TNM093 — Practical Data Visualization and Virtual Reality
Tri-modal Interaction With Scene Graphs

September 4, 2025

1 Introduction

This is the Virtual Reality (VR) exercise covering tri-modal interaction and navigation with scene graphs, as part of the course TNM093. This mini project may be conducted either in pairs (recommended) or individually.

1.1 Purpose

The purpose of this exercise is to let you familiarize yourself with how scene graphs can be used to compose a VR environment with graphical, haptic and sound feedback, also referred to as tri-modal interaction, as well as some basic scene navigation. You will also experiment with aspects of visual, haptic and sound perception to experience their interplay and how these aspects bridge the gap between the virtual environment and the user's workspace.

This is also an exercise in navigating, interpreting and understanding documentation. The laboratory assistants will also be helpful in that regard.

1.2 Examination

The examination of this mini project is split up into two oral presentations; for Part 1 and Part 2 in this document.

- Part 1 is presented to a lab assitant during a scheduled supervised lab session in a regular LiU computer lab room (e.g. TP4003).
- Part 2 will be examined in the VR laboratory (G503) at an individually scheduled session at the end of the course. The reason why the examination for Part 2 is later in the year is that it takes a while for everyone to get access to the VR laboratory, and to perform the tasks in there. Only a few students can be in that lab room at a time.

Information about how to schedule an examination session for Part 2 will be provided via e-mail. Please note that you may not present Part 2 unless you have first passed the examination for Part 1.

Also, you can start preparing the tasks for Part 2 during the supervised sessions in the LiU computer lab room — only the final experiments and presentation must be done in the VR laboratory (G503). Make sure to have continuous discussions with the lab assistants when working on the tasks, to verify that you are on the right track. This will reduce the risk for misunderstandings, and lead to smoother examination.

For a successful examination, also make sure that you fully understand what you have done. Use notes and good file names to remind yourself of what you have done and in which file; it might be some time between doing the first task and finally presenting to a lab assistant. You are allowed to bring and refer to these notes during the examination! Also, during the tasks, save intermediate steps, either in separate files or in version control, so that you can show and explain the result from one individual task (where applicable).

1.3 Reflection Document

At the end of the course, you are to submit a reflection document covering the different parts of the course. You write and submit either individually or as a pair of two students, i.e., the same group as you have been examined. Use the questions marked with **Reflect:** in the laboratory tasks as a foundation for your discussion related to this miniproject. We recommend that you work on the reflection document continuously as you finish the tasks; the notes and reflections will be useful during the examination of this miniproject.

When writing the document, please focus on *reflections*, not *descriptions* of what you did. You are *not* expected to format the reflection document as a lab report; keep it simple focusing on those concepts/aspects that you deem important. Also, please note that you are expected to submit your reflection report *after* you have completed your examination in the VR laboratory (G503). This allows you to incorporate the feedback from your examination session in the reflection.

1.4 Equipment

This exercise can be implemented on computers in the regular LiU computer lab rooms as well as remotely over *ThinLinc* (thinlinc.edu.liu.se)[1], where all the necessary software is installed. Many tasks, however, *must be finalized* on a haptics workstation in the *VR laboratory* (G503). This room has only two tri-modal workstations, so carefully prepare the tasks before booking a slot.

The VR laboratory is in Kopparhammaren 6, floor 5. To gain access you need to follow an *instruction video*, read the *conduct rules* and accept those rules by signing a form. More information should be available in the Lisam online course room.

Important safety remarks:

The haptic devices used in the lab are high-end laboratory equipment and thus both expensive and not very robust. This mini project also contains experiments on the limit of what the equipment can handle. You should therefore be careful to follow the instructions and *not* try parameter values outside the prescribed range.

- Keep the pen within its workspace during experiments. If you are unsure of the workspace, try moving the pen around when no haptic program is running.
- *Always* hold the haptic pen when a program is running. Pick up the pen before starting the program and put it down when the program has finished completely.
- Keep an extra firm grip on the pen when experimenting with stability. It can pull out of a loose grip and thereby cause damage to itself or other parts of the display system.

1.5 Getting Started: Software and Set-up

The software we use in this exercise, the multi-modal scene graph framework *H3D API* in combination with our own extensions *HVR* and *H3D Candy*, is open source and will be provided upon request. Even when installed, however, the system will not find the executable until you load the correct environment variables. This is done a bit differently on each of the three available setups:

- In the LiU computer rooms (currently K4504, K4507, TP4003, TP4004, TP4005, TP4015, TP4016, TP4021, TP4022 and TP4023): To open a command window with the correct H3D environment set under Windows, open `H3D.liu.bin` in the *start menu*.
- In the VR-laboratory, either use the batch file `openenv`, located in the VRLaboratory folder, to open a command window, or run the `setenv` batch script from your own command window.
- In ThinLinc Linux, you need to load the course module in the terminal that you want to run your code from. This is done by executing the command

```
module load courses/TNM093
```

[1] Note that some OpenGL bugs will be experienced when executing H3D over ThinLinc.

After you have successfully loaded the environment in a command window using one of the methods described above, you can open your scene graph by first navigating (using `cd` or `cd /d`) to the folder where your lab files are located and then executing (for example)

```
H3DLoad scene.x3d
```

There is a code stub available for the exercise, that loads the required libraries and disables the standard X3D navigation. The standard navigation is based on moving the viewpoint, which we in VR instead want to control either manually or using head tracking.

1.6 Documentation

The tasks presented as part of this mini project are not a manual, so you are expected to search for information in documentation. Please check the documentation before the first session. As you progress through this mini project, you will return and make frequent use of these resources.

Useful sources for this exercise are

- H3D Wiki — Tutorials, Getting Started, Python documentation and other information about H3D
  https://github.com/SenseGraphics/h3dapi/wiki
- H3D API 2.4 Documentation — reference manual for H3D and X3D nodes
  https://www.itn.liu.se/∼karlu20/work/H3D-docs/H3DAPI/
- H3D Manual — instructions, descriptions and explanations by topic
  https://www.itn.liu.se/∼karlu20/work/H3DAPI_Manual.pdf
- Candy + HVR Toolkit 2.0 Documentation — reference manual over Candy and HVR nodes
  https://www.itn.liu.se/∼karlu20/work/H3D-docs/Candy
- Candy Toolkit Wiki page — documentation over Python scripts in Candy
  https://github.com/SenseGraphics/h3dapi/wiki/Candy
- HVR Toolkit Wiki page — documentation over Python scripts in HVR
  https://github.com/SenseGraphics/h3dapi/wiki/HVR
- Candy and HVR examples
  LiU lab: `C:/Program Files/H3D.liu.bin/Candy/x3d/`
  VR lab: `C:/Apps/H3D.bin/Candy/x3d/`
  ThinLinc: `/opt/liu/h3dapi/2020.08.28-h3dcandy/share/H3D/Candy/x3d/`
- Candy python scripts (accessed in X3D as "`urn:candy:python/`")
  LiU lab: `C:/Program Files/H3D.liu.bin/Candy/python/`
  VR lab: `C:/Apps/H3D.bin/Candy/python/`
  ThinLinc: `/opt/liu/h3dapi/2020.08.28-h3dcandy/share/H3D/Candy/python/`

Observe that fields in X3D are containers, providing the support for routing, and that it might not be obvious from the documentation what they may contain. There are three common cases, however. In all cases, the name of the contained type comes after the prefix SF or MF, where SF corresponds to single-value fields and MF to fields with multiple values (like an array of values). In the first case, they may contain a value, such as bool, float, Vec3f or Rotation (for example `SFBool` and `MFBool`). In a second case, the field works as a pointer to a specific node type, such as `SFAppearance`. In the third case, the field works as a pointer to any sub class of an abstract type. One such example is `SFGeometryNode`.

Abstract node types in H3D have "H3D" or "X3D" first in their names and end with "Node". So for example, for `SFGeometryNode` the correct container type can be found by searching for both `H3DGeometryNode` and `X3DGeometryNode`.

Part 1 — Experiments With Presentation in the Regular Computer Lab

Before we can start experimenting with tri-modal interaction, we need a scene to interact with. This first part of the mini project involves building your scene, adding some simple interaction, as well as experimenting with modifying the point of view.

Examination: The following tasks can be fully completed in the regular LiU computer lab room (e.g. TP4003). When finished, present your work and conclusions to a lab assistant during a supervised session.

2 Preparation — Documentation

Task 1 — Documentation:
Before the first session, familiarize yourself with the documentation (the various links listed above). Try to understand which source contains what kind of information and how each of them may be useful.

Reflect: What is the difference between a *wiki*, a *manual* and a *reference manual*? What type of information is communicated in each type? What does a "getting started guide" add?

3 The Scene as a Graph

As a first step you will create objects to work with in the following tasks. You use X3D to create and distribute these objects in the scene. To create several copies at different positions you may use `DEF` and `USE`, and `Transform`. Check the H3D reference manual for more information about the nodes and their type abstraction.

The algebra behind 3D graphics ($\vec{x}' = P\ V\ M\ \vec{x}$) is scale invariant and independent of where the origin is placed. Theoretically you can run a VR system with inches or millimetres and place origin at the centre of the earth with the same end result as when using meters and placing the origin at the screen. The system default for H3DAPI, however, when calibrated for head tracking in the VR laboratory, is to use meters and with origin at the centre of the screen.

Task 2 — Create a basic scene:
Create your own (simple) chair design by composing several geometrical primitives. Do not forget to adjust the size of your objects to fit the workspace of the display system (see hint below). To make the following tasks more interesting you should distribute the objects so that some are in the plane of the screen ($z = 0$) and some are in front of and behind the screen. Also, do not leave the object colours at default.

Reflect: What are your entered coordinates relative to? What determines the size of your design and its parts? What difference does the z value have at this point?

Hint: We have calibrated the H3D environment for 1-to-1 metric units when run in full screen mode so you may use a ruler to determine appropriate size and translation of your objects before specifying them in your X3D file. Press `F11` to make `H3DLoad` go to full screen mode.

Task 3 — Draw your scene graph:
Draw the scene graph that you have created, to show that you understand what is happening behind the scene. Please try to use the correct structure of your current scene graph and not an abstraction. It is preferred that this task is completed on a piece of paper or digital paper, unless you are fluent in digital drawing.

Reflect: What does elements in your drawing (lines, boxes, ellipses) relate to in your code?

4 The Point of View

In VR, the viewpoint specified to the graphics library has a much larger impact on the experience than in other graphics applications. The reason for this is that the perspective and object positions here should be correctly registered with your real-world coordinates. Also, perspective, stereoscopy and parallax all need to be correct

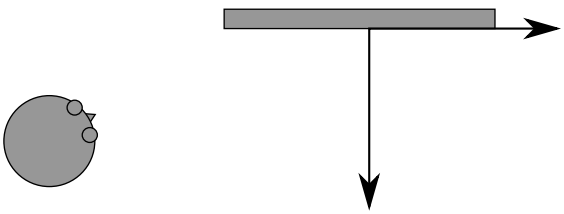

Figure 1: Looking at the screen with an acute angle. The eye's position can be measured with a ruler or estimated, for example using an A4 sheet of paper (210×297 mm).

for correct depth perception (as we will explore later!). This is often done wrong by people beginning to practice VR techniques, however there is an increasing general understanding.

In these tasks, you will manually set the position of the viewpoint. However, in a real VR application this will be controlled with values acquired from head tracking equipment, which will be explored in Part 2. Note that, since the view is dependent on the size and position of the image, it is much easier to work with the following tasks if the application window is in *full screen mode*. Press `F11` to make `H3DLoad` go to full screen mode.[2]

Important: When you do not have stereo rendering you'll need to close one eye, or your stereo vision will interfere with the perception of the perspective.

Note that you can define multiple viewpoint nodes in your scene and switch between them using the `pgup` / `pgdown` keys on the keyboard. With no viewpoint specified in the X3D scene file, the loader will apply the system default, which has been calibrated for the display.

Task 4 — Standard perspective viewpoint:
First, test the standard perspective viewpoint (`Viewpoint` in H3DAPI). This will create a basic symmetric frustum from a given position and orientation. 1) Select a location where to position your eyes in the physical room, 2) measure or estimate to determine this position's coordinates in the 3D scene (origin is the centre of the monitor), and 3) enter the correct numbers into the scene's viewpoint. You may use a view from straight ahead, but you must also try at least one view with an acute angle, as illustrated in Figure 1. How does your scene appear with your eye at the viewpoint location and far away from it?

Reflect: What does the standard viewpoint do, i.e. how does it project your scene to the screen? How should you orient your head to get the correct view; how does head rotation affect the view? Draw a figure if necessary to explain.

Hint: If the 3D monitor is reflected in a mirror, it is the reflection's coordinates you need to measure.

Task 5 — Skewed frustum perspective piewpoint:
Check the documentation for `SMViewpoint` in Candy/HVR ("SM" stands for "Stationary Monitor"). Use it instead of the viewpoint from the previous task. With the correct numbers entered, this node will provide an off-axis/skewed frustum that will match the view through your monitor. Compare with the results from the previous task, at least once with a view that is at an acute angle: How does your scene appear with your eye at the viewpoint location and far away from it?

Reflect: What does this off-axis viewpoint do, i.e. how does it project your scene to the screen? What happens if you have stereoscopy, i.e. a monitor that can show separate rendered views to each of your two eyes? What happens when head tracking is used correctly to control the view? Draw a figure of what happens and how the frustum looks like in each case (symmetric/skewed, from straight ahead and at an acute angle) to explain.

4.1 Navigation

The view is specified relative to the display system, possibly affected by head tracking. So, to enable navigation around the scene (or of individual objects) we need to manipulate the scene graph instead. Most scripts in the

[2] If you are using ThinLinc, it should also be used in full screen mode. `F8` is usually the key to enter ThinLinc menus (where you can configure its display mode).

Candy package are well documented both in the H3D Wiki and in the code (as referenced in Section 1.6, Documentation). These scripts are addressed as `urn:candy:python/XXX` in the code (where `XXX` is the file name of the script).

The mouse will act as wand for interaction on desktop, simulating the haptic pen outside the VR lab (G503). The interaction will also work in the VR lab, where the 3D force feedback stylus device (haptic pen) will provide both position and orientation tracking.

Task 6 — Add navigation:
Use the navigation scripts `ManualRotation` and `ManualTranslation` to allow navigation of one or several of the objects in your scene. See the example for `ManualTransform` in the Candy toolkit documentation. Make sure that you navigate by moving the objects and not by modifying the viewpoint.

Reflect: What is "*navigation*"? How can both head tracking controlled view and wand/stylus actions affect the visual 3D projection of the object, mathematically and conceptually? Does navigation differ between room mounted and head mounted VR displays (RMD and HMD); how in that case?

Part 2 - Experiments With Presentation in the VR Lab

The following tasks should first be set up in a regular LiU computer room, e.g. on scheduled lab hours where you can get guidance from a lab assistant, and subsequently be tested in the VR laboratory for full understanding of the VR principles. In the VR laboratory, comment out your Viewpoint instances (all of them) to enable the default, calibrated viewpoint of the system you load your file on. The mouse will act as wand for testing interaction on desktop while you will have interaction through full 3D force feedback stylus devices with both position and orientation tracking in the VR laboratory.

Examination: This second part will be examined through an oral presentation in the VR laboratory (G503), at the end of the course. At the examination, you should show and explain what you have done, as well as the conclusions from your experiments. Please note that you must first pass the examination for Part 1 before scheduling a presentation for Part 2.

5 Depth Perception

An essential aspect for getting good immersion in the VR application is to provide depth cues. There are several types of depth cues to take into consideration when working with VR, but here you will experiment with a subset: stereopsis, shadows and motion parallax. Observe that not all depth cues are equally effective, or experienced the same for all people, and especially the stereoscopic effects are predominant only at short distances. Therefore, some of the effects will not be very strong and require focus to experience.

In the following tasks you will be using stereoscopic rendering and head tracking, sometimes individually and sometimes in combination. To turn on and off stereoscopic rendering and head tracking, include the following four X3D snippets to your file. They each include both a `Viewpoint` node (controlling view position) and a `StereoInfo` node (controlling inter-pupillary distance).

Only use *one at a time*, and comment out the ones you are not currently using. Note that they will only work in the VR laboratory.

Untracked/Stereocopic	`<Inline url="urn:candy:x3d/view-untracked-stereo.x3d"/>`
Untracked/Monoscopic	`<Inline url="urn:candy:x3d/view-untracked-mono.x3d"/>`
Tracked/Stereocopic	`<Inline url="urn:candy:x3d/view-head-tracked-stereo.x3d"/>`
Tracked/Monoscopic	`<Inline url="urn:candy:x3d/view-head-tracked-mono.x3d"/>`

5.1 Shadows

We start with shadows. They provide depth cues leading to an improved perception of relative positions between objects casting shadows and shadowed objects.

Note that without light, there are no shadows. You will hence have to add a light source to you scene. The simplest type of light source is a *point light*, which only requires a position and emits light in every direction.

Task 7 — Shadow depth cues:
Add shadows to your scene using the `shadow` attribute. Add some geometry below and behind the objects of your scene, for shadows to fall upon. Also add a well positioned light source to make sure that the shadows fall onto the underlying geometry in a good way. Make sure that the background geometry is close enough to the foreground objects for the human perceptual system to easily match them for depth perception.

Reflect: With stereoscopic rendering turned off, in what ways does your perception of the scene change when the shadows are turned off and on? In what type of scenario is this most important?

[Note: There are some bugs in the shadow renderer: only the sphere geometry works over ThinLinc. In the VR-lab, where you will do the depth cue experiments, some geometries might not cast shadows if added with `USE`.]

5.2 Stereoscopy

Stereoscopic rendering in computer graphics (and its perceptual counterpart: stereoscopic vision) connects to two depth cues: vergence and stereopsis. Vergence of the two eyes when focusing on a point at a certain distance will provide an absolute sense of distance. However with respect to depth contrasts, it is weaker than stereopsis. Stereopsis, on the other hand, detect parallax between foreground objects and background, which is a much stronger cue for relative depth, depth contrasts. Looking at your scene with stereoscopic rendering but without contrasts in the background, vergence cues are active but very little stereopsis.

Since vergence is a weak depth cue we want to also trigger stereopsis. For this, contrasts are needed. We achieve this by adding texture to our objects in the scene. Such texture should contain easy to perceive contrasts but must not be repetitive at a low scale. Natural materials are typically good examples, such as stone or wood.

Task 8 — Stereopsis depth cues:
Improve your contrast between foreground and background by adding texture to your background geometry. Make sure that your background is seen around your object.

Reflect: With stereoscopic rendering turned on, in what ways does your perception of the scene change when the texture is there and not? Also, does the background geometry contribute to the sense of depth in the scene in some other ways? In what type of scenario is this most important?

5.3 Motion Parallax

Motion parallax is similar to stereopsis in its use of contrasts in the scene. However, you get the depth cues from dynamic changes instead of static image comparison between the eyes.

Note that to use head tracking in the VR laboratory, the tracker server must be active. When active, the tracker server allows any software and any number of software instances to connect and read off tracker data.

Task 9 — Motion parallax depth cues:
Keep the textured geometry to retain good contrasts in the motion parallax.

Reflect: With stereoscopic rendering turned off, in what ways does your perception of the scene change when head tracking is active and not, respectively? Move your head sideways to get the best effect. In what type of scenario is this most important?

You might have noticed that you and your lab companion have different experience in the previous tasks. This can be caused by subtle differences, for example in physiology and previous experiences.

Task 10 — Individual differences:

Reflect: Which individual differences might have an impact on the depth cues experienced above? What impact might this have for the user of a VR application and how may this influence the design process and possibly the final design?

6 Interactive Sound

Now that we have correct visual projection of our scene, and efficient depth cues, we can move on to audio. The haptic workstations are configured for 3D sound based on Head-related Transfer Functions (HRTF) technology. This can allow for a spatial effect of sound in the virtual environment and even let you hear the direction to a sound source.

In the next tasks you will experiment with spatialized sound. Note that for the spatialization to work, any utilized sound file has to be *mono*. You may use any sound file you want, or use the provided files. If using your own sound file, ensure that it is indeed mono and features as little acoustics as possible Also, well known sounds with rich frequency contents are easier to localize by hearing.

6.1 Point-source of Sound

The `Sound` node of the X3D standard implements only a basic notion of sound suitable for simple visualization but not for VR, so instead you will here have to use the `VRSound` node made available through Candy/HVR. To add sound to an X3D file you will use this `VRSound` node, which specifies the sound position, and the `AudioClip` node, which loads the audio file and handles the clip playback.

The `AudioClip` node is an `X3DTimeDependentNode`, which means that it can automatically activate at a certain time. To trigger the object by touch instead, you need to *connect* touch so that its *activation* time is the *current* time. The `isTouched` field, available in all geometry nodes (sub classes to `X3DGeometryNode`), provides an array of boolean values (`MFBool`), one boolean for each haptic device on the system. When the geometry is touched with a haptics device, the device's value in the array becomes `true`. You connect your geometry node to the audio clip via functions that convert your array of boolean output[3] to the time (now) when the sound playback should start. This can be done by combining the `MFtoSFBool` script with `BooleanFilter` node for filtering away *release* events, and `TimeTrigger` to provide the time of the touch event. For your convenience this has been encapsulated into the `TouchTimeTrigger` Python script provided as a lab file. Examine carefully the code documentation to see how to use the script.

Task 11 — Add spatialized sound:
Put at least two sound sources at the position of two different objects, and make their respective sounds activate when touching the objects.

[Note: Use sound files in WAV format, as we have experienced problems with other file types.]

Hint: You can also move the mouse device in depth (z-direction) by pressing down the middle button (or scroll wheel) while moving the mouse up or down. This will make the (simulated) haptic pen move in and out of the screen, respectively.

Task 12 — Direction of a sound source:
To better experience the effect of direction, place at least one sound source somewhere left/right of, in front/behind, and above/below the user to see if the sound spatialization can accurately render the sound direction in these cases, respectively.

Reflect: Is there a difference between how easy it is for you to hear if sound comes from left or right, from above or below, or from in front or behind? How can the computer simulate the complex effect your outer ear has on incoming sound waves? Do all humans have identical outer ear?

6.2 Adding Volume with Sound Effects

Filtering can be used to create a sense of the environment. The most simple example is the echoes in a large room, but also more complex sound behaviour, such as the sound in a small room connected to a larger room, or a sound source behind a wall, can be perceived by a user and improve their auditory experience of the environment.

Task 13 — Environment:
Use the sound nodes available in the Candy/HVR package to create the impression of a more advanced scene, such as a room with an opening to a large cathedral. You may use *presets* in this task.

Reflect: It is possible to identify the environment from the sound alone; how or why not? What effect does position/direction of the sound have on the environment effect? How does different presets affect your auditory perception of the environment?

[3] One boolean for each haptics device (zero or more) connected to your system.

Task 14 — Distance of a sound source:
Use also different distances to see which effect this has. For this task the effect becomes much easier to hear if you use an environment setting with lots of reverb.

Reflect: Is it easier to hear how far away the sound source is at closer distances ($\sim$ 0.3–1 m) or farther ($\sim$ 5–10 m)? Can you simulate the effect of someone whispering in your ear; how or why not?

Task 15 — Individual differences:

Reflect: Which individual differences might have an impact on the experience of sound in VR? What impact might this have for the user of a VR application and how may this influence the design process and possibly the final design?

7 Haptic Effects

Now that you have a working scene graph with correct viewpoint settings and sound in your environment, it is time to experiment with haptic interaction. This part can also be prepared in a regular LiU computer room, since we have a haptics simulation through the mouse device. You can add visual representation of the force feedback to assist with determining whether or not your scene is correctly configured for haptics. To do so, add the following line to your X3D file:

```
<Inline url="urn:candy:x3d/model_FeedbackTip.x3d"/>
```

This visual haptic feedback should, however, be commented out when working with the haptic pen in the VR laboratory to avoid additional distraction.

For the following task open the page for `H3DSurfaceNode` in the H3DAPI reference documentation. You should then be able to see, in the inheritance diagram, all the different haptic surface models available.

The control system that keeps the pen at or outside of surfaces is a PID controller with the haptic pen as process variable, the surface as the desired set-point and the force feedback as the control signal.

Task 16 — Enable basic touch:
Read the description for each of the three nodes that support friction to determine their different functionality. Find the surface model that supports friction but does not vary properties over a texture, and use this model to assign haptic properties to some objects in your scene. Leave the default parameters and experiment with touching the object with the haptic pen.

Reflect: What are the main consequences of adding even simple haptics? In what type of scenario is this most important?

Hint: `X3DAppearanceNode` has a field for holding an instance that "is-a" `H3DSurfaceNode`.

Task 17 — Change stiffness:
Set `useRelativeValues="false"` and experiment with the stiffness property (never more than 1500 N/m).

Reflect: How does the stiffness affect the interaction between the surface and the haptic pen? What side effect does for example a very low stiffness have? Do you experience any mechanical issues when using very high stiffness?

Hint: You may use several objects with different properties and use different colours to distinguish between them.

Task 18 — Damping surface:
Experiment with different levels of damping in the range 0–5 Ns/m.

Reflect: What difference does the damping make when touching or pressing a hard ($\sim$1500 N/m) or a soft ($\sim$200 N/m) surface? How about tapping quickly? Does the sensation change? Why? In what type of scenario is this most important?

Task 19 — Friction:
Experiment with different dynamic and static friction. Suitable values are in the range 0–2.

Reflect: How do dynamic and static friction, respectively, affect your percept of the object? What is the unit of the parameter and what does this mean?

Task 20 — Create three "realistic" materials:
Combine diffuse and specular colour, stiffness and friction properties to simulate at least *three* different real-world materials of your choice.

Reflect: Would everybody be able to identify them, even without alternatives? How do the different expressions of a material interrelate in our perception of the object? How would thermal conductivity (e.g. cold sensation of metal vs wood) or density play in?

Final Remarks

In this exercise you have experienced and experimented with different aspects of the three primary modalities used in VR, i.e. graphics, sound and haptics. Some of the tasks may appear more clearly related to VR, but all tasks are well connected to aspects that you will come in contact with in your future work with VR.

Further Reading

If you wish to learn more about Virtual Reality and relevant related concepts, feel free to start by examining the literature items listed below. You may also be interested in the Master's level course: *TNM116 eXtended Reality (XR) - Principles and Programming*, which is a continuation of the topics covered in this mini project.

Introductory VR literature:

- Steven M. LaValle. Virtual Reality. Cambridge Univerity Press, October 2023. DOI: 10.1017/9781108182874. URL (open access) `http://lavalle.pl./vr/`
- Mattias Wallergård, Joakim Eriksson, and Günter Alce. Virtual Reality - En Introduktion. Studentlitteratur, 1st edition, 2022. isbn:9789144122281.
  URL `https://www.studentlitteratur.se/kurslitteratur/`
- Paul Bourke. Calculating Stereo Pairs. Online, July 1999.
  URL `https://paulbourke.net/stereographics/stereorender/`