Computergrafik

Matthias Zwicker
Universität Bern
Herbst 2016

Today

- Virtual reality

Introduction

- “a computer technology that replicates an environment, real or imagined, and simulates a user's physical presence”, including interaction with the environment
  https://en.wikipedia.org/wiki/Virtual_reality
- Presence: feeling of being there

History

1950s
Sensorama
https://en.wikipedia.org/wiki/Sensorama

1980s

1990s: Cave
https://en.wikipedia.org/wiki/Cave_automatic_virtual_environment

2000s: HTC Vive, Oculus
Google Cardboard

https://en.wikipedia.org/wiki/Virtual_reality

Applications

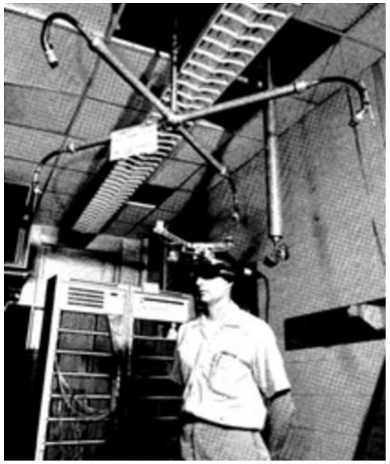
- Education, training (medical, military, etc.)
- Entertainment (games, film, theme parks, concerts, theater, etc.)
- Design (consumer products, engineering, architecture, city planning)
- Virtual heritage, archeology
- Shopping

Why is VR hard?

- Input:
  - Body pose of user (hands, head, etc.); current status of virtual world
- Output:
  - Rendering of virtual world, other sensory outputs (audio, haptics); status of virtual world in next time step
- "Motion-to-photon" latency (input-to-output latency) must be < 20ms
  - Otherwise, "VR sickness"
- Challenging to obtain in a system with many components
  http://oculusrift-blog.com/john-carmacks-message-of-latency/
- Requirements
  - Fast 3D motion tracking
  - Fast rendering, GPUs
  - High-resolution displays

First head mounted display

- With mechanical head tracking (Ivan Sutherland, 1968)
  https://en.wikipedia.org/wiki/Ivan_Sutherland
  http://www.informit.com/articles/article.aspx?p=2516729&seqNum=2
- See-through ("augmented reality")



Consumer VR system (2016)

Head mounted display, tracked (HMD)

3D tracking hardware

Hand-held controller (tracked)

Sound input/output

GPU

Technical challenges

- Head-mounted displays
  - Resolution
  - Binocular depth perception, also called stereopsis (vergence-accommodation conflict)
- Integration of additional input and output devices
  - 3D tracking of head, hands, body
  - 3D audio
  - Haptics (sense of touch)

Head mounted display

Pre-distorted binocular stereo images

http://www.osvr.org/hardware.html

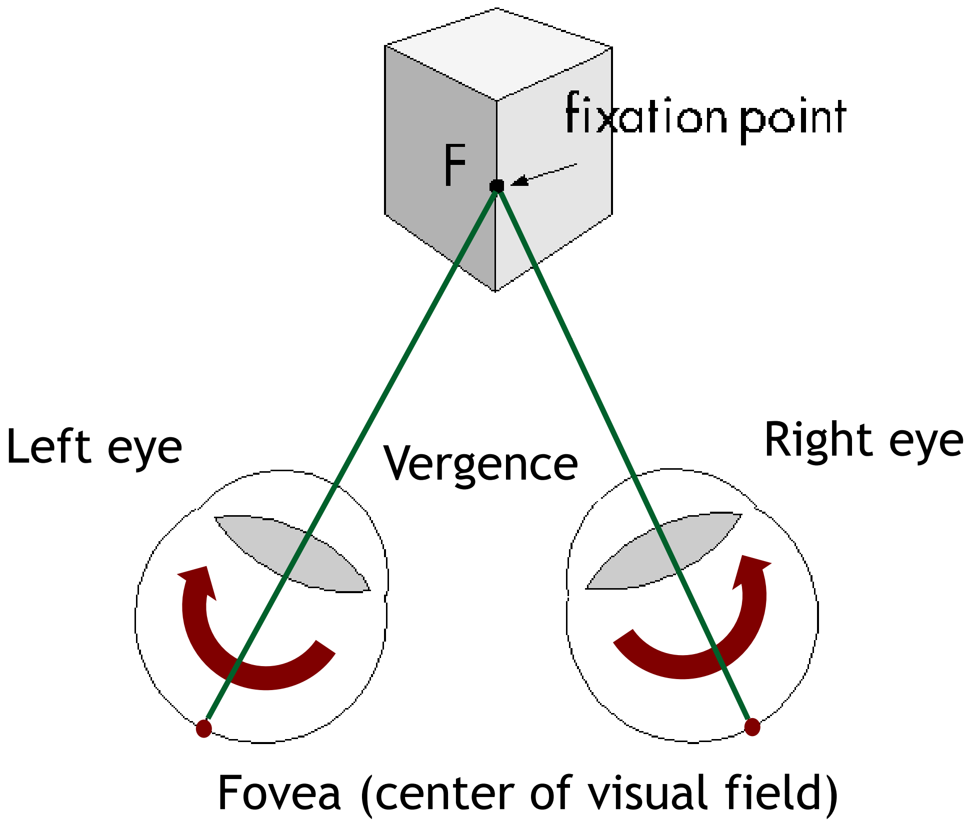
Shows two separate, slightly different images to each eye

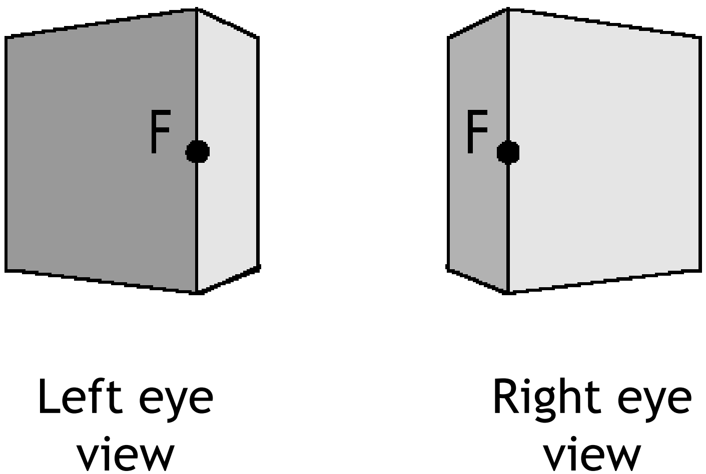
Stereopsis

- Perception of depth (3D structure) through binocular vision
- Binocular depth cues
  https://en.wikipedia.org/wiki/Depth_perception#Binocular_cues
  - Vergence
  - Disparity
- Allow distance estimation based on binocular vision

Vergence

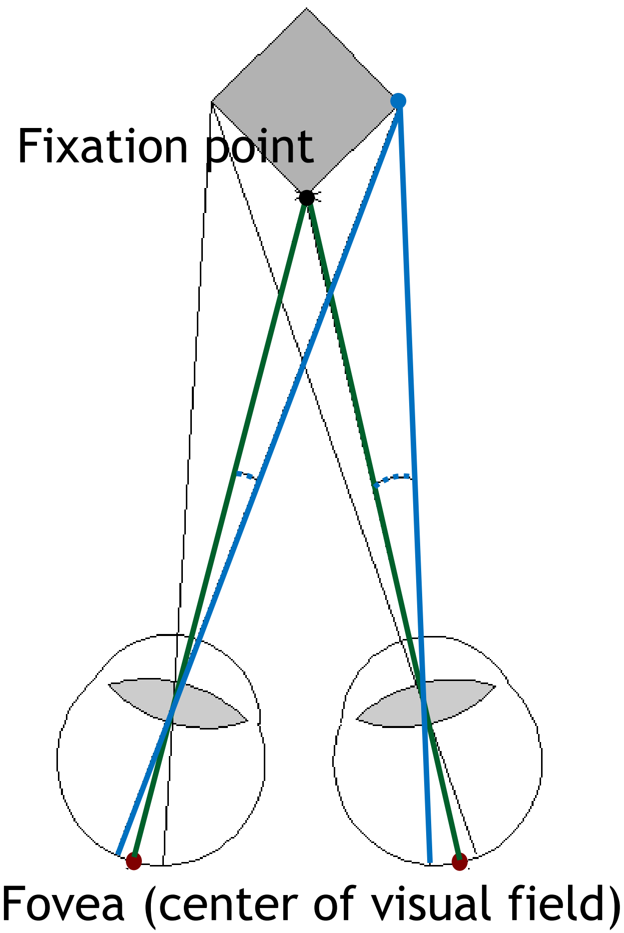
- Both eyes rotate (“vergence”), such that fixation point is in center of visual field (fovea) for both eyes

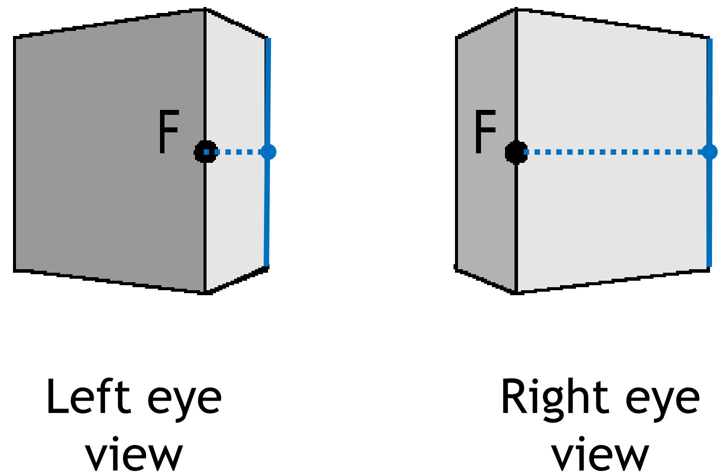




Binocular disparity

- (Angular) disparity: difference in image location (angle) of object seen by left and right eye



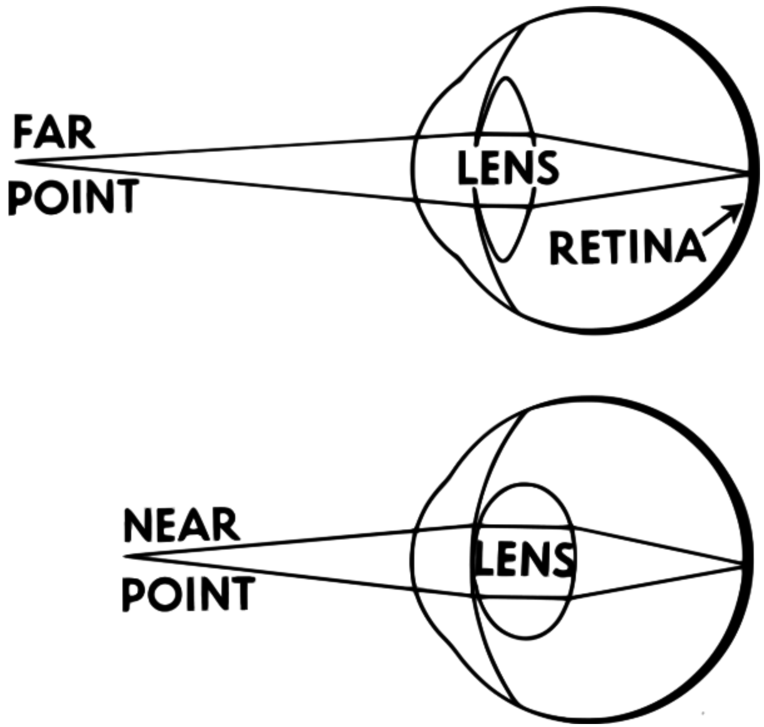


Disparity: difference between and

Accommodation

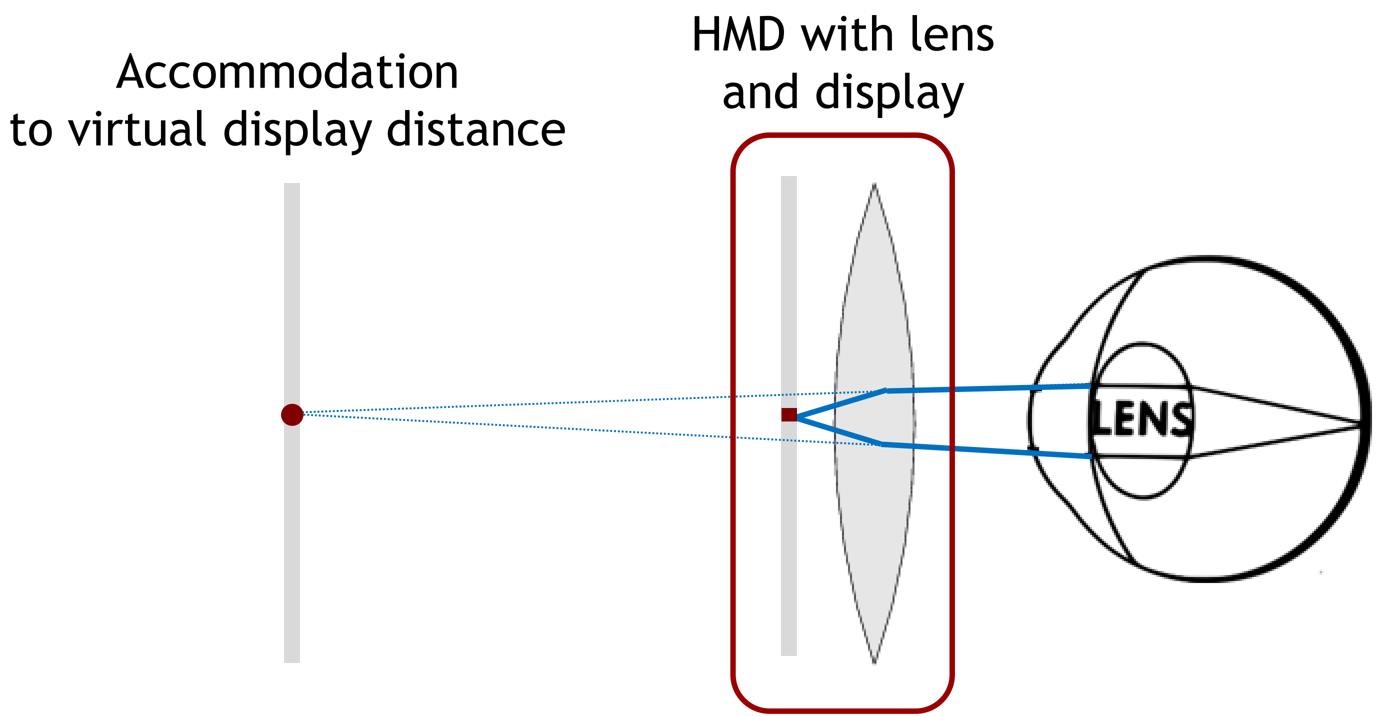
https://en.wikipedia.org/wiki/Accommodation_(eye)

- Lens in eye changes optical power (by deforming) to focus at certain distance



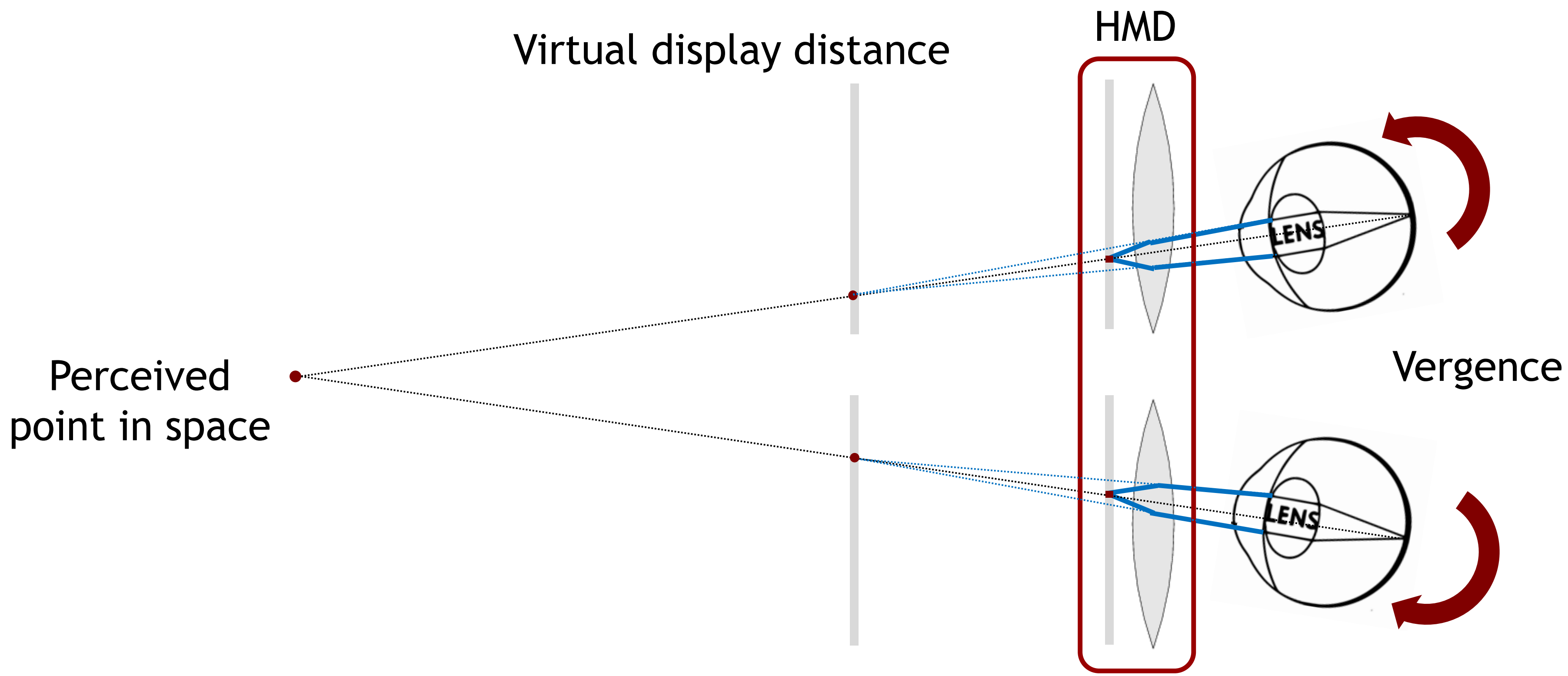
Accommoation

- Head mounted display includes additional lens to allow focus (accommodation) at a comfortable distance



Vergence-accommodation conflict

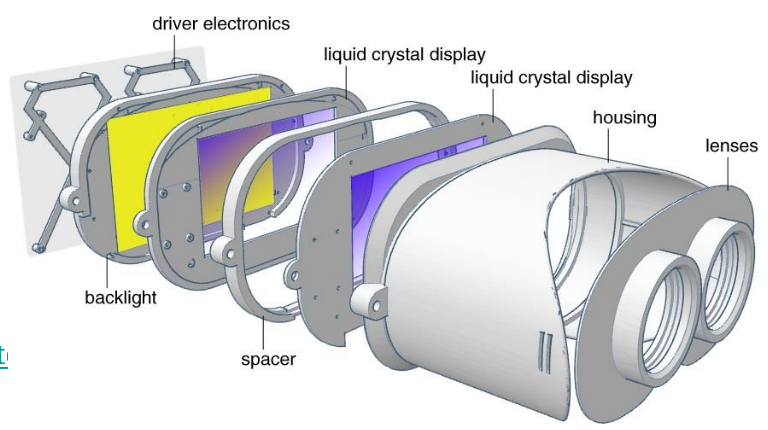
- Vergence: to perceived point in space
- Accommodation: to virtual display surface



Vergence accommodation conflict

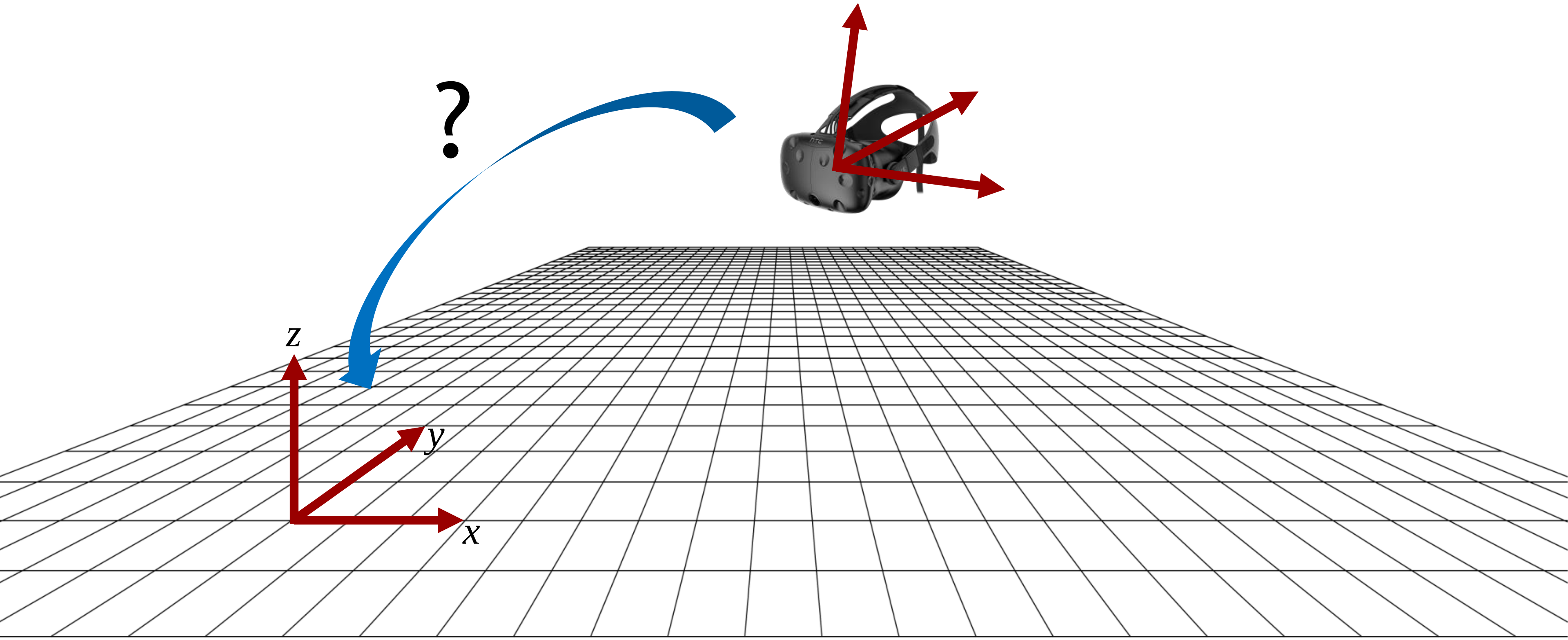
- Can lead to “VR sickness”
- Currently unsolved
- Related: retinal blur
  - Out-of-focus blur of objects at distances away from fixation point
  - Perceivable in real world, but not replicated in current HMDs
- Research ideas exist
  - “Light field stereoscope”

    http://www.computationalimaging.org/publications/the-light-field-st



3D tracking

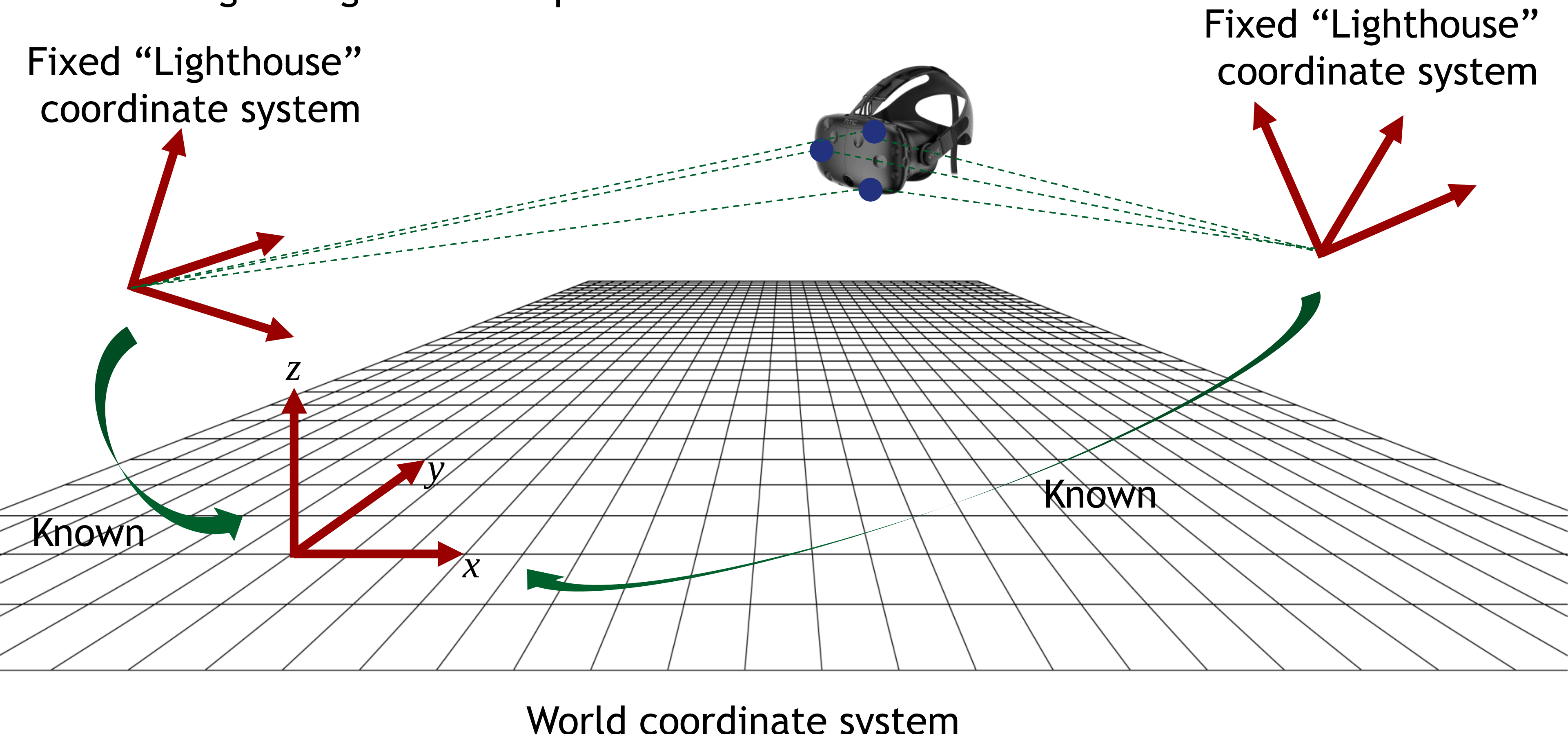
- Goal: determine position and orientation of real-world objects relative to a reference coordinate system



World coordinate system

Triangulation

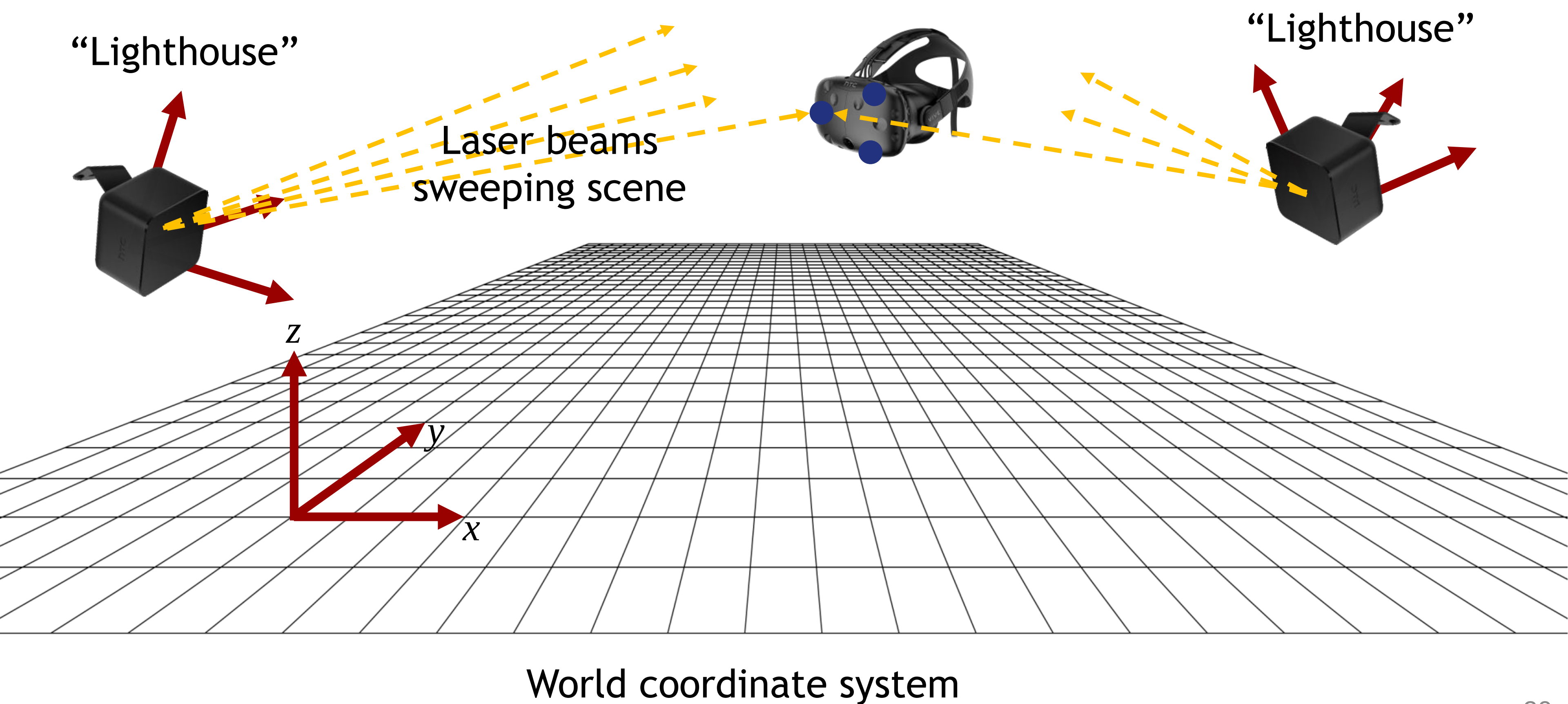
- Assume known “lighthouse” coordinate systems
- Measure directions from lighthouses to tracking point
- Intersection gives 3D position (triangulation) of tracking point relative to lighthouses
- Triangulating 3 or more points determines orientation



HTC Vive/SteamVR tracking tech

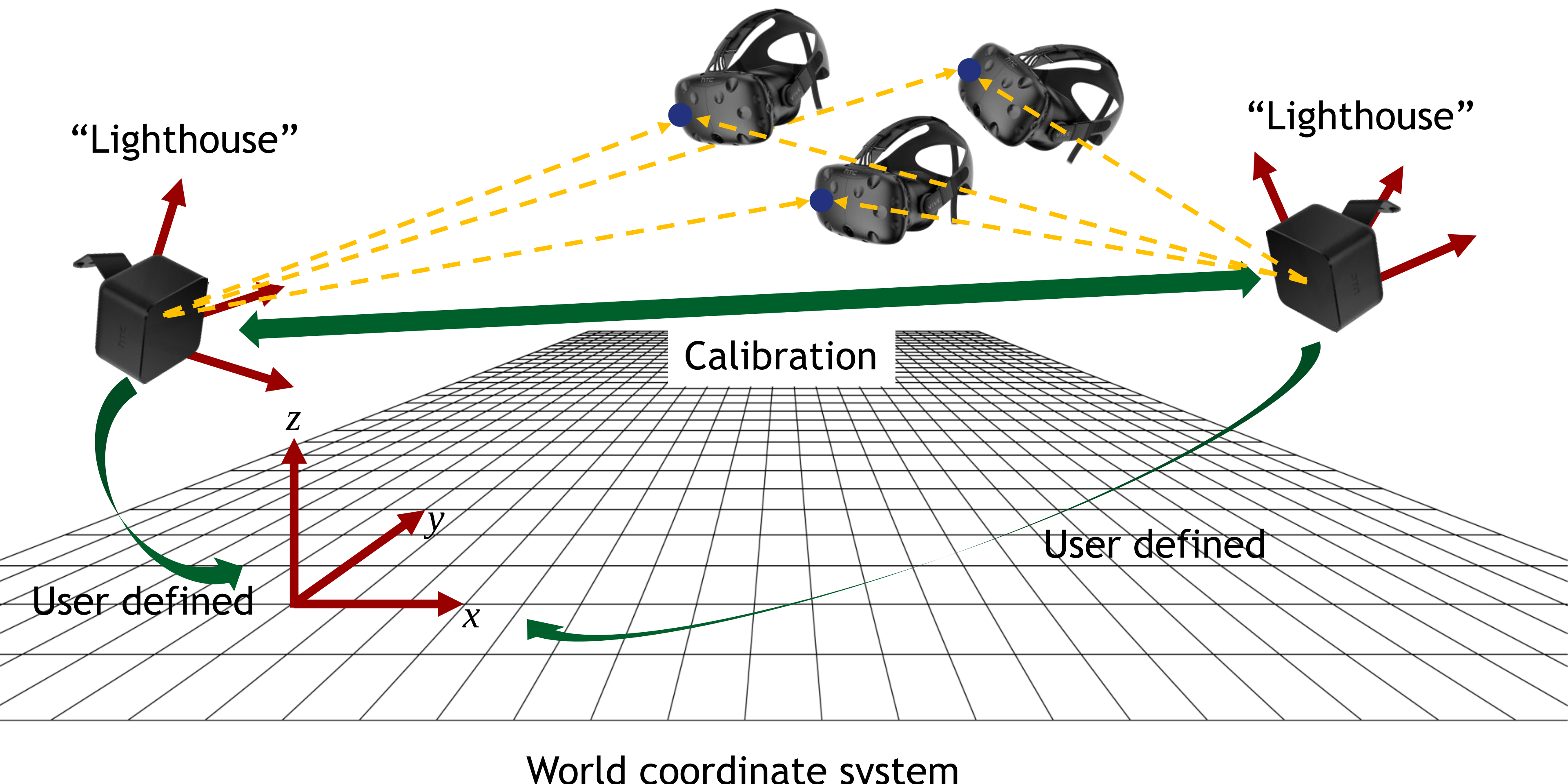
https://partner.steamgames.com/vrtracking/

- Lighthouses emit optical timing signals and sweeps room (left-right and top-bottom) with laser beam, 60 times per second
- Tracking point has light sensor
- When tracking point detects laser beam, can recover laser beam direction from timing information, then perform triangulation from two beams



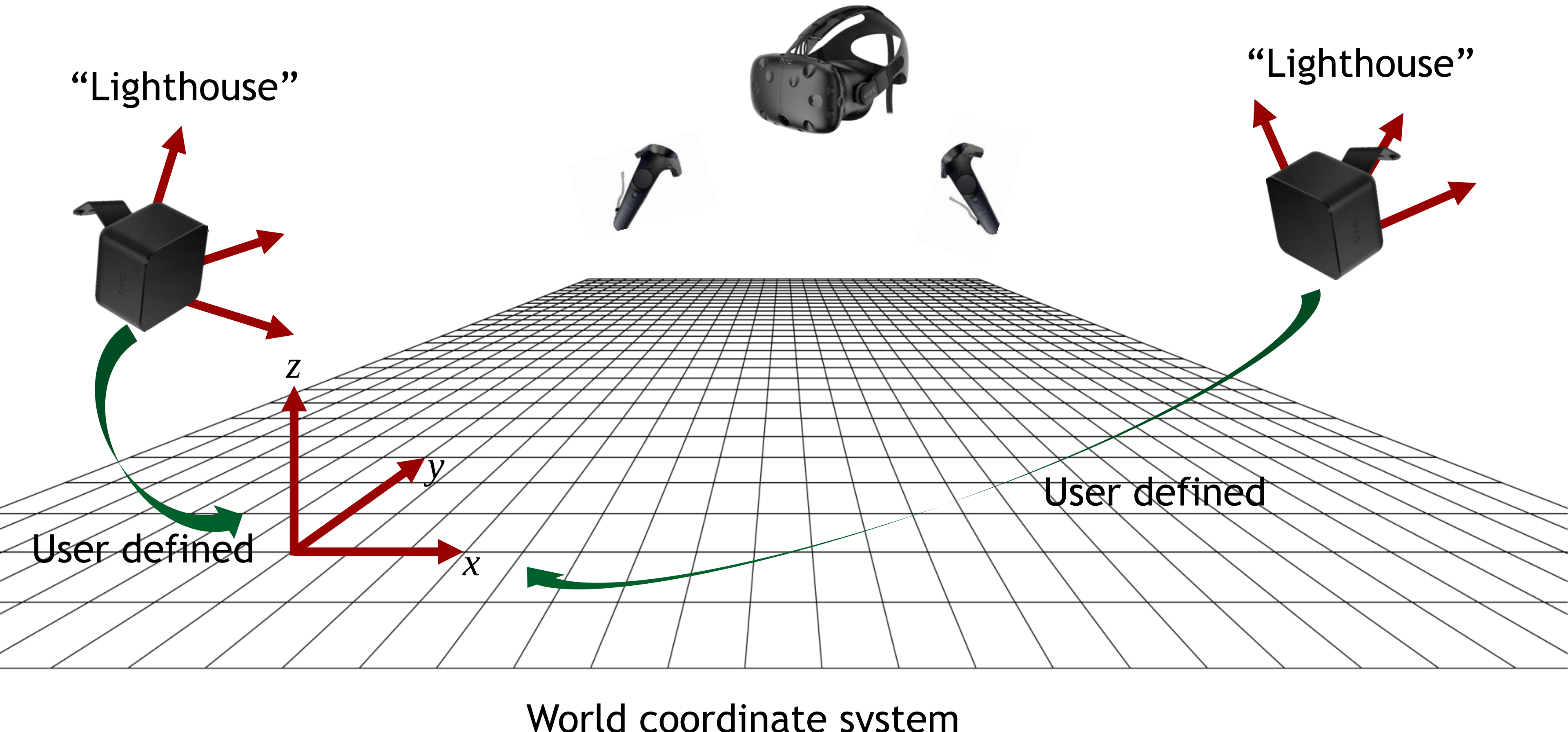
HTC Vive/SteamVR tracking tech

- Calibration: establish relation between lighthouses and world coordinates
- Moving tracking objects around and recording multiple beam directions from both lighthouses allows to reconstruct relation of light houses
- Set world coordinates at center of user space, user defined



HTC Vive/SteamVR tracking tech

- Track several objects, like hand controllers
- 1000Hz refresh rate (using additional inertial sensors in tracked devices)



3D tracking demo

- Juggling

- Many other 3D tracking technologies exist
- Often via triangulation

VR in practice

- GPU-based rendering (almost) as usual (OpenGL, DirectX)
- Interface to VR hardware via OpenVR
  https://en.wikipedia.org/wiki/OpenVR

OpenVR

- Main functionality
  - Provide 3D tracking information (hand controllers, head, etc.)
  - Provide VR camera parameters (two eyes)
  - Display rendered images on HMD
- Source and documentation
  https://github.com/ValveSoftware/openvr
- Java binding via JNA
  https://en.wikipedia.org/wiki/Java_Native_Access

OpenVR

- Basic setup

While running

    `WaitGetPoses` *to get 3D tracking information*

    Render left camera using OpenGL

    `Submit` *to compositor*

    Render right camera using OpenGL

    `Submit` *to compositor*

    Update application logic

`OpenVR API calls in red`

OpenGL rendering

- Using camera and projection matrices provided by OpenVR
- Render into OpenGL frame buffer object
  https://www.opengl.org/wiki/Framebuffer_Object
  - Will not be displayed directly on screen
- Pass rendered image to OpenVR compositor via `submit`
  - OpenVR automatically performs lens pre-distortion
- Optionally, mirror rendered image to screen (OpenGL `glBlitFrameBuffer`)

OpenVR in jrtr

- Encapsulated in `jrtr.OpenVRRenderPanel`, `jrtr.OpenVRRenderContext`
- Some messiness because of JNA access to native functions and data structures
- Base code provided for VR programming exercise
  - Virtual squash game
  - Demo during exercise session