The term "Colosseum" refers to multiple projects, so I'll assume you're referring to the robotics simulator by CodexLabsLLC, built on Unreal Engine with Unity support, as it's the most relevant to Ubuntu installation based on the provided context. If you meant a different Colosseum (e.g., the fraction collector or wireless emulator), please clarify, and I'll provide tailored instructions.

Important Note: The Colosseum robotics simulator does not officially support Ubuntu 22.04 or later due to Vulkan compatibility issues, and Ubuntu 24.04 is explicitly unsupported. The recommended approach for Ubuntu 24.04 is to use Docker. For Ubuntu 20.04, native installation is supported, but I'll cover both versions with Docker as the primary method for 24.04 and an optional native approach for 20.04. Below are clear, explicit instructions for each.

Installing Colosseum on Ubuntu 24.04 (Using Docker)

Since Ubuntu 24.04 is not supported natively due to Vulkan issues, Docker is the recommended method. [redacted] [redacted]

Prerequisites

1. **System Requirements**:
   - Ubuntu 24.04 LTS installed.
   - A machine with a compatible GPU (NVIDIA recommended for hardware acceleration).
   - At least 8GB RAM and 20GB free disk space.
   - Internet connection for downloading dependencies.
2. **Install Docker**:
   - Update the package index:

- Install required packages:

```bash
sudo apt-get install -y curl ca-certificates gnupg lsb-release
```

- Add Docker's official GPG key:

```bash
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrin
```

- Set up the Docker repository:

```bash
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://dov
```

- Install Docker Engine:

```bash
sudo apt-get update
sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compo
```

- Verify Docker installation:

```bash
sudo docker run hello-world
```

This should output a confirmation message. If not, troubleshoot Docker installation.

docs.docker.com

- (Optional) Allow non-root Docker usage:

```bash
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
```

3. **Install NVIDIA Container Toolkit (for GPU support)**:
   - If using an NVIDIA GPU, install the toolkit for hardware acceleration:

```bash
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /usr/share/k
curl -s -L https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-toolkit.list | \
sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg] http
sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
sudo apt-get update
sudo apt-get install -y nvidia-container-toolkit
sudo nvidia-ctk runtime configure --runtime=docker
sudo systemctl restart docker
```

Installation Steps

1. **Clone the Colosseum Repository**:
   - Create a workspace directory:

```bash
mkdir -p ~/workspace && cd ~/workspace
```

   - Clone the Colosseum repository:

```bash
git clone https://github.com/CodexLabsLLC/Colosseum.git colosseum
cd colosseum
```

2 **Build the Docker Image**:

- Build the Docker image with NVIDIA support:

```bash
docker build -t colosseum:nvidia -f Dockerfile_nvidia
```

This creates an image named `colosseum:nvidia` optimized for headless mode with GPU acceleration. [robot-colosseum.readthedocs.io]

3 **Run the Docker Container**:

- Start a container with GPU support and display access:

```bash
docker run --runtime=nvidia --gpus all -e DISPLAY=$DISPLAY \
-v $HOME/.Xauthority:/home/randuser/.Xauthority \
-v /tmp/.X11-unix:/tmp/.X11-unix --net=host -it --rm colosseum:nvidia bash
```

  - This command mounts X11 for GUI support and enables NVIDIA GPU acceleration.
  - You're now inside the container's bash shell.

4 **Run Colosseum**:

- Inside the container, follow the quickstart guide provided in the Colosseum repository (check `docs/` or `README.md` for specific commands to run simulations).
- Example (if provided in the repo):

```bash
./Binaries/Linux/Colosseum
```

Adjust based on the repository's instructions for running simulations.

5. **Verify Installation**:
   - Run a basic example simulation (refer to the Colosseum documentation or GitHub README for sample commands).
   - Ensure the simulator launches and renders correctly.

Notes

- If you don't need GPU acceleration, use the `Dockerfile_mesa` instead:

```bash
docker build -t colosseum:mesa -f Dockerfile_mesa .
docker run -e DISPLAY=$DISPLAY -v $HOME/.Xauthority:/home/randuser/.Xauthority --net=host -it
```

- Check the Colosseum GitHub (`https://github.com/CodexLabsLLC/Colosseum`) for updates or specific configuration steps.
- Join the Colosseum Robotics Discord for community support:

  Colosseum Robotics Discord

  .

Installing Colosseum on Ubuntu 20.04 (Native Installation)

Ubuntu 20.04 is supported natively, but Docker is still an option if you prefer. Below are instructions for a native installation.

Prerequisites

1. **System Requirements**:
   - Ubuntu 20.04 LTS installed.
   - A compatible GPU (NVIDIA or AMD recommended).
   - At least 8GB RAM and 20GB free disk space.
   - Internet connection.

2 **Install Dependencies**:

- Update the system:

```bash
sudo apt-get update && sudo apt-get upgrade -y
```

- Install build essentials and required tools:

```bash
sudo apt-get install -y build-essential git cmake python3 python3-pip python3-dev
```

- Install Unreal Engine dependencies (based on Unreal Engine 4 requirements):

```bash
sudo apt-get install -y clang libpng-dev libjpeg-dev zlib1g-dev libxml2-dev libgl1-mesa-dev libglu
```

3 **Install Vulkan (if using GPU rendering)**:

- Install Vulkan libraries:

```bash
sudo apt-get install -y libvulkan1 vulkan-tools
```

- For NVIDIA GPUs, install the NVIDIA driver and Vulkan support:

```bash
sudo apt-get install -y nvidia-driver-470 nvidia-utils-470
```

Adjust the driver version based on your GPU (check `ubuntu-drivers devices` for compatible versions).

Installation Steps

1. **Clone the Colosseum Repository**:
   - Create a workspace:

```bash
mkdir -p ~/workspace && cd ~/workspace
```

   - Clone the repository:

```bash
git clone https://github.com/CodexLabsLLC/Colosseum.git colosseum
cd colosseum
```

2. **Install Unreal Engine**:
   - Colosseum is an Unreal Engine plugin, so you need Unreal Engine 4 (version compatible with Colosseum, typically 4.26 or 4.27).
   - Download Unreal Engine from the Epic Games website or GitHub (requires an Epic Games account).
   - Follow Unreal Engine's Linux setup guide:

```bash
git clone https://github.com/EpicGames/UnrealEngine.git --branch 4.26
cd UnrealEngine
./Setup.sh
./GenerateProjectFiles.sh
make
```

     This builds Unreal Engine (takes time and significant disk space).

3 **Integrate Colosseum Plugin**:

- Copy the Colosseum plugin to the Unreal Engine plugins directory:

```bash
cp -r ~/workspace/colosseum/Plugins ~/UnrealEngine/Engine/Plugins/
```

- Alternatively, place it in your project's `Plugins/` directory if using a specific Unreal project.

4 **Build and Run**:

- Generate project files for Colosseum:

```bash
cd ~/UnrealEngine
./GenerateProjectFiles.sh
```

- Build the project:

```bash
make
```

- Run Colosseum:

```bash
./Engine/Binaries/Linux/UE4Editor
```

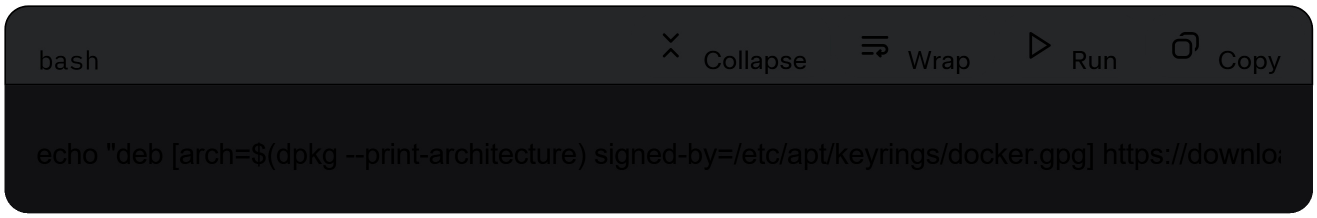
Open your project or the Colosseum example project and enable the Colosseum plugin in the Unreal Editor.

5 **Verify Installation**:

- Launch a sample simulation from the Colosseum documentation or GitHub README.
- Ensure the simulator renders and functions correctly with PX4/ArduPilot integration if needed.

Docker Alternative for Ubuntu 20.04

- Follow the same Docker steps as for Ubuntu 24.04 (above), adjusting the Docker repository for `focal`:



Common Notes and Troubleshooting

- **Vulkan Issues**: If you encounter Vulkan-related errors on Ubuntu 20.04, ensure Vulkan drivers are installed correctly (`vulkaninfo` should output GPU details). For Ubuntu 24.04, stick to Docker to avoid Vulkan incompatibilities.
- **Dependencies**: If additional Python or ROS dependencies are required (e.g., for PX4 integration), create a virtual environment:

  bash Collapse Wrap Run Copy

  ```
  python3 -m venv venv
  source venv/bin/activate
  pip install -r requirements.txt
  ```

  Check the Colosseum repository for a `requirements.txt` file.
- **Documentation**: Refer to the Colosseum GitHub (`https://github.com/CodexLabsLLC/Colosseum`) and its `docs/` folder for specific simulation setup (e.g., PX4, ArduPilot).
- **Community Support**: Join the Colosseum Robotics Discord or Slack (links in the GitHub README) for help with errors.
- **Alternative Colosseum Projects**:
  - If you meant the `pachterlab/colosseum` fraction collector, it requires 3D-printed parts and Arduino firmware. Install the GUI with:

    bash

    ```
    sudo apt-get install -y python3 python3-pip libxcb-xinerama0
    pip3 install colosseum
    ```

    Run with `colosseum` after installing Arduino firmware. [redacted]
  - If you meant the wireless emulator, request access via the form at `colosseum.sites.northeastern.edu` . [redacted]

If you need clarification on which Colosseum project or have specific hardware details (e.g., GPU model), let me know, and I'll refine the instructions!