Step-by-Step Command Guide

🚀 Day 1: Initial Setup

1. Create the project structure

```bash
# Download and run the setup script
python setup_project.py --path robust-quadruped-rl
cd robust-quadruped-rl
```

2. Set up Python environment

```bash
# Create virtual environment
python3 -m venv venv

# Activate it (Mac/Linux)
source venv/bin/activate

# Install dependencies
pip install --upgrade pip
pip install -r requirements.txt
pip install -e .
```

3. Set up Weights & Biases

```bash
# Install and login
pip install wandb
wandb login
# Enter your API key from https://wandb.ai/authorize
```

4. Test installation

```bash
# Test imports work
python -c "import stable_baselines3; import mujoco; import wandb; print('All good!')"
```

📝 Day 2-3: Implement Basic Components

1. Create the main training script

```bash
# Create src/train.py
# This is where you'll implement the training loop
```

2. Test with minimal config

```bash
# Run for just 1000 steps to check it works
python src/train.py \
    --config configs/experiments/ppo_baseline.yaml \
    --override train.total_timesteps=1000 \
    --override train.num_envs=1 \
    --override logging.wandb=false
```

3. Check output structure

```bash
# Should create:
ls experiments/
# ppo_baseline_2025_07_30_143052/

# Check logs exist:
ls experiments/ppo_baseline_*/logs/
# progress.csv
```

🧪 Day 4-5: Local Testing Phase

1. Test baseline PPO (no faults)

```bash
```

```
# 10K steps, ~5 minutes on Mac
python src/train.py \
    --config configs/experiments/ppo_baseline.yaml \
    --override train.total_timesteps=10000 \
    --override train.num_envs=2 \
    --override logging.wandb=true \
    --override logging.wandb_project="robust-quadruped-test"
```

2. View results locally

bash

```
# Start TensorBoard
tensorboard --logdir experiments/

# Open browser to http://localhost:6006
```

3. Test with faults enabled

bash

```
# Quick test of domain randomization
python src/train.py \
    --config configs/experiments/ppo_dr.yaml \
    --override train.total_timesteps=5000 \
    --override train.num_envs=1
```

🖥️ Day 6: Cluster Setup

1. Sync code to cluster

bash

```
# Create sync script
cat > scripts/sync_to_cluster.sh << 'EOF'
#!/bin/bash
CLUSTER_USER="your_username"
CLUSTER_HOST="cluster.university.edu"
REMOTE_DIR="~/projects/robust-quadruped"

rsync -avz --exclude='venv/' --exclude='experiments/' --exclude='__pycache__/' \
    ./ ${CLUSTER_USER}@${CLUSTER_HOST}:${REMOTE_DIR}/
EOF

chmod +x scripts/sync_to_cluster.sh
./scripts/sync_to_cluster.sh
```

2. Create SLURM job script

bash

```
cat > scripts/train_cluster.sh << 'EOF'
#!/bin/bash
#SBATCH --job-name=robust_rl
#SBATCH --output=logs/%x_%j.out
#SBATCH --error=logs/%x_%j.err
#SBATCH --time=48:00:00
#SBATCH --nodes=1
#SBATCH --gpus-per-node=1
#SBATCH --cpus-per-task=8
#SBATCH --mem=32G

# Load modules
module load python/3.9
module load cuda/11.8

# Activate environment
source venv/bin/activate

# Run training
python src/train.py --config configs/experiments/$1.yaml

EOF
```

3. SSH to cluster and setup

```bash
ssh cluster
cd ~/projects/robust-quadruped
module load python/3.9
python -m venv venv
source venv/bin/activate
pip install -r requirements.txt
```

🏃 Day 7-14: Full Training Runs

1. Submit baseline job

```bash
# On cluster
sbatch scripts/train_cluster.sh ppo_baseline
# Note the job ID (e.g., 12345)
```

2. Monitor job

```bash
# Check job status
squeue -u $USER

# Watch logs
tail -f logs/robust_rl_12345.out

# Check W&B dashboard
# Go to: https://wandb.ai/your-username/robust-quadruped
```

3. Submit all ablations

```bash
# Run all 4 experiments
for exp in ppo_baseline ppo_dr ppo_sr2l ppo_dr_sr2l; do
    sbatch scripts/train_cluster.sh $exp
    sleep 10  # Avoid overwhelming scheduler
done
```

📊 Day 15-16: Evaluation

1. Download trained models

```bash
# From your local machine
rsync -avz cluster:~/projects/robust-quadruped/experiments/ ./experiments/
```

2. Run evaluation script

```bash
# Create evaluation script first
python scripts/evaluate_all.py \
    --models experiments/*/models/best_model.pt \
    --output results/evaluation_results.csv
```

3. Generate plots

```bash
# Open Jupyter notebook
jupyter notebook notebooks/visualize_results.ipynb
```

🔧 Common Tasks

Check GPU usage on cluster

```bash
nvidia-smi
```

Kill a running job

```bash
scancel <job_id>
```

Debug a failed run

```bash
```

```
# Check error logs
cat logs/robust_rl_12345.err

# Common fixes:
# - Out of memory: reduce batch_size or num_envs
# - Module not found: check virtual env is activated
# - CUDA error: check GPU is allocated
```

Resume from checkpoint

```bash
python src/train.py \
    --config configs/experiments/ppo_dr.yaml \
    --resume experiments/ppo_dr_2025_07_30_143052/models/checkpoint_5000000.pt
```

Quick visualization

```bash
# Record video of trained policy
python scripts/record_video.py \
    --model experiments/ppo_dr_sr2l_*/models/best_model.pt \
    --num_episodes 5 \
    --fault_scenario "single_joint"
```

📈 W&B Specific Commands

Create new project

```python
```

```
# In your train.py
import wandb

wandb.init(
    project="robust-quadruped",
    name=f"{config.experiment.name}_{timestamp}",
    config=config,
    tags=["ablation", config.experiment.name]
)

# Log metrics
wandb.log({
    "episode_reward": episode_reward,
    "success_rate": success_rate,
    "learning_rate": current_lr,
})

# Log video
wandb.log({"video": wandb.Video(video_path, fps=30)})
```

Compare runs

bash

```
# Use W&B dashboard to:
# 1. Select multiple runs
# 2. Create comparison plots
# 3. Download data as CSV
```

🎯 Quick Testing Cheatsheet

bash

```
# Super quick test (1 min)
python src/train.py --config configs/experiments/ppo_baseline.yaml \
    --override train.total_timesteps=1000 \
    --override train.num_envs=1 \
    --override logging.wandb=false

# Medium test (10 min)
python src/train.py --config configs/experiments/ppo_dr.yaml \
    --override train.total_timesteps=50000 \
    --override train.num_envs=4

# Full local test (1 hour)
python src/train.py --config configs/experiments/ppo_dr_sr2l.yaml \
    --override train.total_timesteps=500000

# Cluster production run (24-48 hours)
sbatch scripts/train_cluster.sh ppo_dr_sr2l
```

📋 Progress Checklist

- [ ] Project structure created
- [ ] Virtual environment set up
- [ ] W&B account created and logged in
- [ ] Basic PPO training works locally
- [ ] TensorBoard shows training curves
- [ ] Cluster access configured
- [ ] First cluster job submitted successfully
- [ ] All 4 ablations submitted
- [ ] Models downloaded from cluster
- [ ] Evaluation results generated
- [ ] Plots and analysis complete

Remember: Start simple, test often, and gradually increase complexity!