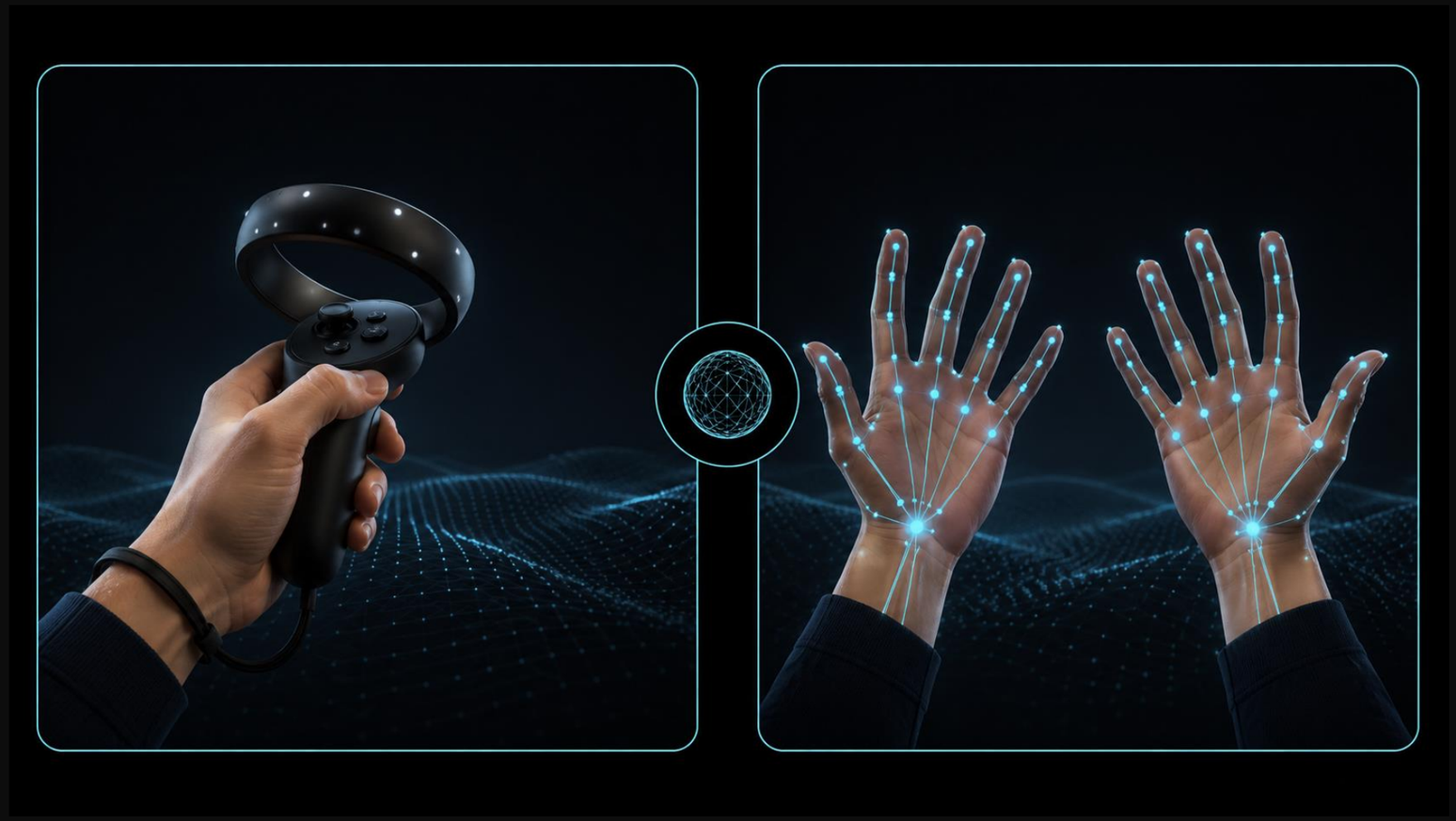
Hand Tracking and Leap Motion

Interaction Modalities in XR

eleonora.chitti@unimi.it
Laboratorio Realtà Virtuale
AA 2025/2025

PART 1

INTERACTIONS IN XR



AR and VR: Interaction Contexts

Different realities imply different interaction constraints

Virtual Reality

Fully simulated environment; strong immersion; interaction is usually spatial and body-centered.

Augmented Reality

Digital information overlaid on the physical world; interaction must respect real-world context.

Mixed Reality

Virtual and real objects coexist and can interact through spatial mapping and tracking.

PART 1

From Traditional Input to Natural Interaction

XR expands the input types

Desktop Input

Mouse, keyboard, gamepad. Precise but indirect.

Touch Input

Direct 2D manipulation. Familiar but surface-bound.

Spatial Input

Controllers, tracked hands, gaze and body movement.

Natural UI

Gestures, voice, and embodied interaction.

Core Interaction Tasks in XR

Most immersive systems combine these four families of actions

Navigation (VR)

Teleportation, continuous locomotion, room-scale movement, world scaling.

Selection

Raycasting, gaze selection, direct touch, proximity-based activation.

Manipulation

Grab, move, rotate, scale, assemble, throw and inspect 3D objects.

System Control

Menus, virtual panels and mode switching (optionally with voice commands shortcuts).

PART 1

VR Controllers

Physical devices that provide tracked input and discrete controls

- Track position and orientation in 3D space.
- Expose buttons, triggers, thumbsticks and capacitive sensors.
- Provide reliable, low-latency input
- Pressure sensors
- Excellent for precision, repeatability and game-like interactions.

Controllers: Strengths and Limitations

A strong engineering solution, but not always the most natural one

Strengths

- Stable tracking and precise input
- Tactile buttons and triggers
- Haptic feedback
- Mature support in Unity and XR toolkits

Limitations

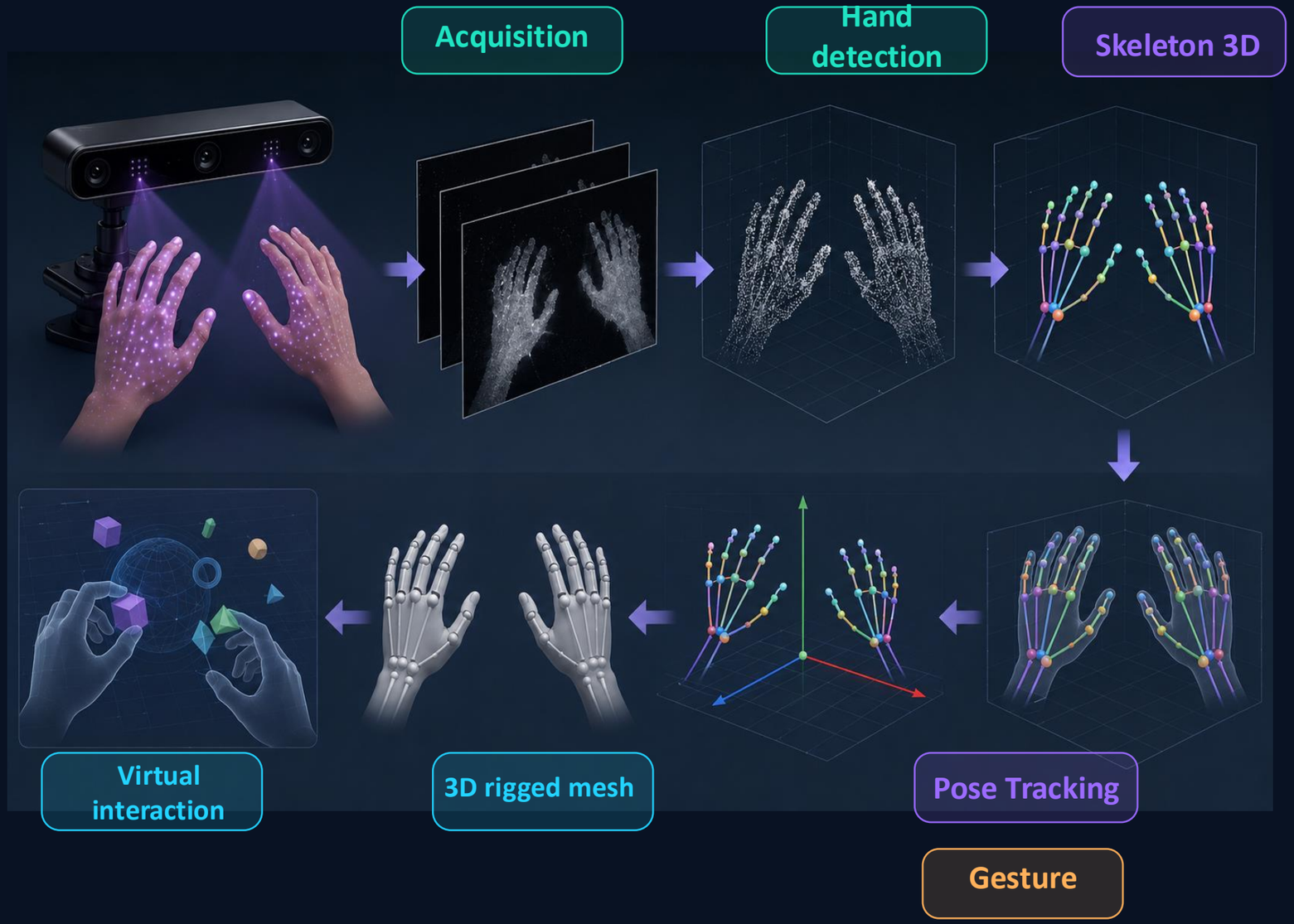
- Requires a physical device
- Button mappings must be learned
- Can reduce body ownership
- Batteries, pairing and hardware dependency

PART 1

Hand Tracking

Using the hands themselves as the input device

- Hand tracking estimates the pose of the palm, fingers and joints.
- It supports natural gestures such as pinch, grab, poke and swipe.
- The design challenge is translating expressive human movement into stable digital commands → possibilty to Record gestures for later recognition.



Gesture Recognition

Gestures are meaningful hand configurations or movements

Static Gestures

Recognized from a pose at a specific moment.

Examples: pinch, open hand, fist, thumbs-up.

Dynamic Gestures

Recognized from motion over time.

Examples: swipe, wave, circular motion, draw-in-air.

Hand Tracking: Strengths and Limitations

A trade-off between naturalness and robustness

Strengths

- No controller required
- High sense of body control (“ownership”)
- Intuitive direct manipulation (Naturalness)
- Strong for training applications (exergames) and for accessibility

Limitations

- Occlusion and tracking loss
- No physical buttons
- Gesture ambiguity and fatigue

COMMON CONTROLLER + HAND TRACKING FEATURES

VR CONTROLLERS

Controller-Based Interaction

- Physical controllers tracked in 6DoF
- Buttons, triggers and joysticks for input
- Precise pointing and selection
- Haptic feedback and vibration

HAND TRACKING (HANDS IN VR)

Natural Hand-Based Interaction

- Hands are tracked in 3D space
- Natural and intuitive interaction
- Gestures, pinches and hand poses
- No physical controllers required

PART 1

Selection Techniques (Controller or Hand Traking)

Choosing objects in 3D is harder than clicking in 2D

Raycasting

A virtual ray is projected from the hand or from the controller. Useful for distant targets.

Direct Touch

The user touches or approaches a virtual object with the hand. Natural but requires precise tracking.

Gaze + Pinch (Hololens)

The eyes or head aim at the target; a pinch confirms selection. Efficient for lightweight AR/VR UI.

3D Manipulation (Controller or Hand Traking)

Once selected, objects can be transformed in space

- Grab: attach an object to one or both hands.
- Move: translate the object through physical hand motion.
- Rotate: use wrist orientation or two-hand relative movement.
- Scale: change size through two-hand distance or explicit handles.
- Inspect: bring objects closer, turn them, explode assemblies or reveal metadata.

https://docs.ultraleap.com/xr-guidelines/index.html

Feedback Channels for Output

Ad ogni input corrisponde un feedback o reazione sincrona del sistema

Visual

Highlight, outline, animation, cursor, hand state.

Audio

Click, spatial cue, collision sound, confirmation tone.

Haptic

Resistance, force feedback, mid-air ultrasound.

IMMERSION, INPUT AND OUTPUT IMPORTANCE

Presence, Immersion, Agency

1 Immersion

Influenced by technical quality of the system: FOV, frame rate, tracking, audio

2 Presence

Psychological perception of “being” in the virtual world

3 Agency

Perception that user actions cause coherent system’s reaction/ feedback in the virtual world

Presence: Embodiment and Naturalness

Interaction quality affects the feeling of “being there”

***Presence**: the psychological feeling of being inside the virtual environment*

- **Embodiment:** the feeling that virtual hands or avatars belong to the user.
- **Naturalness:** the interaction resembles skills already learned in the physical world.

Low latency, stable tracking and clear feedback are essential.

PART 1

Ergonomic Risks both in AR and VR

Natural interaction is not automatically comfortable

Gorilla Arm Effect

Long interactions with raised arms cause fatigue. Avoid forcing users to hold hands in mid-air for too long.

Design Response

Keep gestures short, place UI within comfortable zones, support rest positions, prefer micro-interactions.

Design Guidelines for Hand or Controller-Based XR Interactions

Design for comfort and clarity

- ✓ Use large targets and generous interaction zones (wider than real mesh size).
- ✓ Avoid requiring millimeter-level finger precision.
- ✓ Always visualize the hand/ controller state and selected object state.
- ✓ Prefer reversible actions and clear affordances.
- ✓ Use multimodal feedback: visual + audio + haptic when available.

https://docs.ultraleap.com/xr-guidelines/index.html

PART 1

Current Interaction Patterns

Modern XR often combines multiple modalities

Controllers / Hands

Controllers for precise tasks; hands for natural manipulation.

Hands + Gesture

Hands movements to select or interact; gestures specify target action or parameters selection (e.g. audio on/off).

Hands + Gaze (Hololens)

Targeting with eye/head direction plus pinch for confirmation.

PART 2

LEAP MOTION

Leap Motion

A practical platform for hand tracking

Leap Motion is historically one of the most used optical hand-tracking devices for education and prototyping.

It is developed and maintained by Ultraleap.

It is useful for learning the complete pipeline: sensor, tracking service, SDK, Unity integration and interaction design.

https://docs.ultraleap.com/hand-tracking/getting-started.html

PART 2

Desktop Mode vs HMD Mode

Sensor placement changes the interaction model

Desktop Mode

The sensor is placed on the desk, facing upward. Good for tabletop interaction, data visualization and demos.

HMD Mode

The sensor is mounted on a headset. Good for VR hand presence and direct manipulation.

PART 2

What Is Leap Motion / Ultraleap?

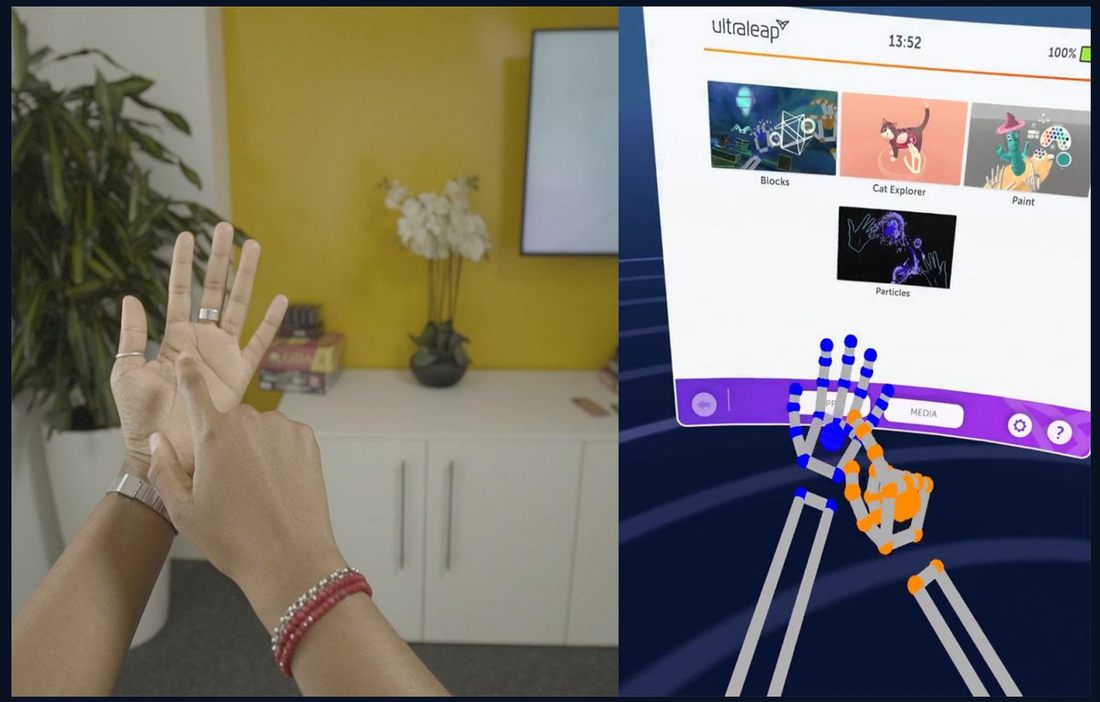
A camera-based hand tracking system for desktop and XR applications

Hardware

Small optical sensor with infrared illumination and cameras that observe hands in front of the device.

Software

Tracking service and SDK estimate hand pose and expose data to applications such as Unity.

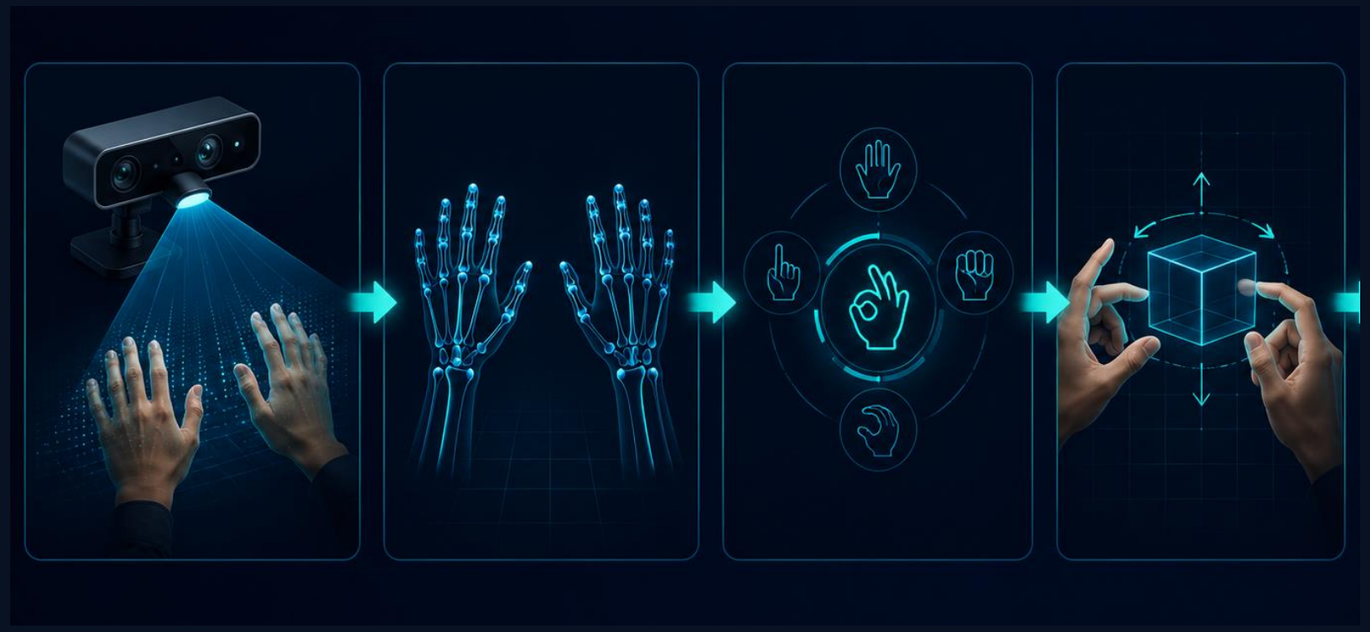


https://docs.ultraleap.com/hand-tracking/index.html

PART 2

How Leap Motion Works

The device observes hands and reconstructs a skeletal model



1. Infrared cameras capture the hands.
2. Computer vision algorithms infer palm, fingers and joints.
3. A 3D hand skeleton is streamed to the application.
4. Unity renders virtual hands (mesh + rig) and uses them for interaction events.

Gestures can be recorded and recognized

(when joints assume specific positions for n secnods → true)

https://docs.ultraleap.com/xr-and-tabletop/xr/index.html

PART 2

Tracking and Common Issues

The sensor only works well when the hands are visible

Good Conditions

Hands in front of the sensor; moderate distance; clear view; stable lighting.

Common Problems

Occlusion, hands too far away, reflective surfaces, unstable USB connection.

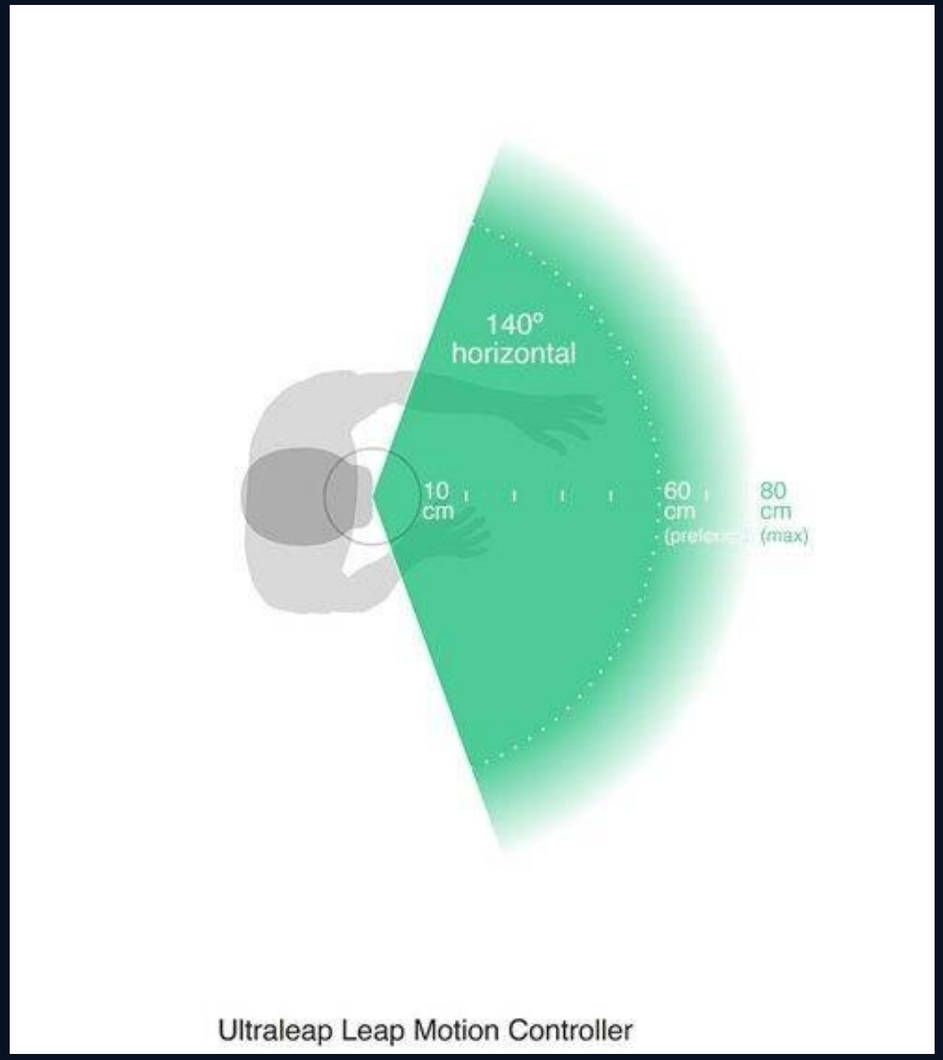
Design Impact

Applications must tolerate temporary tracking loss and avoid fragile gestures.

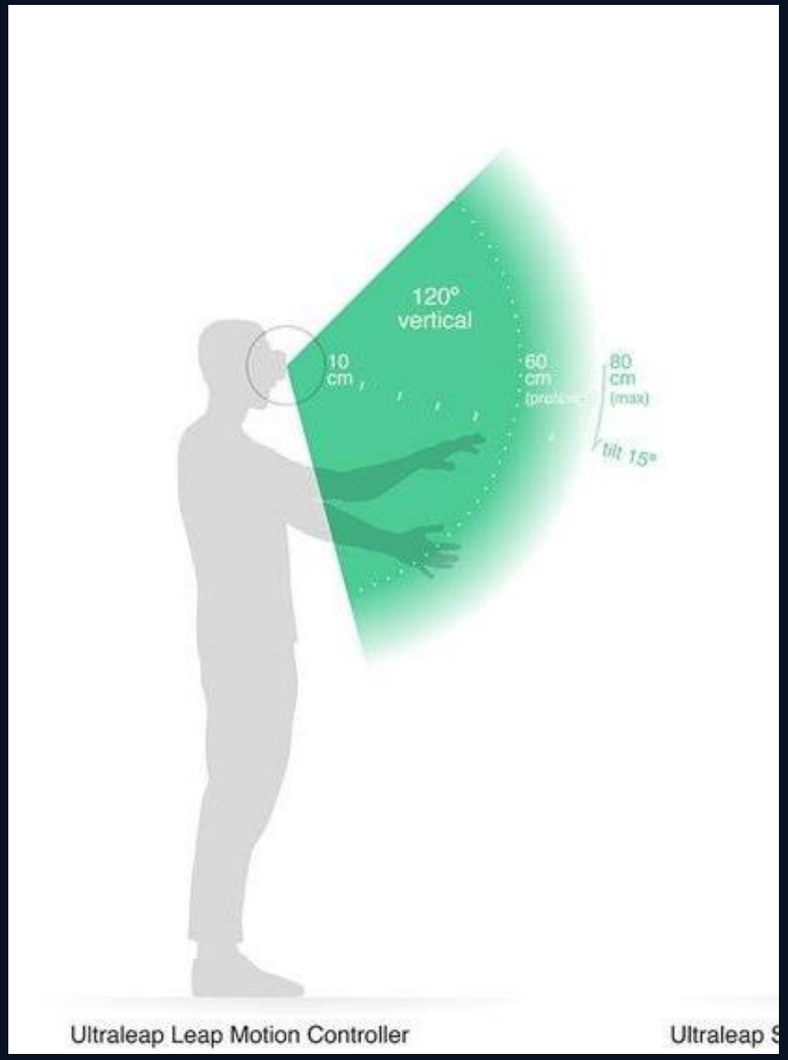
PART 2

Tracking and Common Issues

The sensor only works well when the hands are visible



Ultraleap Leap Motion Controller



Ultraleap Leap Motion Controller

Software Architecture

Tracking Service

Runs in the background and processes sensor data.

SDK / API

Exposes hands, bones, joints, poses and events.

Unity Plugin

Provides prefabs, examples and interaction components.

https://docs.ultraleap.com/xr-and-tabletop/xr/unity/index.html

https://docs.ultraleap.com/hand-tracking/getting-started.html

PART 2

Official Links

Use official sources when preparing the lab environment

Ultraleap Developer Portal: https://developer.ultraleap.com/

Hand Tracking SDK Downloads: https://developer.ultraleap.com/tracking/downloads/

→ Ultraleap Tutorial, XR guidelines and API Documentation: https://docs.ultraleap.com/

Ultraleap in Unity: https://docs.ultraleap.com/xr-and-tabletop/xr/unity/index.html

Unity Plugin on GitHub: https://github.com/ultraleap/UnityPlugin

Unity XR Hands Package: https://docs.unity3d.com/Packages/com.unity.xr.hands@latest

PART 2

Recommended Lab Environment

Prepare these before the practical session

- A computer with Unity Hub and a recent Unity LTS or supported Unity version.
- Ultraleap Tracking Software installed and running.
- Leap Motion Controller / Ultraleap hand tracking camera connected by USB.
- Unity project with Ultraleap Unity Plugin configured.
- A simple test scene with virtual hands and basic interactable objects.

PART 2

Step 1: Install Tracking Software

Start from the runtime before opening Unity

- Go to the Ultraleap Hand Tracking Downloads page and download the tracking software for your operating system: https://www.ultraleap.com/downloads/leap-controller/

- Download Software Hyperion for Leap Motion (1st version)

- Install the drivers and SDK.