① Intro to VR

AR → Augments vision with useful information
↳ still see the real world but with add information overlay on top

VR → Cover entire vision, looking at an entire different place

MR → Mix virtual objects together with real vision allowing both to interact
↳ virtual objects can understand real world eg. virtual dog on a real table

* Milgram - Kishino's Reality - Virtuality Continuum
↳ Slider from Real environment to Virtual environment
3 dimentions ⟷

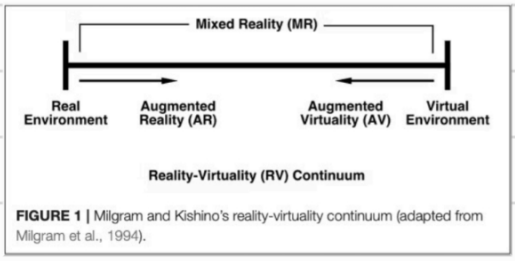


FIGURE 1 | Milgram and Kishino's reality-virtuality continuum (adapted from Milgram et al., 1994).

1) Extent of World Knowledge (EWK)
↳ how much the system understands the real world
↳ how well the real world is replicated

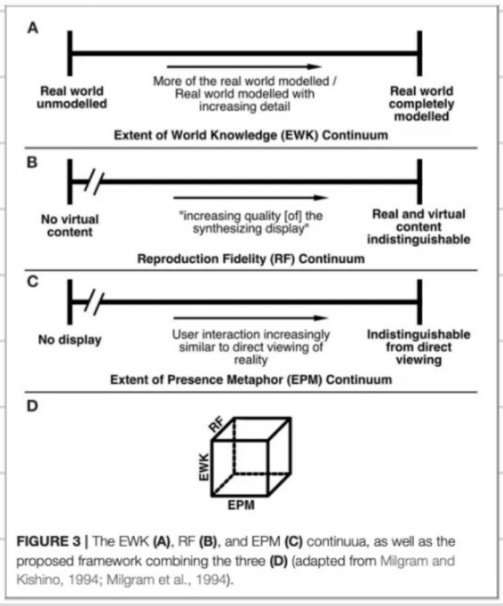


FIGURE 3 | The EWK **(A)**, RF **(B)**, and EPM **(C)** continuua, as well as the proposed framework combining the three **(D)** (adapted from Milgram and Kishino, 1994; Milgram et al., 1994).

2a) Reproduction Fidelity (RF)
↳ how realistic the assets are
↳ system perspective

3a) Extent of Presence metaphore (EPM)
↳ how interaction affords realism
↳ how present is the system to the user (system perspective)
↳ how realistic interactions are

* Newer Version:

2b) Immersion
↳ combined reproduction fiedelity & Extent of presence
↳ how realistic assets and interactions are
↳ system perspective

3b) Coherence
↳ how important it is to represent user
↳ whether the user feels his actions are real
↳ actions simar to real world
↳ coherent with what the user expects

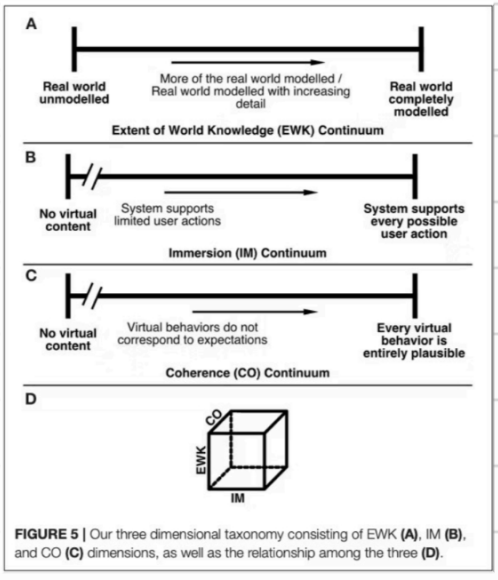


FIGURE 5 | Our three dimensional taxonomy consisting of EWK **(A)**, IM **(B)**, and CO **(C)** dimensions, as well as the relationship among the three **(D)**.

(2) Evaluating Immersive Experiences

Test Driven Development → evaluation at the start

↳ Create test cases first → add code to satisfy test → test → refactor code

Emersion → deep mental involvement

1) Emersion as system properties
   - ↳ software & hardware
   - ↳ specification of the system that produces immersive experiences specs sheet
     - ↳ eg. - 1800 x 1920 pixels per eye, seperate LCD with local dimming
       - 106 degrees horizontal and 96 degrees veritical FOV
       - eye / face motion sensors, 3 mic array
       - interpupillary distance adjustment mechanic
   - ↳ technical qualities of system that help facilitate immersion

2) Immersion as User Experiences
   1) Presence
      - ↳ How much one feels like being transported to another space
      - ↳ feeling of being there
      - ↳ Dimentions
        1) degree of environmental interactions
        2) perceived fidelity and realism of the simulated environment
        3) differentiate physical presence and social presence
           - physical: physically relocated to virtual space
           - social: sence of being around other virtual beings
      - ↳ Individuals modulate the feeling of presence
        - ↳ openness and extroverts experience higher presence
      - ↳ Measuring by:
        1) Subjective data (most common)
           - ↳ self reporting questionairs and interviews
             - ↳ I group Presence Questionair (ipq)

           (physical existence) 1) Spatial Presence – sense of being physically in the VE

           (how captivated) 2) Involvement – measuring the involvement & attention devoted

           (how real it feels) 3) Experienced Realism – measuring the subjective experience of realism

        2) Objective data
        3) Mix of both

2) Flow

- Being in the zone
- Mental state of extream engagement (person loose self consciousness)
- High concentration yet effortless and spontaneous performance of an activity (happyness)
- 8 Dimentions
  1) Sence of clear goals and immediate feed back
  2) Level of challenge that matches skill
  3) Complete consentration on task (oblivious to interactions)
  4) Loss of self consciousness (loose awarness of self & surroundings)
  5) Sense of control (feeling of total control of actions & desisions)
  6) Effortlessness (ease of performing difficult / tiring task)
  7) Transformation of time (feeling time passes fast)
  8) Autotelic Experience [intrinsically motivated purely by activity) - not money
- meusured by validated questionair instruments
  - Flow state scale 2
  - Flow short scale - frequently sampling experiences during activity
    1) Overall flow
    2) Absorption by activity
    3) Fluency of performance

3) Cybersickness

- symptoms of sickness due to cyber activities VR/AR
  - nausea, dizzyness, disorientation, physical discomforts
    - eye ← Oculomoto, Disorientation, Blured vision, Eyestrain, Giddyness, Vertigo
- Meusure by
  - Validated questionairs
    - Simulator Sickness Questionair (SSQ) - global standard for comporison /research
    - Virtual Reality Sickness Questionair (VRSQ) _ for proper design knowledge
    - Cyber Sickness Questionair (CSQ)
  - cybersickness may enhance immersion
  - Higher Visiual Vestibular conflict causes cybersickness
    - difference between visiual and sensory (motion, position, orientation) -> match real world actions

↳ Immersion as experiential perspective
  ↳ Higher spacial presence
  ↳ Higher place illusion
  ↳ Lower cyber sickness

* Affordance
↳ HCI concept
↳ Provides a useful conceptual design tool to relate system properties to user experiences
  ↳ eg. norman's door : metal plate → push handle → pull knob → turn
↳ Relationship between properties of an object and the capabilities of an agent
  ↳ how the object can be use intuitively
↳ Create objects interations that are natural to the real world

* VR user experience for walking
1) Joy stick
  ↳ simple but has large cybersickness
2) Teleportation
  ↳ no cybersickness but lacks desired immersion
3) Walking in place : mechanical locomotion platform
  ↳ questionable usability and not easily accessible
4) Walking in place : Tracker based

	fatigue	effortless	cybersickness	
a) Head bob	8	5	2	
b) Arm swing	0	8	7	→ don't align with visual
c) Leg lift	7	1	0	
d) Full body	7	0	2	

③ Development Primer

(4) Development Tools

Common Tools

1) Unity
   - ↳ Popular for games
   - ↳ Public company → for profits
   - ↳ Professional, polished with alot of features
   - ↳ Visual editor for modifying game objects usin ECS + C# scripting
   - ↳ Easy to create high quality simulation
   - ↳ Covers many common platform – windows/mac/android/web
   - ↳ Has project templates AR/VR templates
     - ↳ set up components and settings
   - ↳ Can connect common XR headsets to test

2) Unreal Engine
   - ↳ Similar to unity
   - ↳ massive feature heavy software for creating real time 3D applications
   - ↳ Private Company (epic games) huge MNC
   - ↳ Editor for ECS and visual scripting language (blueprints) / C++ scripting
   - ↳ Prefered for high quality graphics

3) A-Frame
   - ↳ Web based tool – only for web applications
     - ↳ provided as a javascript library
   - ↳ for new VR developers

cons
   - ↳ limited immersion by web technologies system
     - ↳ improving as web GPU replacing webGL
   - ↳ conforms to webXR standards

pros
   - ↳ good accessibility – any one with web browser can use
   - ↳ open-sourced
     - ↳ can view and change anything to suit needs
     - ↳ good for learning and researchers
   - ↳ JS library creates costom HTML tags < a-scene > < a-box position "1,0,1" >
   - ↳ simple declarative style of programing (intuitive)

4) Babylon JS
↳ Java script library
↳ 3D engine based on webGL & JavaScript
↳ alot of supporting web tools - playground / inspector
↳ better techniques and performance for rendering & physics
↳ large and active community
↳ connected to other open sourced web initiave - react / angular
↳ fully supports webXR
↳ code in TS - strongly typed version of JS

5) Cospaces
↳ for beginers
↳ limited features

* Open standards:
1) WebXR
↳ set of open APIs
↳ standardise the way XR app are created on the web
2) OpenXR
↳ C++ library
↳ native deployment version of webXR
3) OpenCL

* Tool Choice considerations:
1) Cost ⇒ open source → free
2) Stability ⇒ well developed tools
3) Customizability and extensibility ⇒ options / features of tools
4) Community support
5) Learning opportunities ⇒ tool align with learning objective

* Reason for webXR and BabylonJs
↳ facilitate learning & Well supported community
↳ Accessibility for users (web-based, no hardware needed)
↳ webXR - longevity to prototypes accessibility to both users and developers
↳ Control technical details (nuanced technologies)
↳ Contribute to XR-author research project

(5) Hardware & Software Components

* Common XR Devices

AR: Magic Leap , snap AR specs , smart phone , smart glasses

MR: Microsoft Hololens 2 , Meta Quest Pro

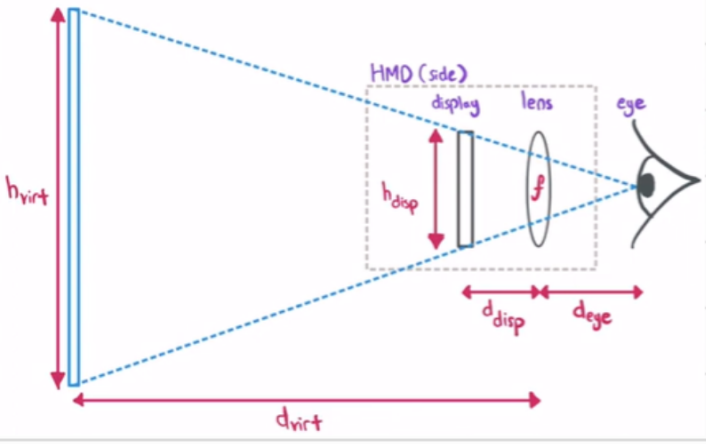
VR: Meta Quest 2, HTC Vive pro2 , Apple vision pro

- ↳ PCVR → head gear connected by a cable to powerful PC + external sensors
- ↳ Standalone → (AIO) (all in one) no cables and inside out tracking

* Hardware

- ↳ Display screen
- ↳ 2 magnifyer glasses
- ↳ sensors e.g. motion tracking
- ↳ many camers
- ↳ battery , speakers , CPU , GPU
- ↳ hand controlers with motion tracking sensors
- ↳ similar to smartphone but with magnifyer lens

Image Formation Process



d_{eye} → eye relif eye to lens

d_{disp} → display to lens

d_{virt} → image to lens

h_{disp} → height of display

h_{virt} → height of virtual image

f → focal length of lens

Gausian thin lens formula

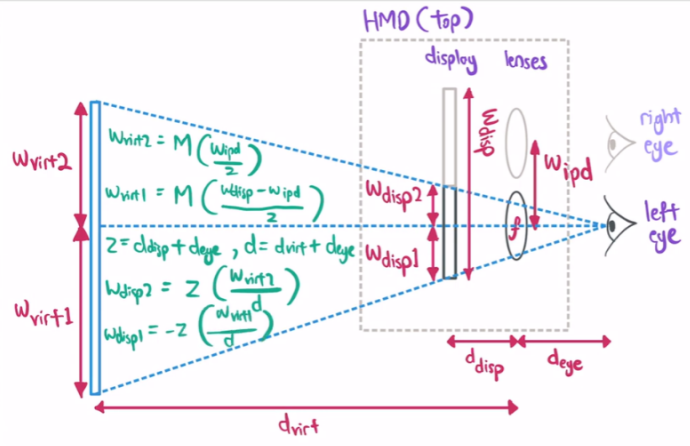
$$\frac{1}{d_{virt}} + \frac{1}{d_{disp}} = \frac{1}{f}$$

$$\frac{1}{d_{disp}} = \frac{1}{f} - \frac{1}{d_{virt}}$$

$$d_{virt} = \left| \frac{1}{\frac{1}{f} - \frac{1}{d_{disp}}} \right|$$

$$\text{Magnification } M = \frac{f}{f - d_{disp}}$$

$$h_{virt} = M \times h_{disp}$$



W_{disp} → with of whole display

W_{isp1} → Temple half of display

W_{isp2} → Nasal half of display

W_{ipd} → Distance between center of 2 lens

W_{virt} → Width of whole virtual image

W_{virt1} → Temple half of virtual image

W_{virt2} → Nasal half of virtual image

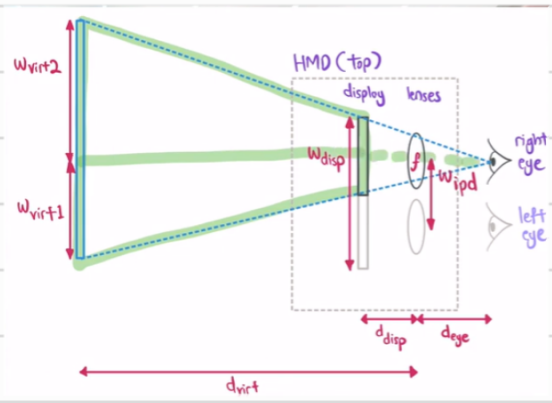
$$W_{virt2} = M\left(\frac{W_{ipd}}{2}\right)$$

$$W_{virt1} = M\left(\frac{W_{disp} - W_{ipd}}{2}\right)$$

$$Z = d_{disp} + d_{eye} \qquad d = d_{virt} + d_{eye}$$

$$W_{disp2} = Z\left(\frac{W_{virt2}}{d}\right)$$

$$W_{disp1} = Z\left(\frac{W_{virt1}}{d}\right)$$



View frustrum

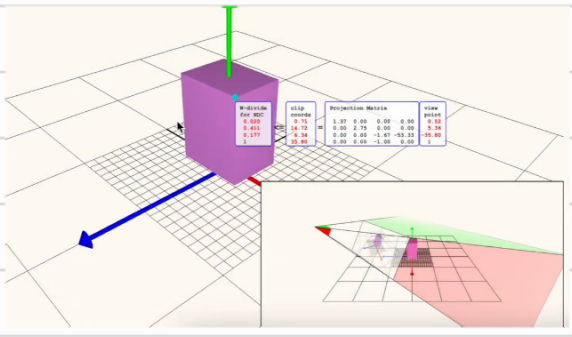
↳ define what is visible

↳ vertically symetric

↳ horizontally asymetric

↳ different image for left & right eye

↳ form stereo image → give perception of 3D

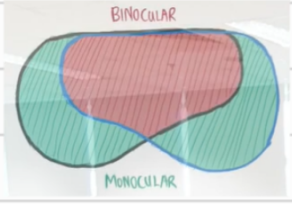


HMD key dimentions

1) focal length
2) IPD
3) d_{disp} – screen-lens distance
4) d_{eye} – eye relief – lens-eye distance

↳ generate projection matrix for left and right view frustrum

↳ near & far plane, height, left and right widths



- Binocular FOV → combined/overlapping FOV
- Monocular FOV → single non overlaping FOV

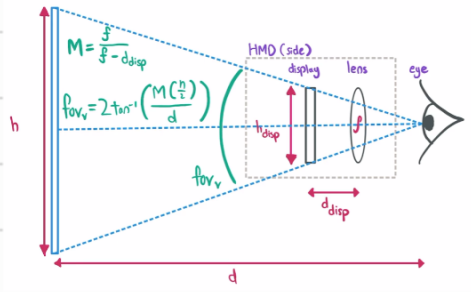
* Field of view FOV

↳ angle between top & botton or left & right

- Virtical:

$$M = \frac{f}{f - d_{disp}}$$

$$fov_v = 2 \tan^{-1}\left(\frac{M\left(\frac{h}{2}\right)}{d}\right)$$

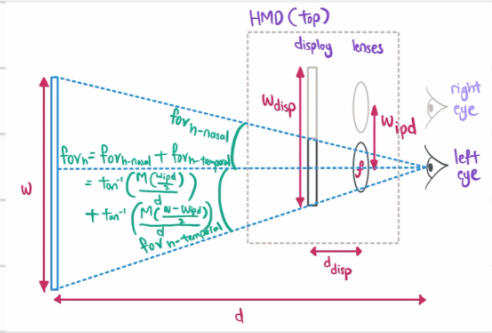


- Horizontal

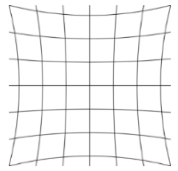
$$fov_h = fov_{h\text{-}nasal} + fov_{h\text{-}temple}$$

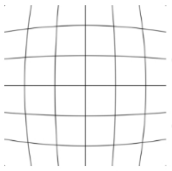
$$fov_{h\text{-}nasal} = \tan^{-1}\left(\frac{M\left(\frac{w_{ipd}}{2}\right)}{d}\right)$$

$$fov_{h\text{-}temporal} = \tan^{-1}\left(\frac{M\left(\frac{w_{disp} - w_{ipd}}{2}\right)}{d}\right)$$



* Lens Distorsion





↳ Natural distortions by magnifier lens

↳ Distortion correction algorithm to counter

↳ further point is from the center more it needs to be shifted for correction

↳ Chromatic aberrations → color distortions need to be corrected too

↳ color artifacts due to different refraction of diff wavelength of light by lens

* Other hardware

1) Additional sensors / trackers on other parts of the body

↳ translate motion from other parts of the body into virtual world

↳ better immersion → lesser visual vestibular confict → lesser cybersickness

2) Eye tracking

↳ eye gaze interactions

↳ foveated rendering → Reduce image quality in peripheral vision

→ Reduce rendering workload

→ Better depth of field effect (blur not focused objects)

* Software components / systems

1) Rendering — drawing all graphis on screen

2) Physics — math need to implement behaviors of objects to simulate real world

3) Input — process inputs based on available hardware

4) Audio — loading and decoding audio files, managing multiple audio sources, volume

5) AI — automated game object behaviors (pathfinding, decision trees, state machines)

6) Networking — connect to other users

↳ can be implemented using existing library and framework

↳ neen architecture to organise code on top of systems → Entity Component System

(6) Creating Virtual Environments

* Model based approach
  - ↳ need to model real world by artist
  - ↳ can interact with objects → full interactive implementations
  - ↳ hand made 3D models using 3D modeling tools
  - ↳ require deep technical art experties
* Image based approach
  - ↳ 360 photos / videos
  - ↳ more accessible to untrained creators
  - ↳ limited to static surroundings
  - ↳ sometimes 3D reconstruction used to bridge towards model based
  - ↳ difficult to seperate objects to interact
  - ↳ more detailed image (high quality)
  - ↳ Only navigating (pre set spots)
  - ↳ some pre determined interactions
  - ↳ easy / fast to capture (less resources)

(8) Interactions

1) View point control
- ↳ dynamic control of users view
- ↳ Related to cyber sickness
  - ↳ higher view point control → lesser cybersickness
    - ↳ better feeling / more accurate "looking around"
- ↳ Passive interaction
- ↳ AR/MR
  - ↳ real world → base environment
  - ↳ user can look around real world
  - ↳ eg. camera feed /transparent lens (best)
- ↳ VR
  - ↳ track head at high resolution
    - ↳ indiscernible experience with looking around real world
- ↳ Smart phones
  - ↳ have inertial measurement units (IMU)
    - ↳ small electronic — accelerometers, gyroscopes & magnetometers
    - ↳ track motion
    - ↳ better in VR (due to purpose driven development)
- ↳ Desk top
  - ↳ keyboard /joy stick eg. for social metaverse application
    - ↳ not as immersive

Highest Fidelity ⟷ lowest

See through lens (AR/MR), VR IMUs, smartphone IMUs, keyboard /joystick

2) Hand Gestures

↳ Active interaction

a) Haptic Gloves

↳ translate finger position to virtual experiences

↳ use electronic actuators to provide tactile feedback (vibration)

↳ e.g. Tech glove Dk1 (from B haptics)

↳ mostly experimental or prototypes

↳ low accessibility

b) Finger tracking based on Computer Vision

↳ meta Quest, hololens, apple vision pro

↳ no haptics

↳ long way to go for tracking speed, robustness and reliability

c) Motion track controllers

↳ used as proxies for hands

↳ Most VR come with controller

e.g. Valve index controller → have sensing of individual finger touches (best)

d) Desktop keyboard

↳ Reduced interaction fidelity → affects immersion

↳ Partial embodiment

Highest Fidelity ⟷ Lowest

using actual hands , haptic gloves , finger tracking , controlers keyboard

3) Body gestures

↳ Embodiment

↳ Feeling of being in control of a virtual representation of self

↳ full embodiment

↳ Need track body movement

a) IMU trackers on body

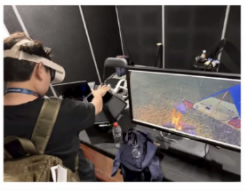
b) 360 treadmills

c) haptic suits

↳ Various use case track different body parts

↳ e.g. cycling → leg trackers

↳ No desktop replacement



Embodiment

- perception that a virtual body is one's own, often correlated to Presence
- improve tracking fidelity
- implement multimodal sensory feedback: visual, auditory, haptic, etc.
- implement personalization: e.g., Meta's mirror

Types:

1) Passive interaction
   - ↳ happens in the background
   - ↳ does not afford conscious effort
     - e.g. viewpoint control

2) Active interaction
   - ↳ needs to be consciously triggered by user
     - e.g. hand / body gestures → pick up virtual objects

* Interaction Authenticity
- ↳ Consideration of wether the desired experience should be natural or artificial

a) Natural Interactions
- ↳ interaction similar to real world
- ↳ used for simulation application
- ↳ might be limited by resoures e.g. Controlen

b) Artificial Interaction
- ↳ Artificial magical interaction
  - ↳ interaction impossible in the real world
  - ↳ e.g. teleport, time travel
  - ↳ not realism but plausibility illusion (belivable)
- ↳ Argumented versions of natural interactions
  - ↳ perform exagerated version of familar action (run fast / high jump)
    - ↳ eg. games half-life alix → gloves to pick up far away objects

* GUI Interaction
- ↳ Graphical user interface
- ↳ e.g. Heads up display HUD
- ↳ Meta Quest
  - ↳ Navigate via viewpoint contol & hand gestures
    - ↳ controllers as proxies for hands / finger tracking
    - ↳ Artifical augmentation of natural pointing
      - ↳ extens a pointer laser ray from finger to menu (ray-cast)
        - ↳ real world finger pointing and laser extention

Gaze control

↳ Passive control (active interaction for viewpoint control)

↳ extend ray from center of view point

↳ stare at button to click

↳ bad for virtual keyboards

* Locomotion Interactions

↳ Joy stick control } artificial magical

↳ Teleportation }

↳ Walking in place

↳ Mechanical natural

↳ Head bob } semi natural

↳ Arm/leg trackers (IMU sensors) }

(9) Implementing Interactions

* npm run build

↳ for distribution

↳ allow others to run

↳ files in disc folder are runable independently

* npm run serve

↳ easily look at changes in web page

↳ not saved build on system

Implementing Interaction

1) Behavior → Predefined, reusable interaction without custom code

↳ Common interactions like dragging, scaling, following

2) Action manager → Define property changes triggered by pre-defined events

↳ Custom interaction parameters (duration, condition, triggers)

3) Observables → notifyObservers

↳ subscribe and receive notifications

↳ Fully customizable interactions

4) Coroutine