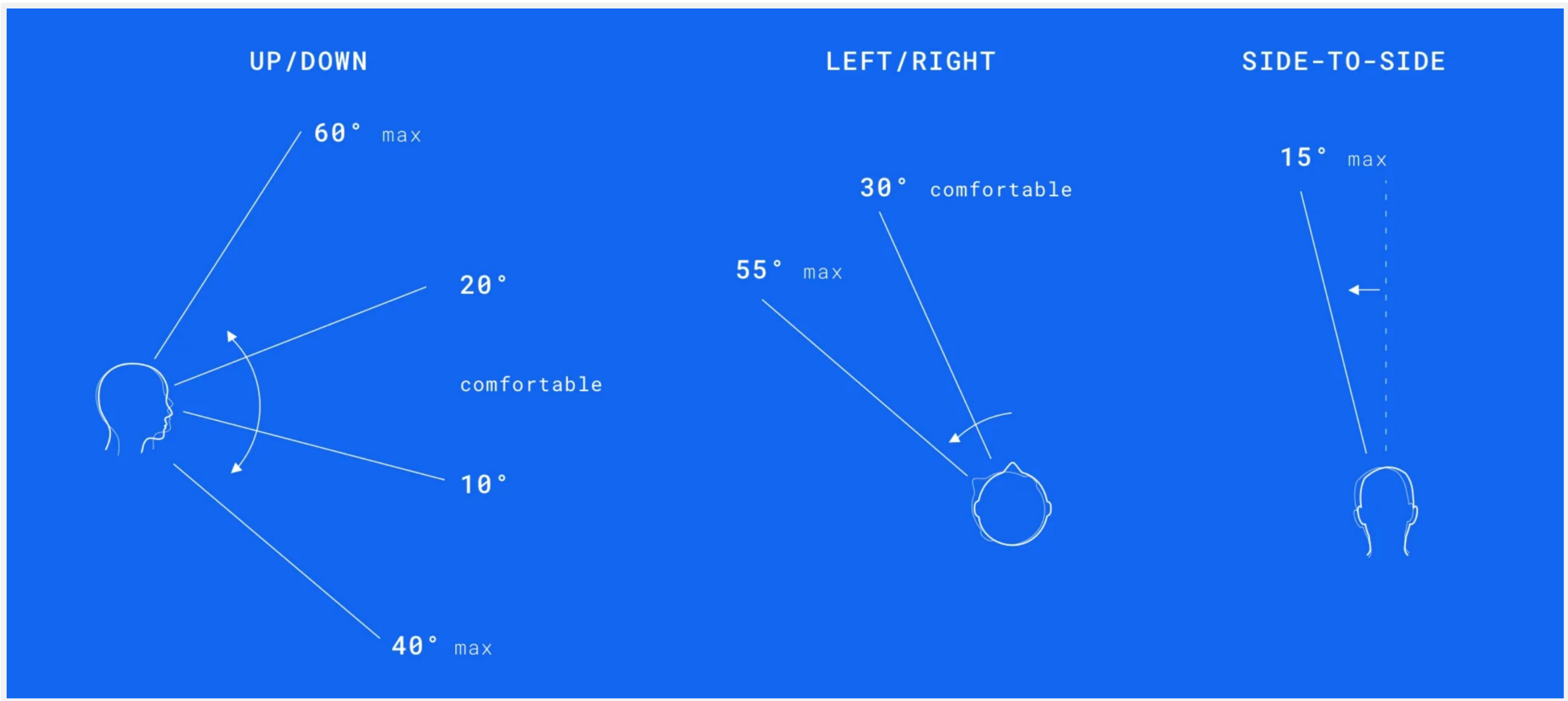
Keep all visual elements of a task at approx. the same depth.

***"Goldilocks"* Zone:** Excessive head movement could cause VR sickness.

Keep interactive elements between desk height and eye level (*"Goldilocks"* zone, not too high and not too low).

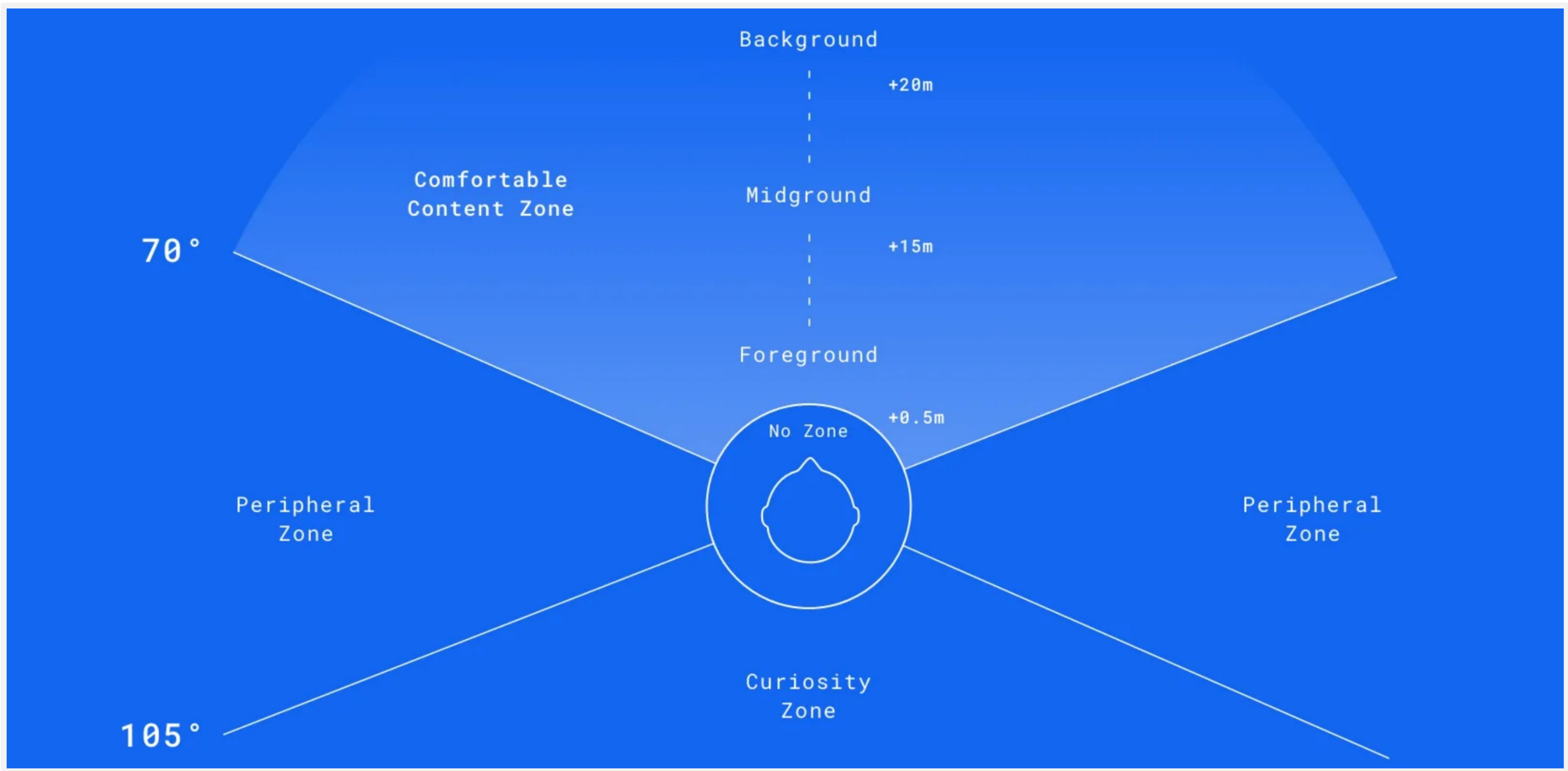
DESIGN: USER VIEWING ZONES (2)

Comfortable range of motion zones for VR UI **(continued)**:



DESIGN: USER VIEWING ZONES (3)

Comfortable range of motion zones for VR UI **(continued)**:

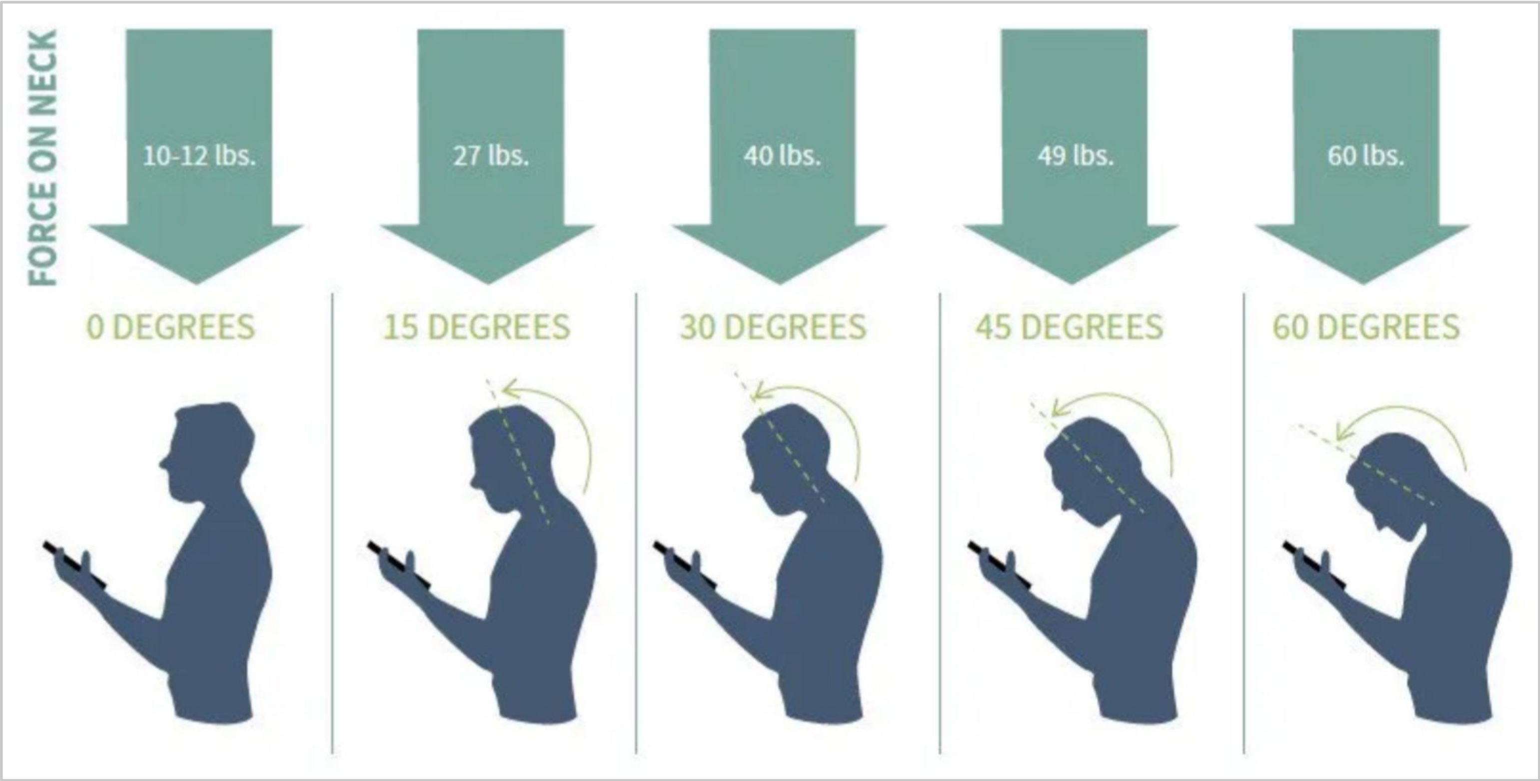


DESIGN: VR UI VIEWING ZONES (4)

Comfortable range of motion zones for VR UI **(continued)**:

***"Text Neck"* Syndrome:** soreness felt from looking down at a smartphone for an extended period of time.

The diagram shows the force on the user neck depending on the viewing angle. Poor posture can result in up to 60 pounds of pressure on the user spine.



THEORY: SEMANTIC/RESPONSIVE GESTURES

Gesture-based interactions can be categorized into 2 main groups:

- **Semantic Gestures:** Common movements the user is familiar with in the real-world (walking, nodding for *"yes,"* etc.).
- **Responsive Gestures:** The main way the user interacts with the environment (picking objects up, throwing objects, pushing buttons, etc.).

Designing responsive gestures is harder because of the specifics of the interacting object (weight, shape, etc.).

The main goal in interaction design for VR is to duplicate reality, so:

- Rely on as many natural gestures as possible,
- Limit the number of *"new"* controls the user has to learn.

THEORY: DIEGETIC INTERFACES

In reality, there are no menus; so, these UIs counteract VR immersion. Better alternatives are diegetic UIs.

Diegetic UI: A UI that looks like it is part of the virtual environment (something that the VR avatar would interact with in the VR world).

See: en.wikipedia.org/wiki/Half-Life:_Alyx

DESIGN: VR TEXT AND IMAGES

Concave Text and Images: Because human vision is curved, it is better to display text and images that are not part of the VR environment as a slightly curved, concave surface.

This allows for a better immersion, and will look smoother during head rotations.

Text Readability: Because of the display resolution, all UI elements could look pixelated.

Text could be difficult to read, and there could be a high level of aliasing.

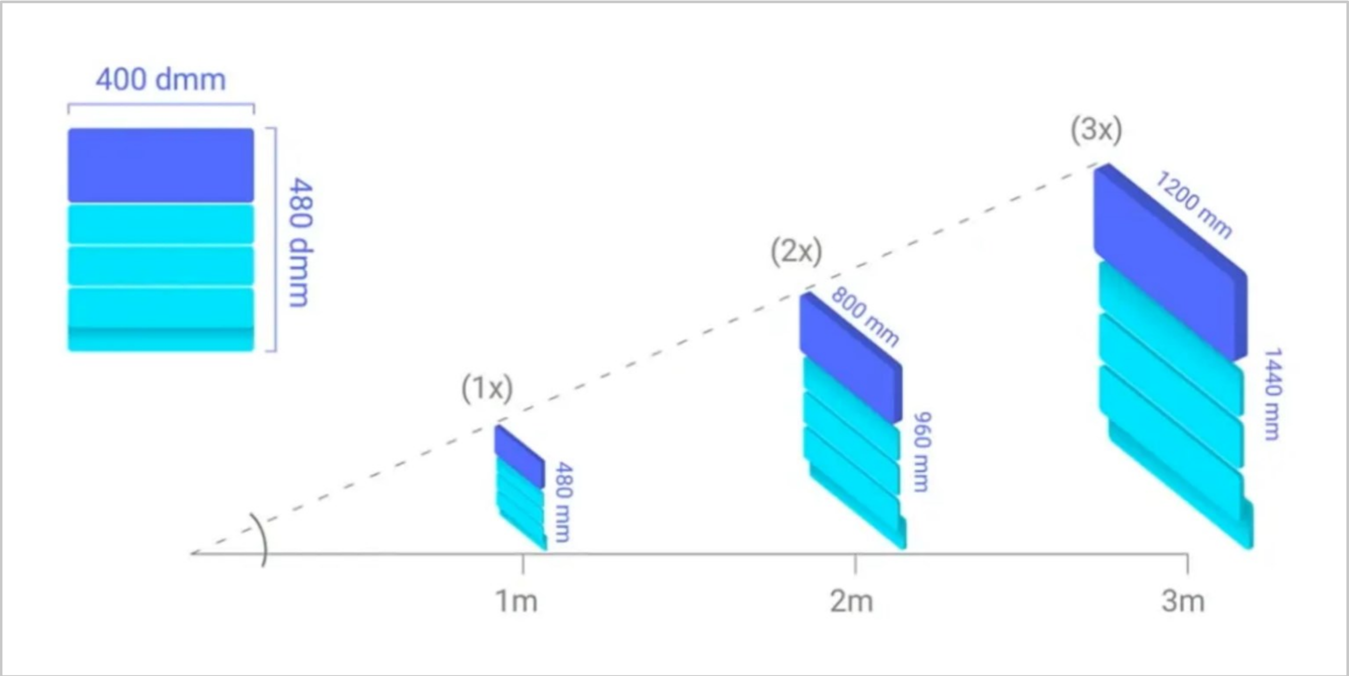
Avoid using big text blocks and highly detailed UI elements.

DESIGN: VR TEXT AND IMAGES (2)

Intended Viewing Distance: The distance at which an element (UI) has been designed to be viewed (comfortably). This affects the size of the (UI) element as well as the density of its content.

A **dmm (distance-independent millimeter)** is an angular measurement unit representing 1 millimeter at 1 meter away (it does not depend on the viewing distance).

Sizing an element (UI) in dmm, facilitates its actual sizing once the viewing distance is decided.



DESIGN: VR TEXT AND IMAGES (3)

Guidelines for text size (in dmm) in VR (from Google VR Sticker sheet, Google I/O 2017).

DESIGN: VR AVATARS

Render VR avatars only when necessary!

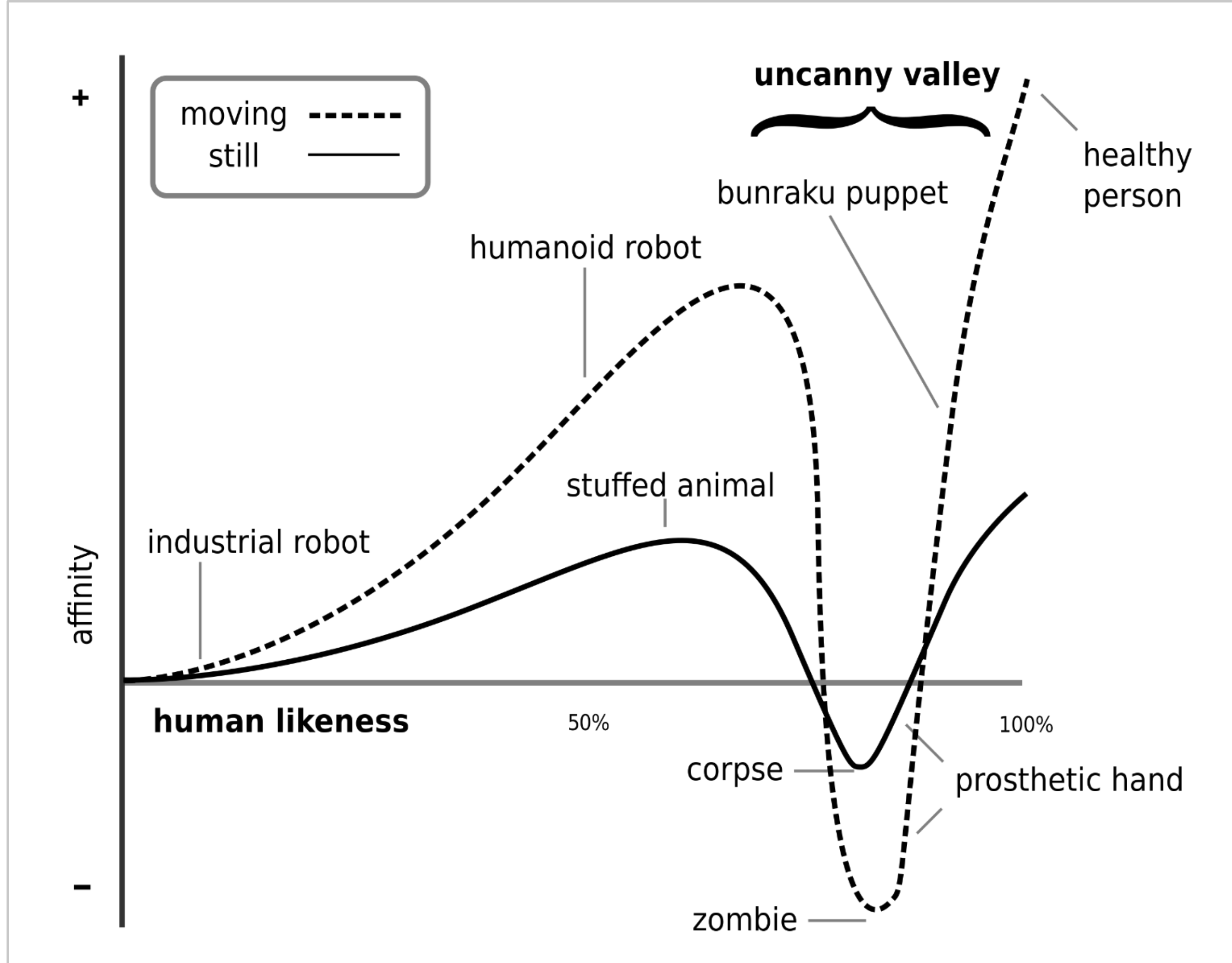
It may be better to have no body at all (no VR avatar for the user) than to have a body with distracting imperfections (not in synch with the user movements, different size than the user body, etc.).

The significance of a VR avatar depends on the application; so, be careful with your design.

DESIGN: VR AVATARS (2)

The *"uncanny valley"* effect is a hypothesized psychological and aesthetic relation between an object's degree of resemblance to a human being and the emotional response to the object.

The *"uncanny valley"* hypothesis predicts that an entity appearing almost human will risk eliciting eerie feelings in viewers.



See: en.wikipedia.org/wiki/Uncanny_valley

DESIGN: USER MOVEMENTS AND POSTURE

Human Arc Movements: Humans move limbs in arcs, not straight lines.

User Posture: Many VR interactions involve hands/controllers; so, encourage users to hold their hands upright and at the ready.

Whenever the user can see their hands in VR, interaction is easier.

DESIGN: VR BEST PRACTICES

This is a summary of the *"best practices"* to design and implement VR systems:

- Do not neglect monocular depth cues (e.g., illumination, details, etc.).
- The most comfortable depth range in VR is 0.75-3.50 m.
- Ensure that text elements in VR are easy to read and avoid small unnecessary elements in areas where the user focuses.
- The most comfortable VR experiences do not integrate any *"self motion"* for the user, excluding head/body movement to look around.
- If *"self motion"* is necessary in VR, slow user movements are the most comfortable.
- Ensure that any form of acceleration (of the user or other elements) is as hort and unfrequent as possible.

DESIGN: VR BEST PRACTICES (2)

This is a summary of the *"best practices"* to design and implement VR systems (continued):

- User head movements and virtual camera movements should always match!
- Do not integrate *"head bobbing"* in the VR avatar movements.
- To achieve comfortable VR experiences, minimize backward and lateral movements.
- Careful with situations inducing strong vection (visual feeling of motion), for example climbing stairs, accelerating, etc.
- Minimize (if possible) both *"lag"* and latency.
- If latency is unavoidable, a variable *"lag"* is worse than a constant *"lag"*.

Follow these guidelines to maximize comfort and usability of a VR experience, minimizing: visual fatigue, disorientation, nausea, etc.

DESIGN: VR UI DESIGN GUIDELINES

These are some guidelines to design and implement UI for VR:

- Integrating UIs in the virtual environment (diegetic UIs) is the best design.
- Visualize reticles/crosshairs on the target (not at a fixed distance).
- Hide virtual instruments when not in use.
- Be careful with inconsistencies of virtual avatars.
- No standard input device is ideal in VR, the best option is a gamepad.
- Use familiar input devices (remember that the user does not see the controllers in VR).
- Be careful with the intense use of gestures and gaze (it can easily cause fatigue).
- Locomotion is still unexplored in VR (it can create issues never seen before).

THEORY: STEREO RENDERING GUIDELINES

Summary of the main concepts for a correct comfortable stereo rendering:

- 3D objects in front of the projection plane will appear in front of the physical display.
- 3D objects behind the projection plane will appear *"inside"* the physical display.
- Usually, it is easier to fuse stereo pairs of 3D objects appearing *"inside"* the physical display; so, the focal point should be placed nearer to the virtual stereo camera than the target 3D objects.
- To facilitate fusing stereo pairs, the absolute value of the horizontal parallax angle (theta) should be smaller than 1.5 degrees for all individual 3D points.

THEORY: STEREO RENDERING GUIDELINES (2)

Summary of the main concepts for a correct comfortable stereo rendering **(continued)**:

- The magnitude of depth perception depends on both (1) the distance between the virtual camera and the projection plane, and (2) the left/right camera separation.
- A left/rigth camera separation too high (aka hyperstereo) could cause the generation of stereo pairs that are difficult to fuse (discomfortable viewing).
- A good approximation for left/right camera separation is 1/20 of the distance between the camera and the projection plane (this can also be considered the max camera separation for a comfortable stereo viewing).
- It is a good practice to ensure that the negative horizontal parallax never exceeds the left/right camera separation.

THEORY: STEREO RENDERING GUIDELINES (3)

Summary of the main concepts for a correct comfortable stereo rendering **(continued)**:

- The aperture of the virtual camera should equal the *"sweet spot"* of the VR headset.
- Properly select the focal distance (associated with individual 3D points with zero horizontal parallax) considering the 3D content, to avoid 3D object too close to the virtual camera (nearer than half the focal distance).
- Using a left/right camera separation similar to the average IPD (human): (1) we can achieve a realistic sense of scale and distance in VR, (2) we could obtain weak dept perception for 3D content distant from the virtual cameras.

PART 3: VR SW DEVELOPMENT IN UNITY

"Unity is so much more than the world's best real-time development platform – it's also a robust ecosystem designed to enable your success. Join our dynamic community of creators so you can tap into what you need to achieve your vision."

UNITY TECHNOLOGIES

Applicazion Fields: videogames, architecture, automotive, movies, XR, etc.

"Create once, deploy across 25+ leading platforms and technologies to reach the largest possible audience."

UNITY TECHNOLOGIES

See: unity.com

REFERENCES

- These slides are available online at: github.com/turinig/vrphd

- Unity website: unity.com
- The UX of VR website: www.uxofvr.com
- Mike Alger website: mikealger.com

- *“Virtual Reality”* S. M. LaValle: lavalle.pl/vr
- *“Visual Design Methods for Virtual Reality”* M. Alger: aperturesciencellc.com/vr

- *“Interaction Design Foundation: Virtual Reality”*
- *"How to Design for Virtual Reality: Basics and Best Practices for VR Design"*
- *"Designing User Experience for Virtual Reality Applications"*
- *"Virtual Reality (VR) Design & User Experience"*
- *"New to VR? Here is What You Must Know to Get Started"*
- *"Design Practices in Virtual Reality"*

APPENDICES

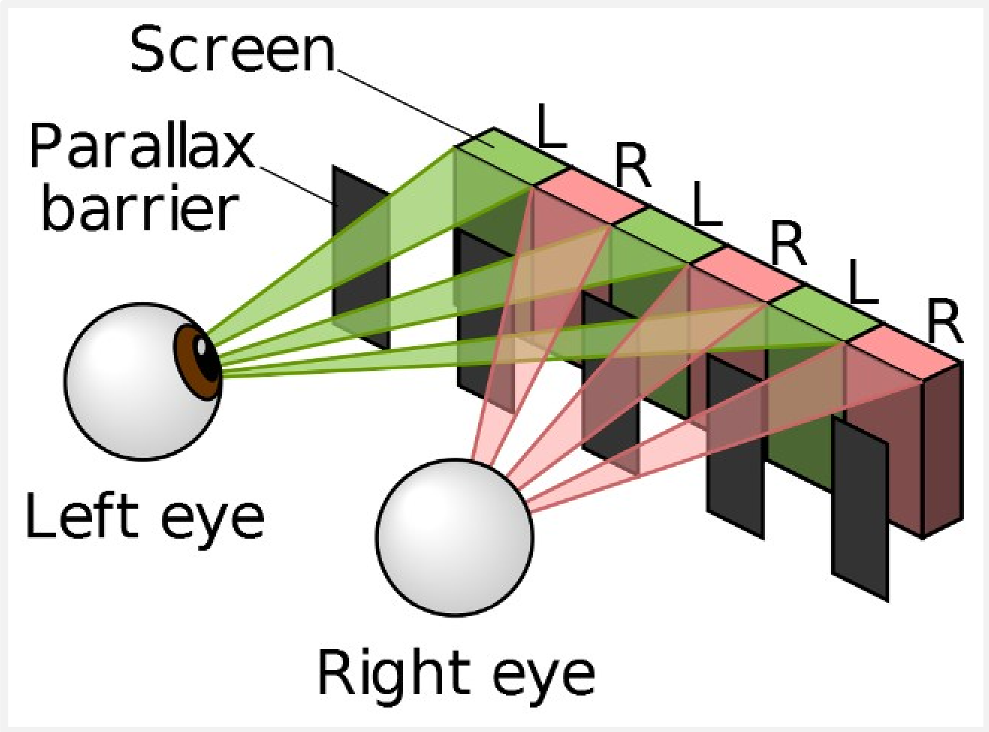
- Additional VR Hardware: Anaglyph Glasses, Parallax Barriers, and Lenticular Lenses
- Topics Related to VR Systems: Serious Games, Real-Time Interactive Simulators

HARDWARE: ANAGLYPH GLASSES

Anaglyphs rely on the encoding of a stereo pair into a single image exploiting color filters, and then using special glasses (**anaglyph glasses**) including the proper color filters to block/enable (decoding) the proper left/right image to the relative eye.

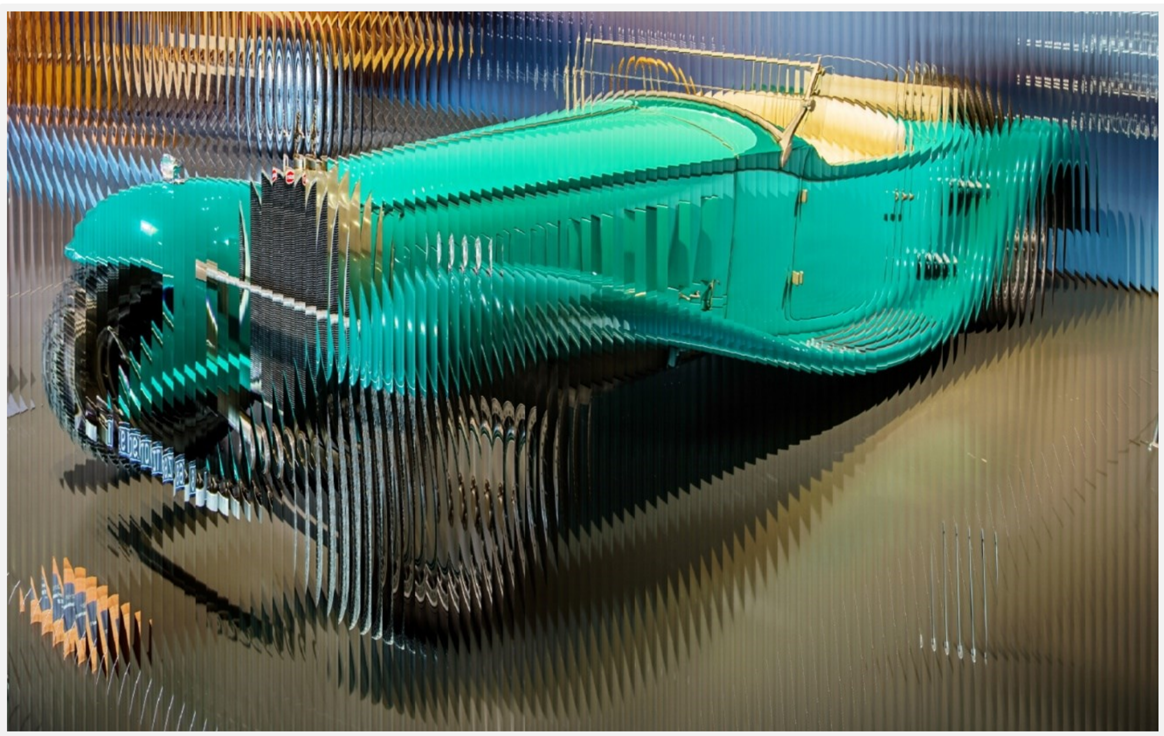
HARDWARE: PARALLAX BARRIER

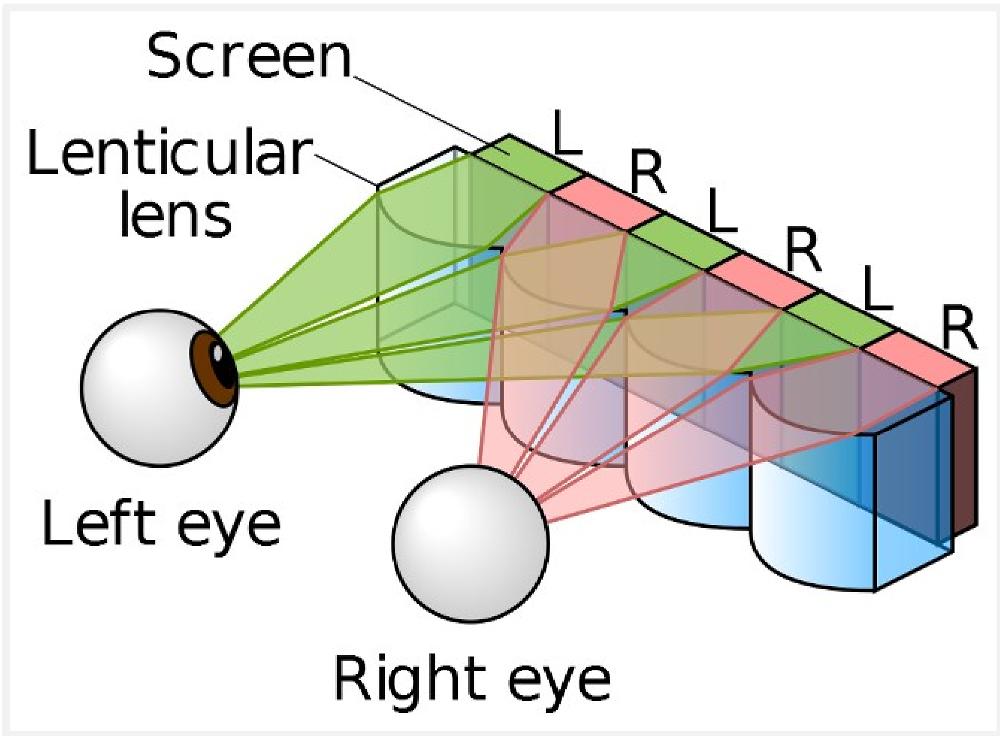
A **parallax barrier** is an autostereoscopic system allowing the visualization of stereo pairs without the need of additional optical devices. The image visualized integrates the stereo pairs interlacing the left and right images, that are delivered them to the proper eye using a semi-transparent barrier exploiting horizontal parallax.



HARDWARE: LENTICULAR LENS

Lenticular lenses are an autostereoscopic system allowing the visualization of stereo pairs without the need of additional optical devices. The image visualized integrates the stereo pairs interlacing the left and right images, that are delivered to the proper eye using a semi-transparent lens sheet to steer the view properly.





SERIOUS GAMES VS INTERACTIVE SIMULATORS

Serious games are games that have an *"...explicit and carefully thought-out educational purpose and are not intended to be played primarily for amusement..."*, but this *"does not mean that serious games are not, or should not be, entertaining..."*

CLARK C. ABT. SERIOUS GAMES. 1970.

Real-time interactive simulators are serious games consisting in 2D/3D graphics-oriented simulations that can realistically replicate real-world events, while providing a high degree of interactivity and *"real-time"* system reactions.

Note: In most cases there is not a clear distinction between a serious game and a real-time interactive simulation. For example: a serious game can integrate (and usually does) elements of real-time interactive simulation, and viceversa.

A SERIOUS GAME EXAMPLE

See: *"Vital Signs: Emergency Department"* medical interactive trainer by BreakAway Games

REAL-TIME INTERACTIVE SIMULATION

A real-time interactive simulation is a digital simulation, usually graphics-oriented, designed to be interactive and performed with discrete-time. In particular:

- These are **graphics-oriented** simulators, including 2D or 3D interactive visualization.
- These systems can **realistically simulate** natural phenomena and real-life events.
- These simulators are **highly interactive**, including complex user interfaces.
- These softwares can react in real-time, so **performance optimization** is critical.
- These simulators integrate elements of: **game development, AI, HCI, networking**, etc.
- These systems can be designed for **single-player or multi-player** environments.

Real-time interactive simulation involves a multidisciplinary approach, drawing from: computer science, computer engineering, mathematics, and physics.

REAL-TIME INTERACTIVE SIMULATION (2): DISCRETE TIME

In a real-time interactive simulator the simulation is performed with discrete-time.

Usually, the simulation discrete-time step is constant, that is: time moves forward in steps of equal duration, and this is commonly known as fixed time-step simulation.

However, in some cases, a variable time-step simulation can also be used.

Fixed time-step simulation is preferrable in some situations, whereas variable time-steps simulations provide better results in others.

Example: In an interactive video game integrating some physics simulation, the graphics visualization runs at a variable time-step (frame rate) while the physics updates run at a fixed time-step (usually ~50 Hz).

Example: In a real-time interactive simulator integrating physics and haptics, the physics updates run at a fixed time-step (usually ~50 Hz) while the haptic rendering runs at a different fixed time-step (usually ≥1 KHz).

REAL-TIME INTERACTIVE SIMULATION (3): ONLINE OR OFFLINE

To solve the mathematical equations at a given time, each simulation variable is solved successively as a function of variables of the previous time-step.

In a discrete-time simulation, the time required to solve all mathematical equations at a give time may be shorter or longer than the duration of the simulation time-step.

If the computing time to solve all mathematical equations (update time) is shorter than the simulation time-step, the simulation is called accelerated simulation (not real-time).

If the computing time to solve all mathematical equations (update time) is longer than the simulation time-step, the simulation is called offline simulation (not real-time).

If the computing time to solve all mathematical equations (update time) equals the simulation time-step, the simulation is called real-time simulation (or online simulation).

REAL-TIME INTERACTIVE SIMULATION (4): MULTITHREADING

In a real-time interactive simulator the simulation is usually performed in a multi-threaded environment (*i.e.*, multiple software modules executed concurrently), allowing the decoupling of simulation tasks with different requirements.

In particular:

- Multithreading allows **optimal use of multi-core CPUs**.
- The **graphics rendering** thread usually runs at a variable frequency (frame-rate, >30 Hz), depending on the content visualized, and that should be constantly maximized.
- The **physics simulation** thread usually runs at a constant frequency (physics updates, >50 Hz), depending on the mathematical equations solved.
- The **collision detection** thread is usually integrated in the physics thread.
- The **haptics feedback** thread usually runs at high constant frequency (>1 kHz), to provide appropriate tactile/force feedback.

REAL-TIME INTERACTIVE SIMULATION (5): -IN-THE-LOOP

The capability to solve simulation equations in real-time enables different types of testing:

- **Hardware-in-the-Loop (HIL):** A testing technique for complex real-time embedded hardware modules, by enabling their interactions with a simulated system.

  For example: An hardware-in-the-loop test can simulate a car engine interacting with the ECU (engine control unit) with real inputs/outputs to enable the ECU testing.

- **Human-in-the-Loop:** A testing technique for complex real-time systems, by using interactive simulation involving always a human user that influences its outcomes.

  For example: A human-in-the-loop test can simulate an airplane cockpit interacting with the pilot with real inputs/outputs to enable the pilot testing/evaluation/training.

- **Software-in-the-Loop (SIL):** A testing technique for software algorithms, by simulating their interaction with a software environment/platform.

  For example: A software-in-the-loop test can simulate a self-driving car interacting with an AI algorithm with real inputs/outputs to enable the testing of its code.