Surgical Robotics Environment for NVIDIA Isaac Sim

EN.601.456.01.SP24 Computer Integrated Surgery II

#### Team #8 Isaac Sim Quick Start Guide

Tae Wan Kim (tkim104@jhu.edu)

April 28th, 2024

# **Table of Contents**

[Team #8 Isaac Sim Quick Start Guide](#_95jtzw3bgnlj) 1

[**Table of Contents**](#_2tfz573ccpjq) **2**

**Isaac Sim Installation Notes**

Simple installation process for Windows and Linux, MacOS is possible but more steps are involved.

*Windows*

* Grab Installation of Omniverse from NVIDIA website: https://developer.nvidia.com/isaac-sim
* Login info required
* Download installer for NVIDIA Omniverse Launcher
* Go to Libraries, search Isaac Sim, download and launch

*Linux*

* Grab Installation of Omniverse from NVIDIA website: https://developer.nvidia.com/isaac-sim
* Login info required
* Download AppImage
* Use AppImage installer to download NVIDIA Omniverse Launcher
* Go to Libraries, search Isaac Sim, download and launch

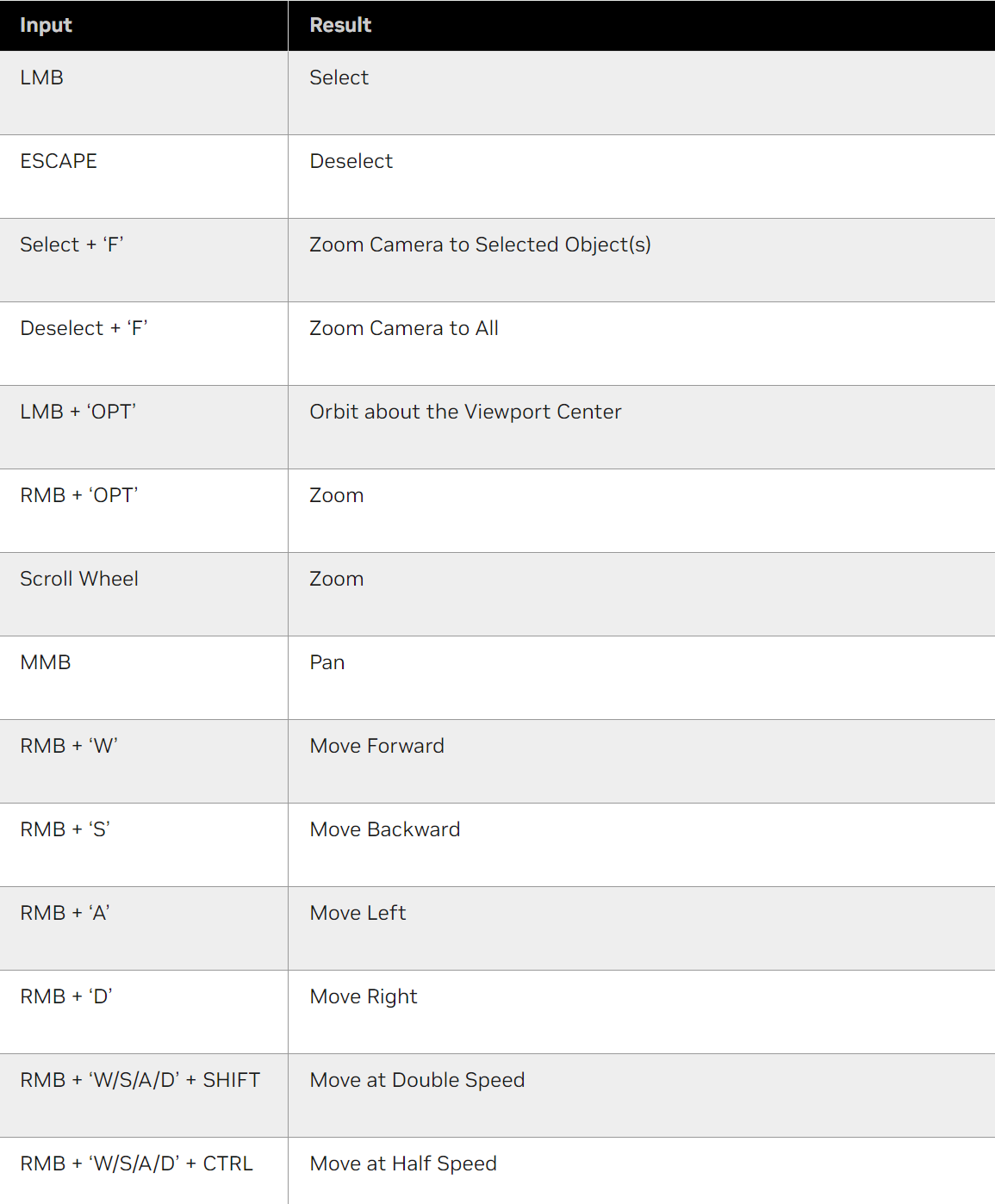
*Nucleus (Access to NVIDIA Assets)*

Account Info:

* Username: cis2team8
* Password: surgicalrobotics8
* First Name: Tae Wan
* Last Name: Kim
* Email: armonesteel@outlook.com

*Troubleshooting*

* CUDA Error 999; Sometimes you are unable to launch Isaac Sim, this may be in issue with your CUDA installation or incomplete download of a data package

**Isaac Sim Tutorial**

*Workflows*

1. GUI
2. Extensions (Headless available)
3. Standalone Python (Headless available)

*Script Editors*

1. Script Editor in GUI
   * Shares Libraries and Variables between tabs
2. Isaac Python REPL Extension (Linux Only)
   * Where you can directly use the APIs (USD or Isaac Sim Core) and put scripts directly into a scene

*Action Graphs*

Action graphs in Omnigraphs allow one to manipulate the objects within the simulator which is built in visual coder, containing nodes of several types to interact with and create a scene or environment. It can be mainly used to control joint movements in Isaac Sim.

Node types can be found here: https://docs.omniverse.nvidia.com/extensions/latest/ext\_omnigraph/node-library/node-library.html

*Direct Python Extensions / Coding*

You can directly create an Isaac Sim scene / environment directly with python by utilizing the Omniverse source API, allowing one to simulate without the need of the GUI. This allows a headless utilization of Isaac Sim, and complete API control of the simulator.

To get to user examples from terminal, follow this path:  
*/home/{User}/.local/share/ov/pkg/isaac\_sim-2023.1.1/extension\_examples*

However, you can go one folder before to reach the base of the Isaac Sim files. Here you can access the built in python and jupyter notebook interpreters to run several programs directly on the GUI or as a standalone file.

* */home/{User}/.local/share/ov/pkg/isaac\_sim-2023.1.1*
  + *./python.sh {folder}/…*
  + *./jupyter\_notebook.sh {folder}/…*

Note: When using Isaac Sim through jupyter notebook, do not use the play, pause, or stop options in the GUI, utilize the notebook instead.

*Importing URDF Files to USD for Isaac Sim*

An extension is available to import and export URDF files to USD and vice versa for Isaac Sim. This can be found in the Windows/Extension tab of the GUI and needs to be enabled if not already. Following this, one can then go to the Utilities Tab and Workflows to open the exporter or importer windows. From there, several aspects of the file can be customized before loading into Isaac Sim and the transformed data types can be further edited within the program utilizing several of Isaac Sim's data types such as Xforms, camera, lights, objects, and meshes.

*Hierarchy, Xforms, and Joints*

Like other simulation software, robots are defined with a hierarchical structure where joints and shapes can be defined through several transformations from a root node. There is only 1 root node in a model that defines the kinematic root for the rest of the linked objects. Ultimately, the two fundamental primitives needed to articulate and control a robot are Xforms and Shapes in Isaac Sim, from there several options are available to enable visuals, physics, and properties on them.

*Xforms*

Xforms is a tool to abstract away from specific objects and meshes to apply more operations on your target. One can use them to group primitives or meshes together based on function, location, or joint connections. Often they can be treated as object themselves by varying the scale, translation, and orientation of all the bodies under a specific Xform. At the same time, environment objects, such as a ground plane, lights, and cameras can be placed in their own Xforms to apply more transformations to them.

Xforms can be placed in other Xforms, they can then act as transformations from one reference frame to another by directly applying all the previous parent configurations on any primitives below it in a hierarchical manner. Overall, they act as one of the most generalizable objects in Isaac Sim.

*Shapes and Meshes*

A layer down from the Xform in terms of abstractions, shapes or primitives act as the building blocks to basic contraptions and models in Isaac Sim. The standard shapes include spheres, cubes, and more for the user to start off with. These primitives may be used to act as the actual bodies of a kinematic model, or they could be used to serve as the collision primitives when calculating the physics of a scene.

On the other hand, meshes allow one to create any type of shape beyond the simple ones mentioned above. They can be convex or concave and contain information about their collisions. While they provide a more robust way to represent any sort of 3D shape, the computation required to calculate their physics and collisions are more intensive than the basic primitives. Thus, depending on the application, it may be better to use them for visualization rather than act as the actual bodies for a scene.

*Kinematic Model / Articulation Root*

To control a model, it requires a chain of shapes and joints linked together to outline how to control each object relative to each other and the world. Usually, after you define the joints of a model and how they connect each body together, you are able to then define the articulation root of the chain that serves as the head of the kinematic model. This is often placed on the body or a robot, or frame of a machine that serves as a sort of origin. If the model is well defined (bodies are rigid or soft, physics are applied on objects, and joints are defined) you are able to create the articulation for the robot and analyze it with the Articulation Inspector in Isaac Sim. This utility is found under "Isaac Utilities" as "Articulation Inspector". After clicking the option, it lists every articulation found in the scene. Clicking on a specific one, you are then able to read all the joint positions, names, velocities, and effort applied to the model in question. If no articulation is shown, the articulation root applied did not create an articulation of the model, indicating an error with the setup. These could be caused if the complete model is defined as a kinematic model / chain anywhere on any of the objects involved. A model can either be an articulation, controllable by the user by inputting position or velocity values, or a kinematic model, an interactable group of bodies that is affected by physics.

*Basic Robot Tutorial (Summarized from Isaac Sim Documentation)*

1. Right click on the scene and select create. Select Cube.
2. On the bottom-right of the screen, you can manual change the properties of the cube
   1. Transform shifts the cube in space
   2. Orientation changes the angles of the shape
   3. Scale changes width, length, and height of the cube
3. Right click on the scene again and select create. Select Cylinder.
   1. Modify the scale values to your desired shape
4. Modify the translation and orientation aspects of the cylinders to be parallel and flat against the cube
5. On the right of the screen of objects inside the scene, alt right click on the cube then one of the cylinders
6. Select physics, create joints. Add a revolute joint.
7. Repeat for all cylinders
8. Right click on the scene space on the right and select create Xform.
9. Drag and drop all bodies and joints to the Xform
10. For each joint on the right window, right click and select add, physics, angular drive
    1. Modify parameters of dampness and stiffness depending on control scheme
       1. Position: Stiffness high, damping low
       2. Velocity: Large Damping, small stiffness
11. Right click on the scene, select physics, select ground plane to add a plane for the basic robot to drive on.
12. Hit play on the left to see your robot drive off.