EnergyShield (ICCPS '23, Submission 251)

EnergyShield is an Autonomous Driving System framework designed to save energy on-vehicle by wirelessly offloading NN computations to edge computers, while at the same time maintaining a formal guarantee of safety. In particular, EnergyShield uses a barrier function/controller shield to provide provably safe edge response times for NN offloading requests; on vehicle evaluation provides a safety fallback.

EnergyShield is described in the paper:

> *EnergyShield: Provably-Safe Offloading of Neural Network Controllers for Energy Efficiency.*
> Mohanad Odema, James Ferlez, Goli Vaisi, Yasser Shoukry and Mohammad Abdullah Al Faruque.
> ICCPS 2023: 14th ACM/IEEE International Conference on Cyber-Physical Systems.

which is appended to this document, and hereafter referred to as [ES23].

[ES23] contains a number of experiments showing the efficacy of EnergyShield in the Carla simulation environment. This README describes how to replicate those results using code packaged in a Docker image. In particular, this artifact reruns from scratch the following experiments from [ES23]:

Experiment 1 ***[ES23, Section 5.2]***: *Energy Efficiency and Safety Evaluation of EnergyShield through Carla Simulation runs*

Experiment 2 ***[ES23, Section 5.3]*** : *Performance Gains from EnergyShield given wireless channel variation*

Experiment 3 ***[ES23, Section 5.4]***: *EnergyShield generality to other DRL agents*

For each of these experiments, this artifact re-generates both new raw data and the analogous plots shown in [ES23].

Contents

0. Terminology

This repeatability artifact uses Docker. We will use the following terminology throughout:

- The **HOST** will refer to the system running Docker (e.g. your laptop).
- The **CONTAINER** will refer to the "virtualized" system created by Docker (this is where the code from the artifact is run).

Commands meant to be executed from a shell on the host or the container will be prefixed with one of the following comments, respectively:

```
# <<< HOST COMMANDS >>>
# <<< CONTAINER COMMANDS >>>
```

1. System Requirements

HARDWARE (HOST):

1. An x86-64 CPU
2. 32GB of RAM
3. An NVIDIA GPU with 8GB of VRAM **(Geforce 20xx series or later; RTX 2070, GTX 2080Ti and V100 cards were tested)**; headless GPUs will work (e.g. servers and Amazon EC2/Microsoft Azure instances)
4. At least 80GB of free disk space on the filesystem where Docker stores images (the filesystem containing `/var/lib/docker` by default)
5. An internet connection that can download ~30GB of data

SOFTWARE (HOST):

1. Un-virtualized Linux operating system; headless and cloud (e.g. Amazon EC2/Microsoft Azure) installs will work (tested on Ubuntu 20.04, but any distribution that meets the remaining requirements should work)
2. Official Linux NVIDIA drivers **(version >= 515.76 is *****REQUIRED*****)**
3. A recent version of Docker Engine **(version >= 19.03)**; also known as Docker Server but **not** Docker Desktop
4. A recent version of `git` on the path
5. The `bash` shell installed in `/bin/bash`
6. A user account that can run Docker containers in privileged mode (i.e. with the `--privileged` switch)

> **WARNING:** NVIDIA driver version >=515.76 is a **STRICT REQUIREMENT**. This repeatability artifact **WILL NOT WORK** unless the host has official NVIDIA drivers version 515.76 or higher installed.

2. Setup

(i) Host Paths

Choose an install location on the host:

```
# <<< HOST COMMANDS >>>
INSTALL_LOCATION=/path/to/some/place/convenient
cd "$INSTALL_LOCATION"
```

EnergyShield will be installed on the **host** at:

```
ES_PATH="$INSTALL_LOCATION/EnergyShield"
```

> **NOTE:** All subsequent **HOST** paths in this readme are assumed to be relative to `$ES_PATH` unless otherwise specified.

(ii) Installing EnergyShield/Building the Docker Image

To install EnergyShield and build the EnergyShield Docker image, execute the following in a Bash shell on the host (from `$INSTALL_LOCATION`):

```
# <<< HOST COMMANDS >>>
cd "$INSTALL_LOCATION"
git clone --recursive https://github.com/MohanadOdema/EnergyShield
cd EnergyShield
./dockerbuild.sh # WARNING: downloads ~30GB of data, and may take > 1 hour
even after download!
```

> **NOTE:** `dockerbuild.sh` only ever needs to be run once per host user account, unless an updated version is released.

(iii) Starting a Docker Container

First, make sure you have successfully built an EnergyShield Docker image. See Section 2 *(ii)* above.

Then to start the EnergyShield Docker container, execute the following in a Bash shell on the host (from `$ES_PATH`):

```
# <<< HOST COMMANDS >>>
./dockerrun.sh --interactive --start-carla
```

This should place you at a Bash shell inside a container with EnergyShield installed. The container's Bash shell will have a prompt that looks like:

```
carla@ece2ade62bc5:~$
```

where `ece2ade62bc5` is a unique container id (i.e. yours will be different).

> **WARNING:** if you exit the container's Bash shell, then the **container and all experiments will stop**. You may restart the container with the **host** command:
>
> ```
> # <<< HOST COMMANDS >>>
> ./dockerrun.sh --interactive --start-carla
> ```

(iv) Testing the Container

First, make sure you have an EnergyShield container running. See Section 2 *(iii)* above.

Then from the **container**'s Bash shell execute:

```
# <<< CONTAINER COMMANDS >>>
ps | grep Carla
```

You should see output listing two processes related to Carla:

```
     45 pts/0    00:00:00 CarlaUE4.sh
     52 pts/0    00:01:00 CarlaUE4-Linux-
```

> **WARNING:** if the above command produces no output or output unlike the above, then Carla is not running, and this repeatability artifact **WILL NOT WORK**.
>
> **FIX:** Double check that your host system meets the requirements in Section 1, especially the NVIDIA driver requirements (incorrect NVIDIA drivers are usually what prevents Carla from starting). If your system meets these requirements, then try restarting the container using the following sequence of commands:
>
> ```
> # <<< CONTAINER COMMANDS >>>
> # Exit from the container if it's still running:
> exit
> ```
>
> And then in `$ES_PATH`:
>
> ```
> # <<< HOST COMMANDS >>>
> ./dockerrun.sh --remove # Remove any existsing container
> ./dockerrun.sh --interactive --start-carla
> ```

3. Experiment 1 - Energy Efficiency and Safety Evaluation

In this experiment, we compared EnergyShield with purely on-vehicle NN controller evaluation, both in terms of energy consumption and safety; see [ES23], Section 5.2. This comparison was made using a single RL-trained NN controller driving a fixed track; safety entails avoiding randomly spawned stationary obstacles along this track. This artifact reuses the same track and NN controller from our experiment, but the obstacle locations and instantaneous wireless-link performance are randomized.

To rerun this experiment, ensure that you have an EnergyShield container running (see Section 2 *(iii)*), and execute the following commands in the **container**'s bash shell:

```
# <<< CONTAINER COMMANDS >>>
cd /home/carla/EnergyShield
# Run Experiment 1 Carla simulation
./scripts/run_exp1.sh NUM_EPS
```

`NUM_EPS` is an optional argument specifying the number of episodes to run (default: `NUM_EPS=3`); an episode is defined as one run of the fixed track until either the vehicle completes the track or hits an obstacle. For [ES23], we ran this experiment for 35 episodes.

> **NOTE:** Running this script with the default `NUM_EPS=3` takes around 1 hour on a workstation with 32 GB RAM and NVIDIA GPU 2070 RTX super.

> **NOTE:** When the script finishes, you will be returned to a shell prompt in the container (see Section 2 *(iii)*). If the script is successfully running, status information will be output to the console regularly.

> **NOTE:** Experiments can be run consecutively in the same container without interruption, or in between container restarts.

> **WARNING:** Occasionally, the experiment script may fail to connect to Carla even if Carla is running (see Section 2 *(iv)*). This is a known issue in Carla on slow host machines; if it occurs, simply re-run the script above. If this fails, try restarting the container according to the directions in Section 2 *(iv)*.

The primary outputs of this script are figures that summarize the energy savings and safety properties of EnergyShield (across the number of episodes specified by `NUM_EPS`). These figures can be found on the **HOST** at the following paths:

```
$ES_PATH/container_results/Fig5_Energy.pdf
$ES_PATH/container_results/Fig5_Safety.pdf
$ES_PATH/container_results/Fig6_traj_noise_False.pdf
$ES_PATH/container_results/Fig6_traj_noise_True.pdf
$ES_PATH/container_results/Fig7_Ergy_v_dist.pdf
```

Their filenames match them to the figures that appear in [ES23]. For reference, the figures from [ES23] are also available with the same filenames in `$ES_PATH/paper_results`.

This script also outputs the raw Carla simulation data of each episode (i.e. the simulation time-stamped positions, control actions, etc. of the vehicle); this data is placed in `$ES_PATH/container_results/raw_data`; the analogous raw data from our simulations can be found in `$ES_PATH/paper_results/raw_data` for comparison. The format and structure of this data is described in the subsequent Appendix.

Finally, Figure 7 (Energy vs. distance) is derived from some summary statistics of raw data noted above; these summaries are produced as several `.CSV` files output to `$ES_PATH/container_results/distance/*.csv`.

4. Experiment 2 - Performance under wireless channel variation

In this experiment, we evaluated the energy savings provided by EnergyShield as a function of different wireless-link conditions to the edge; see [ES23], Section 5.3. This experiment uses mostly the same setup as Experiment 1, including the same NN controller and track. However, in this experiment, batches of episodes are run under five different simulated wireless-link conditions. As before, this artifact reuses the same track and NN controller from our experiment, but the obstacle locations and instantaneous wireless-link performance are randomized.

To rerun this experiment, ensure that you have an EnergyShield container running (see Section 2 *(iii)*), and execute the following commands in the **container**'s bash shell

```
# <<< CONTAINER COMMANDS >>>
cd /home/carla/EnergyShield
# Run Experiment 2 Carla simulations:
./scripts/run_exp2.sh NUM_EPS
```

The optional parameter `NUM_EPS` has a similar interpretation to Experiment 1 (default `NUM_EPS=3`), but here it represents the number of episodes to run under each wireless-link condition (of which this experiment contains five).

> **NOTE:** Running this script with the default `NUM_EPS=3` takes around 2 - 3 hours on a workstation with 32 GB RAM and NVIDIA GPU 2070 RTX super.

> **NOTE:** When the script finishes, you will be returned to a shell prompt in the container (see Section 2 *(iii)*). If the script is successfully running, status information will be output to the console regularly.

> **NOTE:** Experiments can be run consecutively in the same container without interruption, or in between container restarts.

> **WARNING:** Occasionally, the experiment script may fail to connect to Carla even if Carla is running (see Section 2 *(iv)*). This is a known issue in Carla on slow host machines; if it occurs, simply re-run the script above. If this fails, try restarting the container according to the directions in Section 2 *(iv)*.

The primary outputs of this script are figures that summarize the energy performance of EnergyShield under these different wireless conditions, including the number of local-execution fallbacks required. These figures can be found on the **HOST** at the following paths:

```
$ES_PATH/container_results/Fig8_windows.pdf
$ES_PATH/container_results/Fig9_energy.pdf
```

As before, their filenames match them to the figures that appear in [ES23]. Likewise, the figures from [ES23] are also available with the same filenames in `$ES_PATH/paper_results`.

The raw data from this experiment is similarly placed in `$ES_PATH/container_results/raw_data`, and the analogous raw data from our simulations is in `$ES_PATH/paper_results/raw_data` for comparison. See the the subsequent Appendix for a description of the format and structure of this raw data.

5. Experiment 3 - Comparison Between Multiple Controllers

In this experiment, we again compared EnergyShield with purely on-vehicle NN controller evaluation, both in terms of energy consumption and safety; however, in this experiment, we compared this performance between different NN controllers (four total, including the one used in Experiment 1). Hence, this experiment effectively amounts to re-running Experiment 1 on three additional RL-trained NN controllers; see [ES23], Section 5.4. As before, this artifact reuses the same track and NN controllers from our experiment, but the obstacle locations and instantaneous wireless-link performance are randomized.

> **NOTE:** To get complete results you should run Experiment 1 first.

To rerun this experiment, ensure that you have an EnergyShield container running (see Section 2 *(iii)*), and execute the following commands in the **container**'s bash shell:

```
# <<< CONTAINER COMMANDS >>>
cd /home/carla/EnergyShield
# Run Experiment 3 generate model statistics
./scripts/run_exp3.sh NUM_EPS
```

`NUM_EPS` is an optional argument that has a similar interpretation to Experiment 1 (default `NUM_EPS=3`), but here it represents the (common) number of episodes to be run on *each* of the other three agents.

> **NOTE:** Running this script should take about three times as long as running Experiment 1 for the same number of episodes.

> **NOTE:** When the script finishes, you will be returned to a shell prompt in the container (see Section 2 *(iii)*). If the script is successfully running, status information will be output to the console regularly.

> **NOTE:** Experiments can be run consecutively in the same container without interruption, or in between container restarts. However, to incorporate the results of Experiment 1, do not *remove* the container between runs of `run_exp1.sh` and `run_exp3.sh`.

> **WARNING:** Occasionally, the experiment script may fail to connect to Carla even if Carla is running (see Section 2 *(iv)*). This is a known issue in Carla on slow host machines; if it occurs, simply re-run the script above. If this fails, try restarting the container according to the directions in Section 2 *(iv)*.

The primary outputs of this script are a collection of `*.CSV` files containing summary data for each agent, as reported in Table 1 of [ES23]. This data includes average center deviance (CD), Track Completion Rate (TCR) and average energy consumption (E). These files are output as:

```
$ES_PATH/container_results/stats_for_casc_agent_1_new.csv
$ES_PATH/container_results/stats_for_casc_agent_2_new.csv
$ES_PATH/container_results/stats_for_casc_agent_3_new.csv
$ES_PATH/container_results/stats_for_casc_agent_4_new.csv
```

The number in each file name matches the data to the corresponding Controller 1 through 4 as indicated in Table 1. Since Controller 1 was used in Experiment 1, the summary data in `stats_for_casc_agent_1_new.csv` is derived from the output of `run_exp1.sh`.

The raw data for each additional agent is likewise output to `$ES_PATH/container_results/raw_data`, with one folder for each of these additional agents. See the Appendix for a description of the structure and format of this data.

6. Appendix

Raw Data

Under `$ES_PATH/container_results/raw_data/`, every directory describes the evaluation results for a set of episodes. The name suffix of each directory describes the experimental configuration setting these evaluations belong to. For instance:

- Directory `Town04_OPT_ResNet152_Shield2_early`: experiment1 evaluations for EnergyShield early mode at default wireless settings
- Directory `Town04_OPT_ResNet152_Shield2_belay_10Mbps`: experiment2 evaluations for EnergyShield uniform mode when varying wireless channel throughput to 10Mbps

Within each of these directories, four subdirectories exist to describe whether the experiments were conducted with/without gaussian noise and with/without safety filter activated. In each one of these subdirectories, we can find the experimental data in `*.csv` format as follows:

- `plots/*.csv` files containing the individual episode's data incurred by the carla simulator when running each episode for the corresponding experimental setting.
- `valid_data.csv` is the file describing the final statistics for every episode within the corresponding experimental configuration

Information in the former include simulation time, position, steering angle, relative obstacle position, wireless conditions, safety time window, offloading actions, and performance evaluations in terms of latency and energy - all calculated instantaneously for every simulation tick.

Information in the latter include statistics for each episode about total number of ticks, reward, distance traveled, speed, average latency, average energy per inference, average center deviance, and whether the agent has hit an obstacle/curb.

Distance data

For figure 7 (normalized energy variation vs distance (meters)), raw data is used to construct the tables in `$ES_PATH/container_results/distance/*.csv` which aggregate average normalized energy consumption and #occurrences across 1 m increments of 'distance from obstacle' parameter from all episodes within the subdirectory. Each of these tables is then used to construct a plot from the figure describing how this relation varies under different experimental settings.