Notes on designing for the motion simulator (SIGGRAPH Single-Seat Edition)

- **The Platform can be anything you imagine.** It may look like a bike from the outside, but it is an immersive black-box theater. Anything that moves can be simulated (more or less).
- **Players have limited time.** The runtime clock is controlled by the show system (your game) and is currently set at 1 minute 45 seconds. This is a balance of what the industry calls "throughput," how many people can go through the attraction in an hour. Longer ride cycles reduce throughput which reduces revenue. In our case for SIGGRAPH, we want a steadily moving flow of guests playing all the different experiences.
- **Players have to 'get it' right away.** Because of the limited time in the attraction, the players need to understand their roles and controls within the first 5-10 seconds. The experience can ramp up from there, but they have to be engaged and empowered immediately.
- **All players need to be engaged.** This is a one-seater platform. But there are people in the queue, waiting to get on the platform.
- **Don't forget the people in the queue.** While waiting in line to ride, they can be learning about the experience - learning the controls, story, or other info that will help them get into the story and their roles faster. They can also play along or against the people inside the motion platform using external computers or mobile devices. Extend the experience and increase the audience engagement.
- **Get creative with controls.** The platform will have handlebars, possibly with one or more buttons, and possibly a twist-throttle axis. The two passive axes, roll and yaw, are also inputs and should factor into your vehicle control.

Unity engine notes

- **Use the provided script (Unity package)** - put it on anything you want to drive the platform
- **Use forces** - you can drive your object with transform.position, or velocity, but forces will give the best results and should be the default until you can make an argument for another option
- **The object with the script must stay alive** to prevent ethernet communication errors (so do not destroy between levels/scenes)

Show PC specifications

- Windows 10
- Intel Core i9-7900X 3.3GHz 10-Core Processor
- GeForce GTX 1080 Video Card
- 16GB RAM

Display Specifications

- Rider: HTC Vive
- Queue: Shared flat panel, likely 1920x1080 res or higher

Final Polish notes

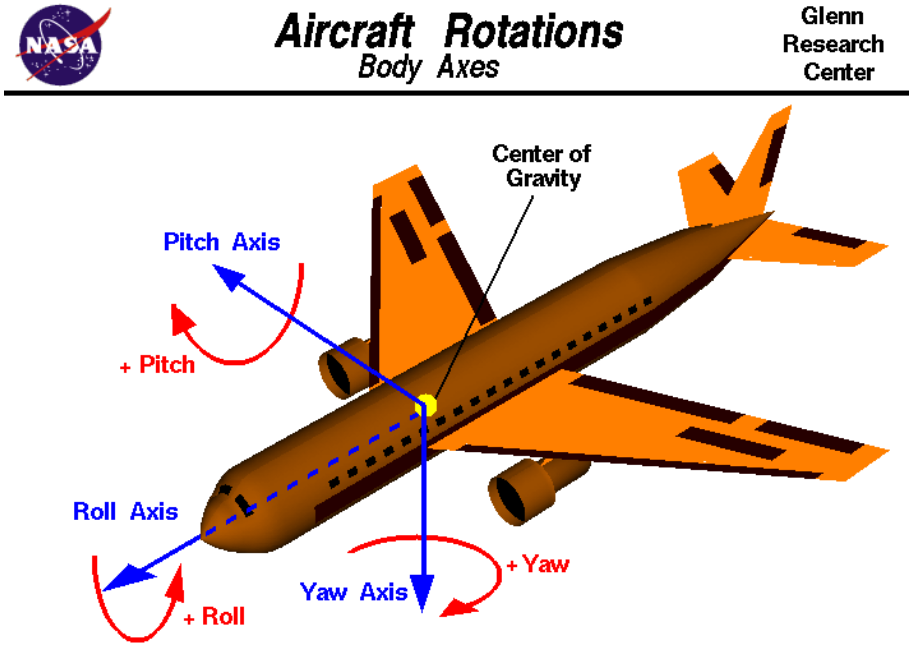
- We currently use Steam to launch games
- Create and embed a game icon
- Create and assign a game banner image for Steam tile/grid view (jpg or png, 460 x 215 resolution)
- Skip the dialog for final game builds (not for dev builds). When making a build to run on the motion system, the platform and controls are known so bypass the launch dialog for a streamlined launch. VR for rider display. Full screen at 1920 x 1080 resolution for the shared queue screen.

Design specs/guidelines

- 1.2 degree of freedom (DoF) platform (TBD)[1]
  - 1 active axis (pitch) (TBD)
  - 2 passive axes (roll, yaw)
  - roll and yaw mechanically linked so you get one with the other
  - electric linear actuator
  - gas springs for re-centering
- motion control system (TBD)
  - Unity script is placed on object
    - object driven by Unity physics system (forces)
  - Unity to custom middleware
  - middleware to SimTools
    - converts physics results to motion cues
  - SimTools to control boards
- game environment
  - single build multiplayer
  - asymmetric
  - 1 Rider on platform
  - 3-6 players at queue stations
  - multiple displays
    - Rider VR, 360
    - Queue, shared overview, single large flat panel
  - rider player inputs
    - (input via Happ/Suzo UGCI board supplied by MSL)
    - roll & yaw rotations (HTC Vive tracker?)
    - handlebar-mounted
      - at least one momentary button on each grip
      - potential twist input for acceleration
      - need way to control pitch input – simple thumb input (rocker, d-pad, or mini-trackball?
  - queue player inputs
    - Gamepads by default, one per station
    - (Hardwired Joysticks and/or Buttons?)
  - 4D FX?
    - rumble
    - wind, fog?
- Guest flow
  - Rider (5-minute cycle)
    - ~1.5m load
    - 1.5-2 minute ride duration
    - (~1.5m pre-load w/2nd HMD)
    - ~1.5m unload
  - Queue

[1] Due to various project and real-world constraints, the active pitch actuator may be cut for this year's model, leaving the platform just an input device with roll and yaw action.

        - playable stations spaced through queue OR final spaces in queue
        - (if 2nd HMD, then final space is not playing and pre-loaded)
    - Post (survey, swag, sell, shake)
- Rider Coordinate system
    - We use a **variation**[2] of aerospace coordinate standards (where Y & Z are swapped, we invert the rotation for them, too – see below this graphic)



    -
    - X = (linear) “surge” +forward, -reverse
    - Y = (linear) “lift” +up, -down
    - Z = (linear) “sway” +left, -right
    - X = (rotation) “pitch” +nose up, - nose down
    - Y = (rotation) “yaw” +left, -right
    - Z = (rotation) “roll” +left, -right
    - *FYI, rotations get different labels in other systems.*

[2] This is to map to the Motion Simulation Lab’s motion control system on the Voyager system so work can migrate platforms easily.