

# Robust Quadruped RL Project Structure

## Directory Structure

```
robust-quadruped-rl/
├── configs/           # All configuration files
│   ├── experiments/  # Experiment-specific configs
│   │   ├── ppo_baseline.yaml
│   │   ├── ppo_dr.yaml
│   │   ├── ppo_sr2l.yaml
│   │   └── ppo_dr_sr2l.yaml
│   ├── env/          # Environment configs
│   │   └── realant.yaml
│   ├── train/        # Training configs
│   │   └── default.yaml
│   └── eval/         # Evaluation configs
│       └── default.yaml
│
├── src/              # Source code
│   ├── agents/       # RL algorithms
│   │   ├── __init__.py
│   │   ├── ppo.py
│   │   └── sr2l_ppo.py
│   ├── envs/         # Environment wrappers
│   │   ├── __init__.py
│   │   ├── realant_env.py
│   │   ├── fault_injection.py
│   │   └── curriculum.py
│   ├── utils/        # Utilities
│   │   ├── __init__.py
│   │   ├── config.py
│   │   ├── logger.py
│   │   └── metrics.py
│   └── train.py      # Main training script
│
├── scripts/          # Setup and utility scripts
│   ├── setup_env.sh  # Environment setup
│   ├── install_deps.sh # Install dependencies
│   ├── run_experiment.py # Experiment runner
│   └── sync_to_cluster.sh # Sync to cluster
│
├── experiments/      # Experiment outputs
│   └── [auto-generated experiment folders]
│
├── logs/
├── models/
├── videos/
└── metrics/
```

```
|
|
| — tests/           # Unit tests
|   | — test_env.py
|   | — test_agents.py
|
|
| — notebooks/       # Analysis notebooks
|   | — visualize_results.ipynb
|   | — ablation_analysis.ipynb
|
|
| — requirements.txt  # Python dependencies
| — requirements-dev.txt # Dev dependencies
| — setup.py         # Package setup
| — README.md
| — .gitignore
```

## Key Design Principles

### 1. Configuration Management

- YAML-based configs for easy editing
- Hierarchical config system with inheritance
- Separate configs for experiments, environment, training
- Command-line override capability

### 2. Experiment Tracking

- Automatic experiment naming with timestamps
- All hyperparameters logged
- Tensorboard integration
- Weights & Biases (optional)
- CSV logs for easy analysis

### 3. Portability

- Environment variables for paths
- Cluster-aware configurations
- Easy sync scripts
- Reproducible seeds

### 4. Ablation Support

- Modular algorithm components
- Easy enable/disable of DR and SR2L
- Consistent evaluation protocol
- Automated comparison scripts

## Configuration Examples

### Base Training Config (configs/train/default.yaml)

yaml

*# Training hyperparameters*

seed: 42

num\_envs: 8 *# Parallel environments*

total\_timesteps: 10\_000\_000

eval\_freq: 10\_000

save\_freq: 50\_000

*# PPO hyperparameters*

ppo:

learning\_rate: 3e-4

batch\_size: 2048

n\_epochs: 10

clip\_range: 0.2

gamma: 0.99

gae\_lambda: 0.95

ent\_coef: 0.0

vf\_coef: 0.5

max\_grad\_norm: 0.5

*# Network architecture*

policy:

hidden\_sizes: [64, 128]

activation: relu

*# Logging*

logging:

tensorboard: true

wandb: false *# Set to true if using W&B*

log\_interval: 10

verbose: 1

Experiment Config (configs/experiments/ppo\_dr\_sr2l.yaml)

yaml

*# Inherits from base configs*

defaults:

- /train/default
- /env/realant

experiment:

name: "ppo\_dr\_sr2l\_full"

description: "Full method with DR and SR2L"

*# Override specific parameters*

domain\_randomization:

enabled: true

curriculum:

enabled: true

phases:

- name: "warmup"

epochs: [0, 200]

fault\_prob: 0.0

sensor\_noise: 0.01

- name: "isolated"

epochs: [200, 600]

fault\_prob: 0.2

sensor\_noise: 0.05

- name: "full"

epochs: [600, -1]

fault\_prob: 0.4

sensor\_noise: 0.1

sr2l:

enabled: true

lambda: 0.01

perturbation\_std: 0.05

*# Fault injection parameters*

faults:

actuator\_dropout:

enabled: true

max\_failed\_joints: 3

sensor\_noise:

enabled: true

position\_std: 0.05

velocity\_std: 0.1

## Environment Config (configs/env/realant.yaml)

yaml

```
env:
  name: "RealAnt-v0"
  render: false # Set true for visualization

# Observation space
obs:
  include_joint_pos: true
  include_joint_vel: true
  include_orientation: true
  include_contact: true

# Action space
action:
  type: "continuous"
  dim: 8

# Reward
reward:
  forward_weight: 1.0
  ctrl_cost_weight: 0.01
  alive_bonus: 0.1

# Episode
episode:
  max_steps: 500
  early_termination: true
```

## Setup Scripts

### install\_deps.sh

bash

```
#!/bin/bash
# Install dependencies for robust quadruped RL

echo "Installing dependencies..."

# Create virtual environment
python -m venv venv
source venv/bin/activate

# Upgrade pip
pip install --upgrade pip

# Install PyTorch (CPU for local testing, CUDA for cluster)
if [ "$1" == "cluster" ]; then
    pip install torch torchvision --index-url https://download.pytorch.org/whl/cu118
else
    pip install torch torchvision --index-url https://download.pytorch.org/whl/cpu
fi

# Install main dependencies
pip install -r requirements.txt

# Install package in development mode
pip install -e .

echo "Setup complete!"
```

**requirements.txt**



```
# Core RL libraries
stable-baselines3>=2.0.0
gymnasium>=0.28.0
mujoco>=2.3.0

# RealAnt specific
# Add RealAnt-RL when available

# Experiment tracking
tensorboard>=2.13.0
wandb>=0.15.0 # Optional
hydra-core>=1.3.0 # For config management
omegaconf>=2.3.0

# Scientific computing
numpy>=1.24.0
scipy>=1.10.0
pandas>=2.0.0
matplotlib>=3.7.0
seaborn>=0.12.0

# Utilities
tqdm>=4.65.0
pyyaml>=6.0
joblib>=1.3.0
```

## Usage Example

### Local Testing

```
bash

# Quick smoke test
python src/train.py --config configs/experiments/ppo_baseline.yaml \
    --override train.total_timesteps=10000 \
    --override train.num_envs=1

# Run specific ablation
python scripts/run_experiment.py --experiment ppo_dr --local
```

### Cluster Training

```
bash
```

```
# Sync to cluster
```

```
./scripts/sync_to_cluster.sh
```

```
# On cluster (SLURM example)
```

```
sbatch scripts/train_cluster.sh ppo_dr_sr2l
```

## Next Steps

1. **Set up the base environment**
2. **Implement config management system**
3. **Create environment wrappers for fault injection**
4. **Implement PPO with SR2L extension**
5. **Set up experiment tracking**
6. **Create evaluation scripts**