unity

Unite
India
2017

Postmortem: Hephaestus

A VR based Architectural Drawing Tool

Hello!

Chandan Singh

Head – VR

Bonnie Mathew

Lead – Engineering

What is Hephaestus?

- Hephaestus - the Greek god of craftsmanship
- A POC to test validity of VR for creating architectural designs
- 5 person dev team, 3 months dev time

Why did we build Hephaestus?

Why did we build Hephaestus?

- Existing CAD tools aren’t good for visualization
- Tons of errors, tools don’t help here much
- Collaboration is a pain

Hephaestus Design Goals

Ease of use & visualization

- It should be easier to use than existing Architectural CAD tools
- VR is naturally suited for tasks like these
- Learning to use such tools is easier in VR
- Scale - true to the real world

No errors, clean data

- Every object has volume
- Objects can’t intersect
- Objects have to be on a surface or attached to a surface

Compatibility with the ecosystem

- Data export to 2D and 3D formats
- Store data in a clean format that makes conversion possible

Demo

What went right?

Code Architecture

- Proper design was done before the implementation.
- Design fit well with Unity’s component based design model
    - Composition over inheritance
- Extensively used design patterns - Strategy, Memento, State Pattern, Composite, Factory etc.

Session Data Management

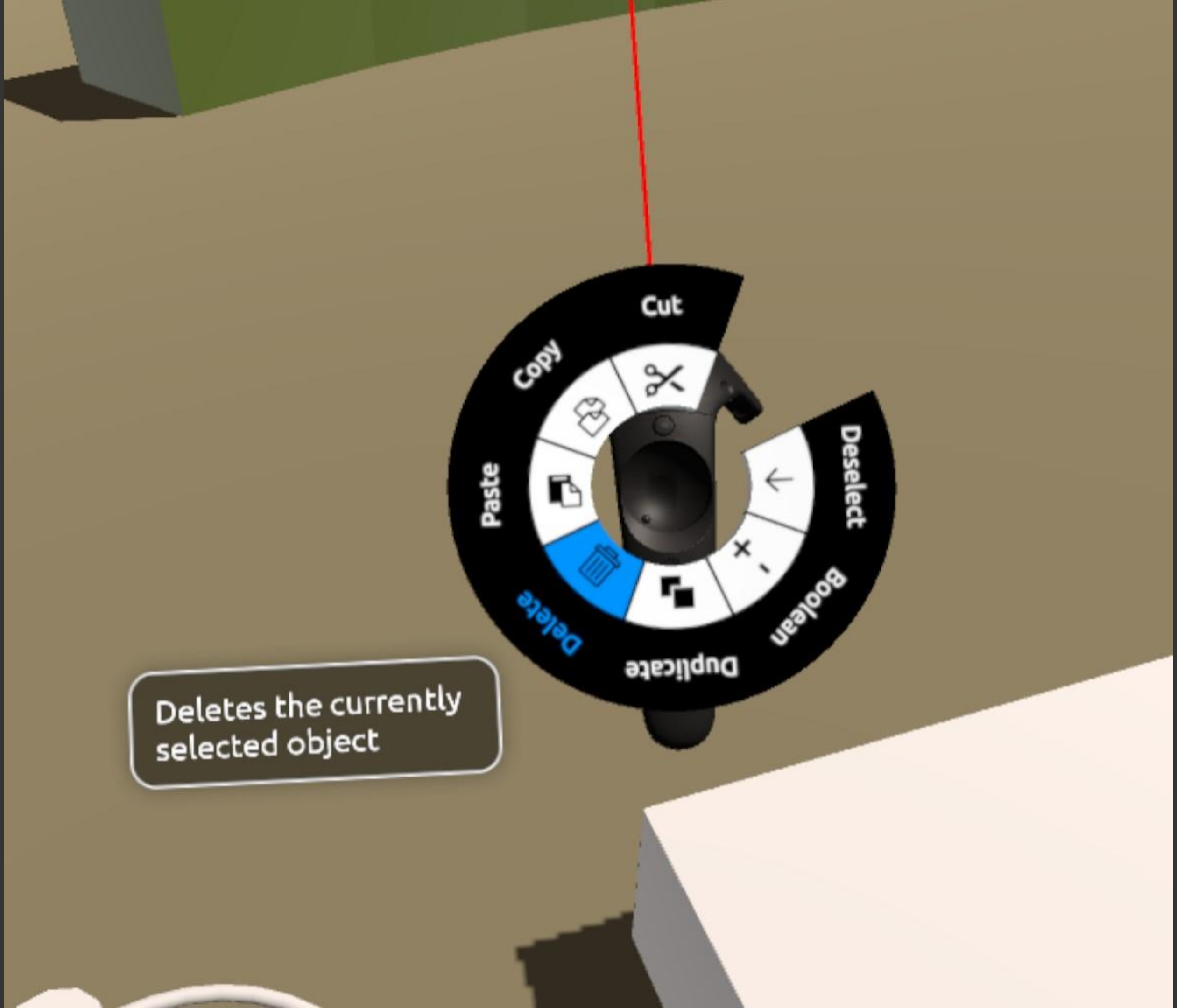
- Save the session and reload the session next time
- Serializer/Deserializer included in every .NET/Mono installation
- Runtime import from disk and export to disk happens parallely in different thread
  - Carefully designed the data structures and importer so as to have minimum dependency on the main thread.

Drawing and design tools

- All actions in the VR workspace are done via tools
- Robust design to add various drawing tools
  - ToolsManager to handle various tools
  - DrawTools , AnnotateTools, MeasureTools, Object Creation Tools, Object Manipulation Tools
- Tools gets input events from the InputHandling module in the form of OnPressed(), OnReleased(), OnPressMove() etc
- Excellent procedural mesh support by Unity
  - ~ 90% procedural meshes
  - Had to re-calculate bounding meshes in order to support accurate physics behaviour

Action History

- Each action performed by tool is saved as a state in stack
- Allows for non-destructive workflow
- Allows for Undo & Redo



Deletes the currently selected object

Error proofing

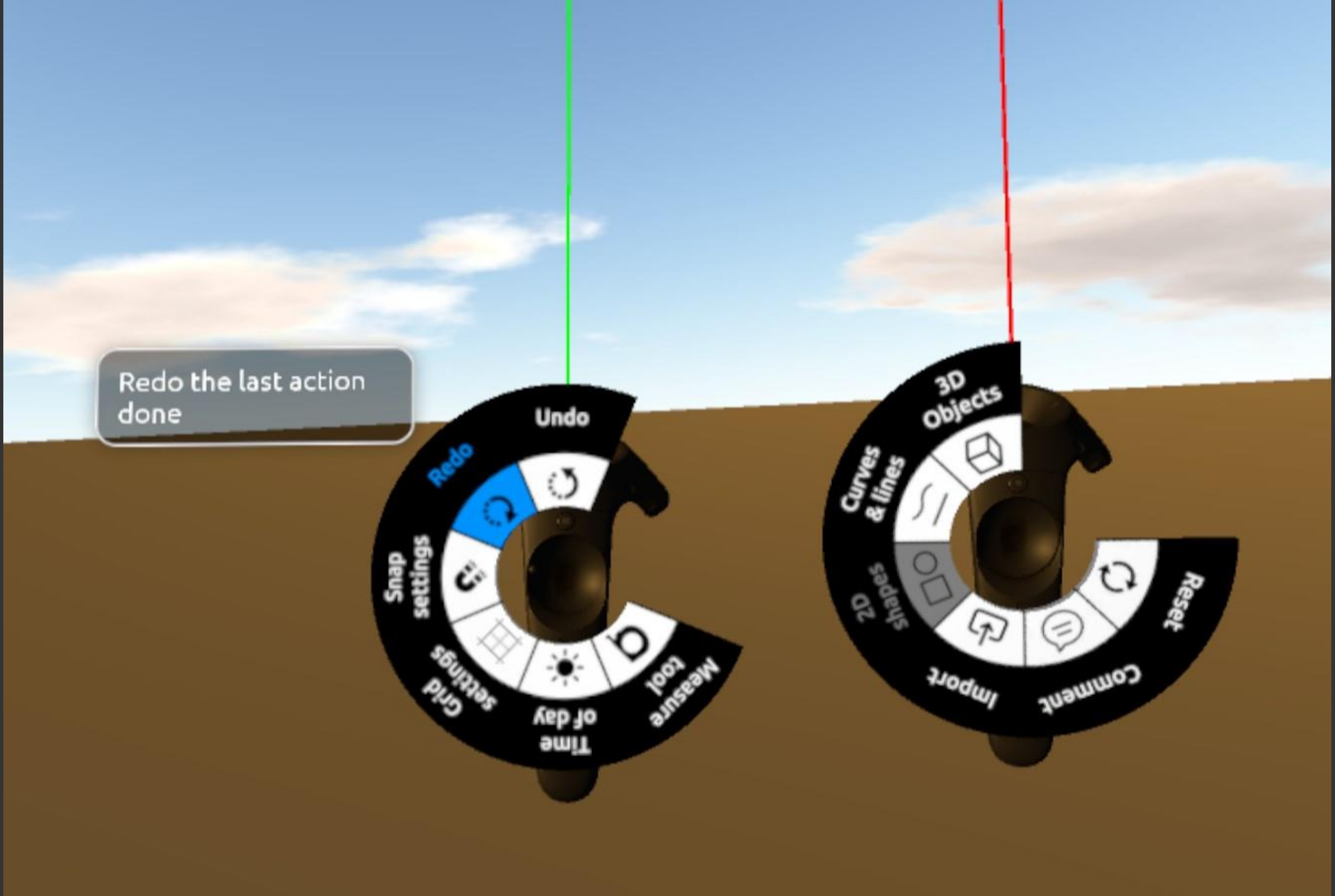
- Whenever possible, we didn’t allow the user to make mistakes
- Grid and snapping tools to make precise adjustments
- Physics and constraints to ensure results are valid
- Tools to help the user to make informed decisions

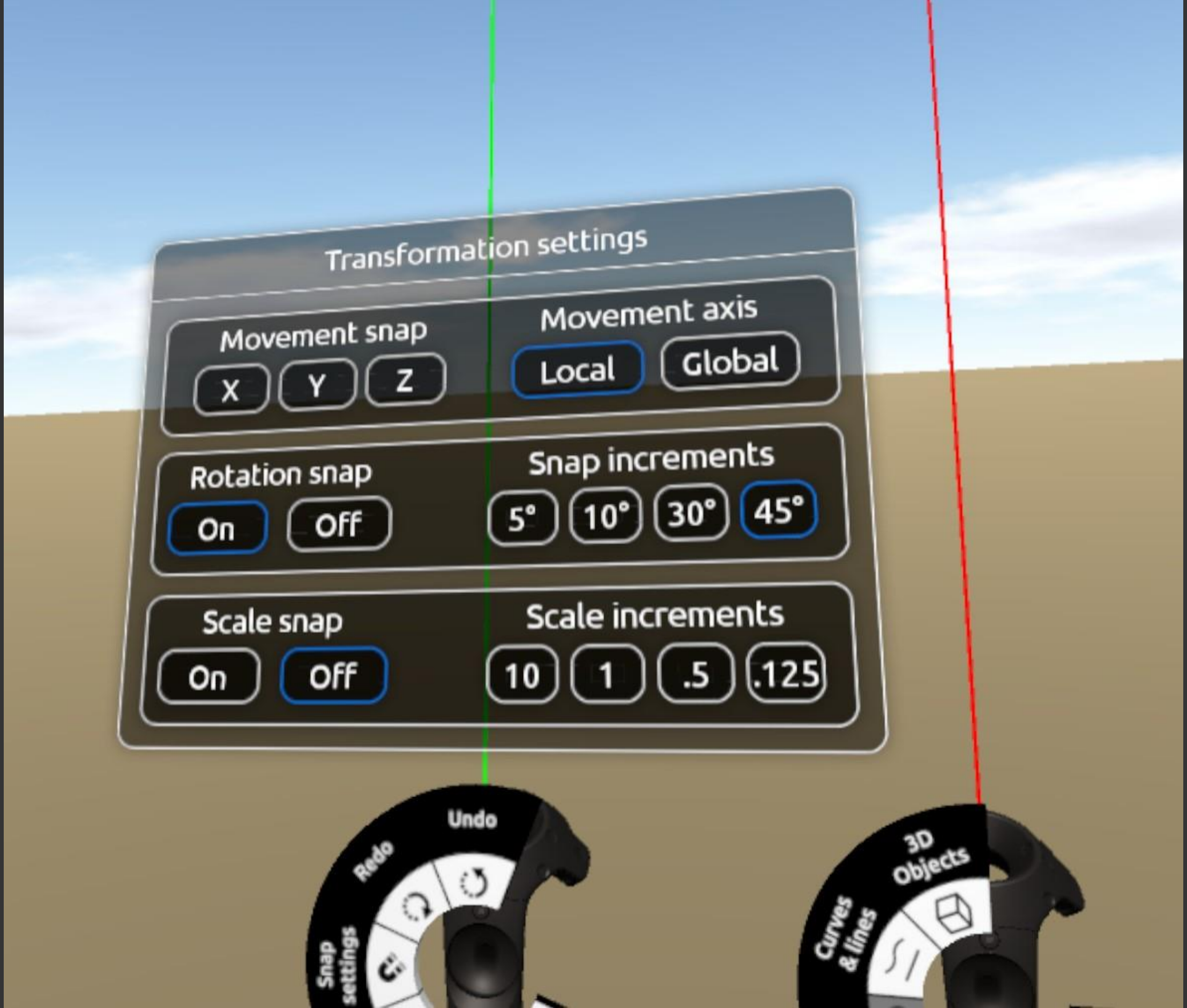
Performance in VR and user comfort

- Rock solid 90 FPS all the time
- 8x MSAA
- Not a single complain about sickness

The UI

- Excellent readability
- Tooltips
- Helpers for in-world widgets
- Accessible to people with experience with 3D applications


Redo the last action done



Transformation settings

Movement snap: X Y Z

Movement axis: [✓ Local] Global

Rotation snap: [✓ On] Off

Snap increments: 5° 10° 30° [✓ 45°]

Scale snap: On [✓ Off]

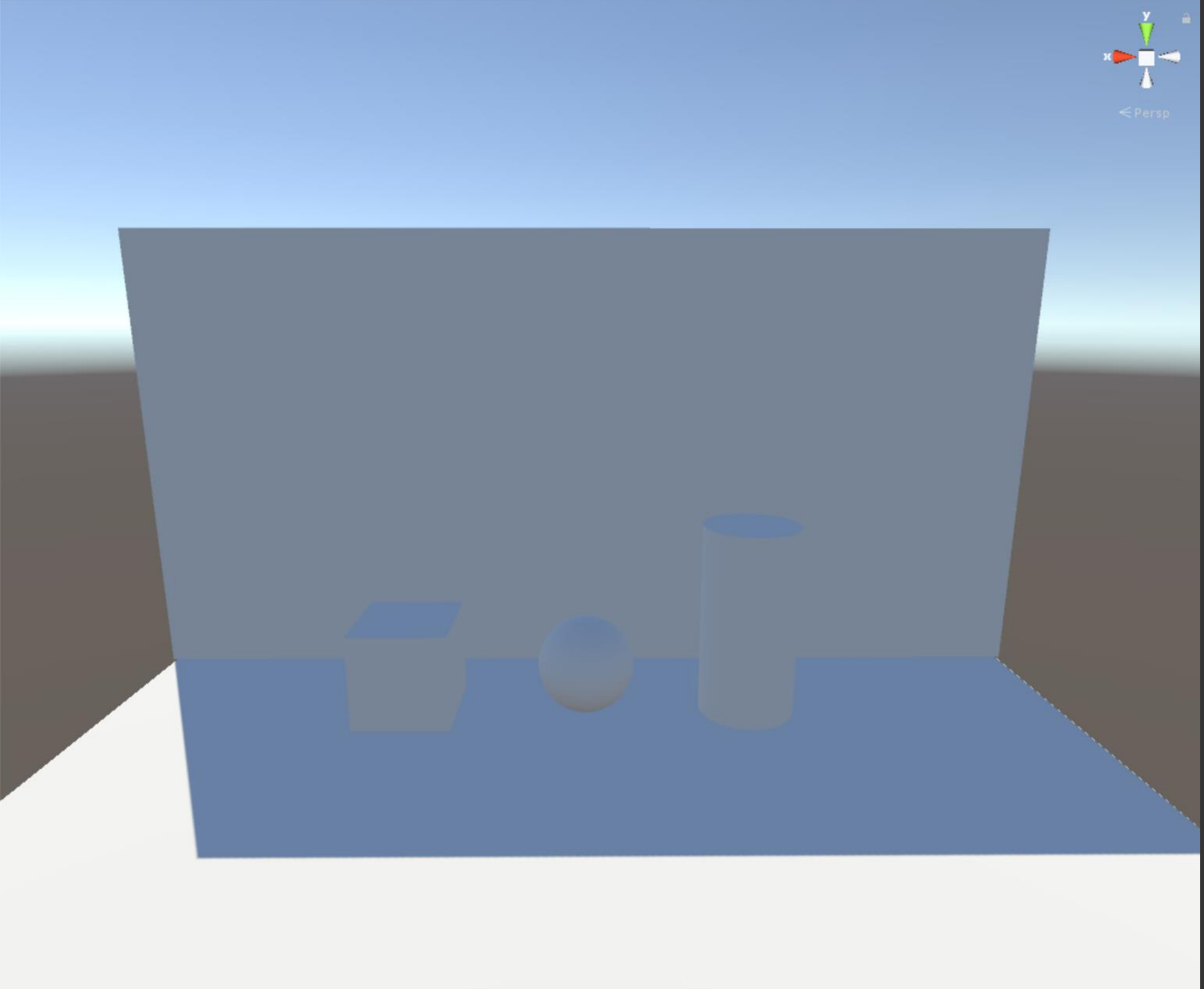
Scale increments: 10 1 .5 .125

Collaboration

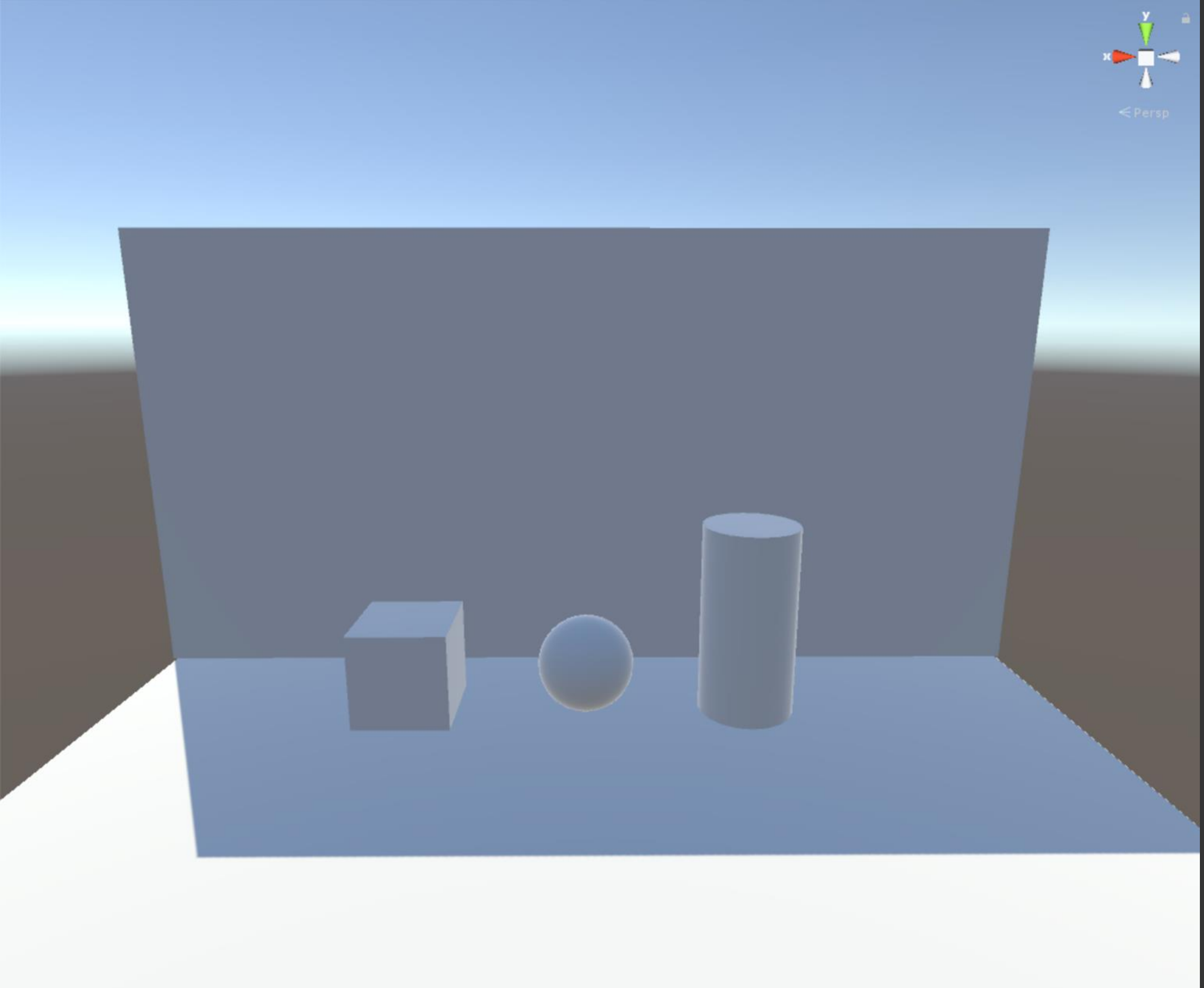
- User’s scene and actions shared with client devices via network messages
- View the scene and camera even if not in VR
- Annotate objects and comments in world space
- Works on PC and mobile devices

World and object rendering

- Dynamic time of day system (plugin from Asset Store)
- 1 dynamic shadow-casting directional light
- Fresnel based additional highlight to make objects ‘pop’



Unity3D standard shader



Hephaestus custom shader

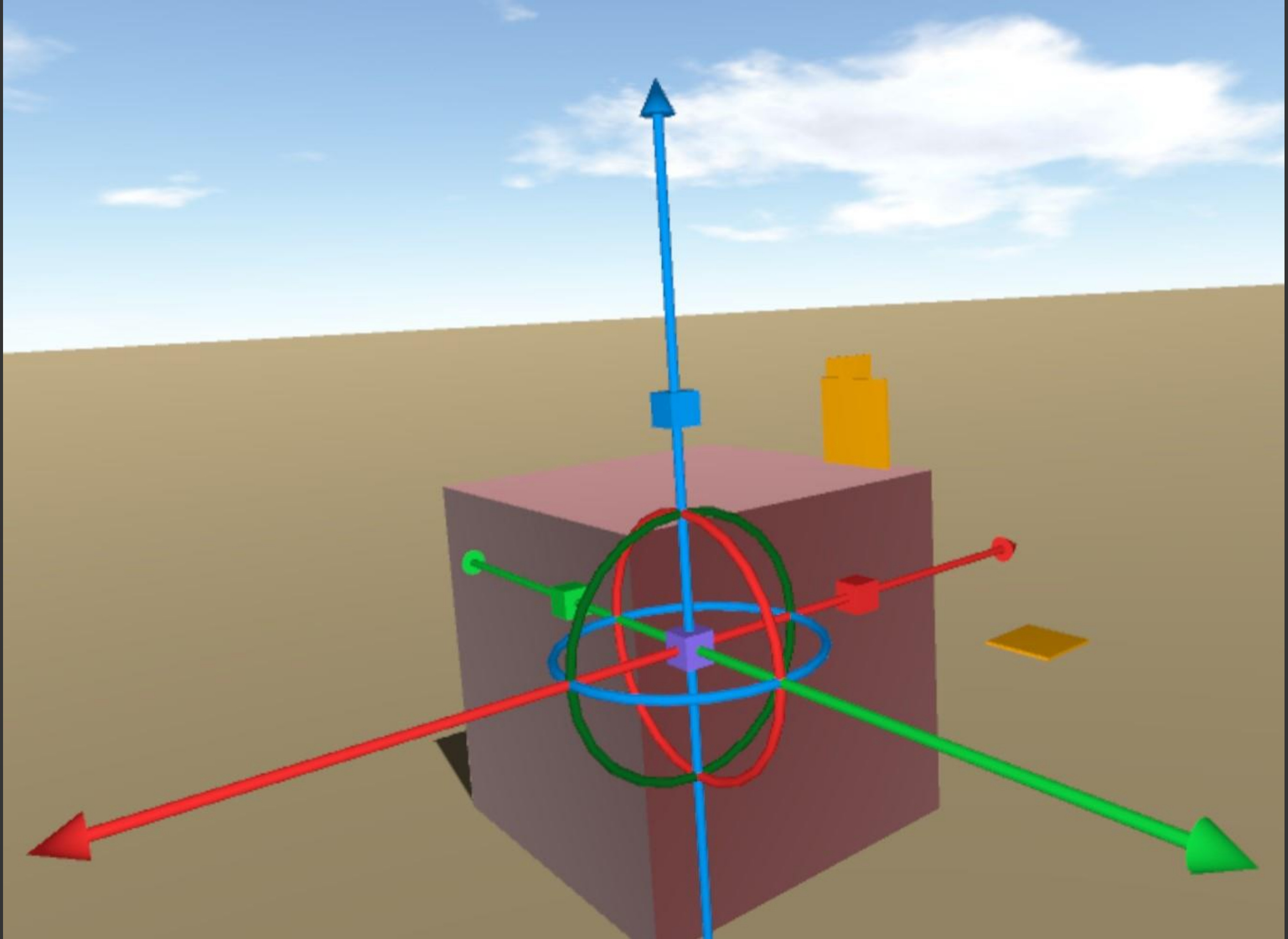
Unity3D game engine

- Really good for quick prototyping
- Easy to pick up, none of the programmers had prior Unity experience
- Asset store plugins came in handy
- Scaling across platforms is simple

What went wrong?

The UI

- Not enough time spent iterating on UI
- None of the UI panels had a close button
- Had a learning curve for Architects, Civil Engineers and ‘old school’ people
- Traditional transformation gizmos don’t work well



Mesh operations

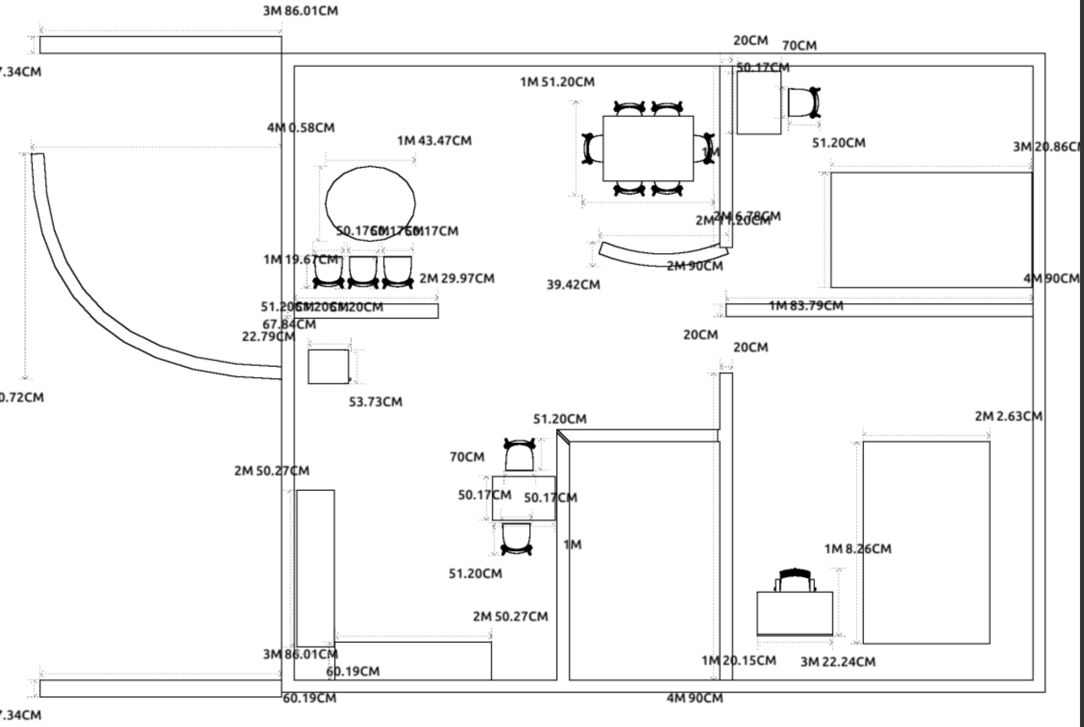
- Used CSG for creating complex shapes
- Booleans are complex operations!
- Repeated boolean operations resulted in the meshes bigger than 64K
- Boolean operations resulted in concave meshes, Mesh colliders don't work well on concave meshes
- Unity Asset store has lot of CSG tools but none of them work at runtime

Collaboration

- Didn’t consider collaboration and network support at the start
- Had to refactor lot of code later for network support
- Limitation of network packet size greater than 64K

Exporting data

- Exporting to CAD turned out to be tricky
- Top down screenshot with edged outlines and dimensions
- Not perfect, but gets the job done for a POC



“CAD” output from Hephaestus

Not enough user testing

- Spent a lot of time developing, not enough testing
- Example: creating objects bigger than yourself
- Dev team didn’t had architectural experience
- Didn’t take it to the end-users to take feedback till fairly late

Conclusion

- VR is definitely the way forward for the AEC industry
- People aren’t ready to give up existing tools yet
- Prototyping greatly helps to decide

Questions?

Thank you!

PS: We are hiring!
careers@smartvizx.com