Interaction and Virtual Environments

Developing Immersive Applications

Created by: Chek Tien TAN

Learning Objectives:

- differentiate model-based vs image-based methods for creating virtual environments
- classify interaction mechanics along the natural-artificial spectrum and explain how hardware enables each
- understand the concept of embodiment and how it relates to immersion
- apply authenticity and interaction-design considerations to immersive use cases
- implement common interaction mechanics in WebXR using BabylonJS behaviours, actions, and observables

Creating Virtual Environments

Two fundamental approaches to constructing a VE:

Model-Based:

- Build 3D geometry (meshes, materials, lighting) from scratch or from assets
- Full 6-DOF exploration and interaction with objects
- Higher development effort, but supports dynamic, interactive scenes

Image-Based:

- Use captured images (360-degree photos, photospheres, light fields)
- Faster to produce from real-world scenes
- Limited to fixed viewpoints (3-DOF); less interactive

Choosing the Right Approach

Criterion	Model-Based	Image-Based
Exploration freedom	Full 6-DOF	Fixed viewpoints (3-DOF)
Object interaction	Rich (grab, manipulate)	Limited (point-and-click)
Visual realism	Depends on asset quality	Photorealistic capture
Production time	Longer (modelling needed)	Shorter (capture and stitch)
Hardware needed	3D modelling tools	360-degree camera
Best for	Training, games, simulations	Virtual tours, real-world showcase

Hybrid approaches combine captured environments with selective interactive model-based elements.

Image-Based: 360-Degree Video

360-degree capture enables rapid creation of photorealistic VEs from real-world scenes.

Key Characteristics:

- Captured using omnidirectional cameras
- Provides 3-DOF viewing (rotate, no translate)
- Ideal for virtual tours, documentaries, and education
- Can be enriched with interactive hotspots

Trade-off:

Fast production, photorealistic, but limited interactivity compared to model-based approaches.

360-degree video lecture capture

Video: https://youtu.be/2kKcFqSI-Ig

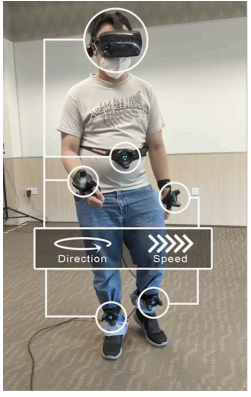
Interaction in Immersive Environments

Interaction is how users communicate intent and receive system feedback.

Key Interaction Modalities in XR:

- **Gaze-based**: user looks at targets to select
- **Controller-based**: buttons, triggers, thumbsticks
- **Hand tracking**: bare-hand gestures, pinch, grab
- **Raycasting**: virtual laser pointer for distant selection

Full-body interaction appears in specialized setups with additional trackers.



Natural-Artificial Interaction Spectrum

Interactions classified by how closely they mirror real-world actions:

Natural Interaction

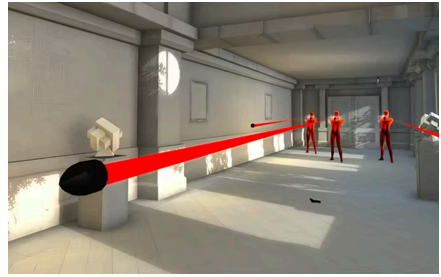
- 1:1 mapping of physical action to virtual effect
- E.g., physically walking, reaching out to grab

Augmented Natural Interaction

- Real physical action with enhanced virtual effect
- E.g., walk-in-place on a slidemill

Artificial (Magical) Interaction

- No real-world analogue; entirely virtual mechanic
- E.g., teleportation, time manipulation in SUPERHOT VR



SUPERHOT VR: artificial mechanics, high immersion. Source: superhotgame.com

Authenticity and Plausibility

Slater's framework for immersion distinguishes two key illusions:

Place Illusion (PI):

The feeling of "being there" in the virtual environment. Driven by sensory fidelity, tracking quality, and field of view.

Plausibility Illusion (Psi):

The feeling that events in the virtual world are really happening. Driven by how credibly the environment responds to user actions. **Design implication**: an interaction does not need to be realistic to feel immersive. It needs to be *plausible* within the context of the virtual world.

Embodiment in VR

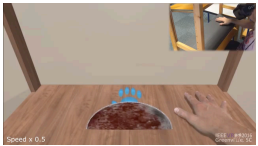
Embodiment is the sense of ownership and agency over a virtual body.

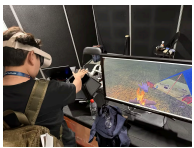
Factors that increase embodiment:

- **Visual appearance**: realistic hand/body representation
- **Visuomotor synchrony**: virtual body moves in sync with physical body
- **First-person perspective**: view from the avatar's position

Why it matters:

- Higher embodiment increases presence and emotional engagement
- Users avoid virtual hazards (e.g., routing hands around a virtual saw blade)
- Partial embodiment (hands only) can still be effective





Video: youtu.be/oE1XanOoRvw

Hardware Enabling Interaction

Different hardware capabilities enable different interaction modalities:

Hardware Feature	Interaction Enabled	Example
Hand tracking (skeletal)	Direct manipulation, pinch	Quest 3 hand tracking
Motion controllers	Button, trigger, thumbstick	Quest Touch controllers
Eye tracking	Gaze selection, foveated input	Quest Pro, Vive Pro Eye
Room-scale tracking	Physical locomotion	Lighthouse, Inside-out
Haptic gloves	Tactile feedback for grabs	SenseGlove, HaptX
Body trackers	Full-body input	Vive Trackers, KatVR

The choice of hardware constrains and enables the interaction design space.

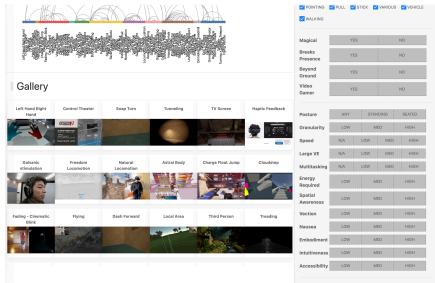
Locomotion in VR

How users move through virtual spaces is a fundamental design challenge.

Five Key Approaches:

- **(1) Teleportation**: point and jump to a target
- **(2) Joystick (continuous)**: smooth thumbstick gliding
- **(3) IMU-based WIP**: walk-in-place using body trackers
- **(4) Real walking**: physical room-scale movement
- **(5) 360 treadmills**: slidemill/omnidirectional WIP

Each approach balances comfort, presence, and hardware requirements differently.



Locomotion Vault: catalog of VR locomotion techniques

Source: `https://locomotionvault.github.io`

Locomotion: Pros and Cons

Approach	Pros	Cons
Teleportation	No motion sickness; simple to implement	Breaks spatial continuity and presence; jarring transitions
Joystick (continuous)	Intuitive for gamers; smooth traversal	High motion sickness risk from sensory mismatch
IMU-based WIP	Authentic walking motion; physically engaging	Requires extra hardware (body trackers); still WIP technology
Real walking	Most natural; best presence and comfort	Limited to room-scale physical space
360 treadmills	Room-scale walking in large virtual spaces	Expensive hardware; still WIP technology

The best locomotion method depends on the application's goals: realism (training), comfort (accessibility), or engagement (games).

360 Treadmills: Bridging the Gap

Omnidirectional treadmills (slidemills) enable natural walking input mapped to large virtual spaces.

How they work:

- Low-friction concave surface
- User wears special shoes; harness keeps them centered
- Foot sliding motion mapped to virtual locomotion

KatVR omnidirectional slidemill

Trade-off:

Combines presence of real walking with unlimited virtual distance, but requires dedicated hardware and space.

Project: https://www.immersification.org/projects/2018/11/08/vrmt.html

Locomotion: Comfort Considerations

Cybersickness from locomotion is primarily caused by **sensory mismatch**: visual motion without corresponding vestibular (inner ear) input.

Design strategies to reduce discomfort:

- Use teleportation or snap turns instead of smooth rotation
- Add a vignette (tunnel vision effect) during continuous movement
- Provide a stable visual reference (e.g., a virtual nose or cockpit frame)
- Allow user control over movement speed and turning rate
- Let users choose their preferred locomotion method

The best locomotion method depends on the application's goals: realism (training), comfort (accessibility), or engagement (games).

Pattern 1: Behaviours

Fastest path for standard drag/grab interactions. Reusable, attach-to-mesh patterns.

```
// Attach drag behavior to any mesh
const dragBehavior = new SixDofDragBehavior();
dragBehavior.rotateWithMotionController = false;
mesh.addBehavior(dragBehavior);
// Also: PointerDragBehavior, FollowBehavior
```

When to use: You need standard grab, drag, or follow behaviour with minimal code.

Docs: BabylonJS Mesh Behaviors

Pattern 2: Actions (ActionManager)

Clean discrete event mapping. Trigger responses on pick, intersect, or keyboard input.

```
// Click handler on a mesh
mesh.actionManager = new ActionManager(scene);
mesh.actionManager.registerAction(
  new ExecuteCodeAction(
    ActionManager.OnPickTrigger,
    () => { console.log("Object clicked!"); }
  )
);
```

When to use: You need a simple click, hover, or intersection handler on an object.

Docs: BabylonJS Actions

Pattern 3: Observables

Fully customizable continuous frame logic. Observer pattern for per-frame monitoring.

```
// Custom per-frame logic
scene.onBeforeRenderObservable.add(() => {
  const dist = Vector3.Distance(
    player.position, target.position
  );
  if (dist < 2.0) { triggerProximityEvent(); }
});
```

When to use: You need custom continuous tracking, distance checks, or per-frame logic that no built-in pattern covers.

Docs: BabylonJS Observables

Implementing Teleportation in WebXR

BabylonJS provides a built-in teleportation feature for WebXR:

```
const teleportation = featureManager.enableFeature(
  WebXRFeatureName.TELEPORTATION, "stable", {
    xrInput: xr.input,
    floorMeshes: [ground],
    timeToTeleport: 2000,
    useMainComponentOnly: true,
  }, true, true
);
teleportation.parabolicRayEnabled = true;
```

Key parameter: `timeToTeleport` sets the hold duration (ms) before teleportation triggers.

GUI Design in Immersive Environments

GUI in VR/AR requires different approaches than traditional 2D interfaces:

Approaches:

- **World-anchored UI**: panels placed in 3D scene; feels natural and spatially grounded
- **Head-locked HUD**: UI follows viewpoint; always visible but can cause discomfort
- **Diegetic UI**: exists within the game world; most immersive (e.g., virtual clipboard, wrist display)

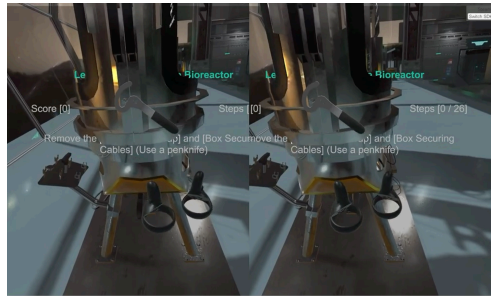
BabylonJS: `GUI3DManager` and `AdvancedDynamicTexture`

Video: https://youtu.be/8gPKw2jrD2c

Case Study: VR Bioreactor Training

When designing interactions for an immersive application, consider:

1. **User's goal?** Training fidelity, entertainment, exploration
2. **Hardware available?** Controllers, hand tracking, room-scale
3. **Natural-artificial spectrum?** Natural for training; magical for games
4. **Comfort budget?** Duration, experience level, motion tolerance
5. **Maintain plausibility?** Consistent responses, even if artificial



VR Bioreactor training application

This VR bioreactor project applies authentic interaction for lab training.

Video: https://youtu.be/zMEs2bhJOMI

Project: https://www.immersification.org/projects/2021/06/13/vrbioreactor.html

Summary

Today we covered:

- Model-based vs image-based approaches to creating virtual environments
- Interaction modalities: gaze, controller, hand tracking, full-body
- The natural-artificial interaction spectrum
- Embodiment and its role in immersion
- Authenticity and plausibility as design principles
- Locomotion techniques and comfort trade-offs
- BabylonJS interaction patterns: behaviours, actions, observables

Next: WEEK10 begins industry-led sessions with our Associate Faculty.

Further Reading

Interaction Theory:

- Marco Gillies: SUPERHOT and the Magic of Interaction
- Slater (2009): Place Illusion and Plausibility

Locomotion Resources:

- Locomotion Vault: catalog of VR locomotion techniques
- Centre for Immersification: VR locomotion methods

Embodiment Research:

- Siju Philip: Partial embodiment with hands in VR
- Disco-VR: Review of haptic gloves

See WEEK09 pre-class material for complete list