METRICS 2nd RAMI CASCADE CAMPAIGN 2023

Marine Domain

Docker basic guidelines

Fidel Gonzalez Leiva
Gabriele Ferri
Tommaso Fabbri

This document is subject to change, refinement and development.

Please visit the METRICS official website https://metricsproject.eu for the current version.

Content

List of acronyms

CSV Comma Separated Values

METRICS Metrological Evaluation and Testing of Robots in International CompetitionS

RAMI Robotics for Asset Maintenance and Inspection

1. Introduction

This document is a basic tutorial explaining very basic concepts related to Docker as well as its basic usage to start using it practically, oriented to METRICS RAMI 1st Cascade Campaign for Marine Robots.

2. Work environment and requirements

In this section the "Virtual Machine" (Docker container) provided to participants will be described along with the software and hardware requirements for getting started.

2.1 Docker

Docker is a tool designed to make easier to create, deploy, and run applications by using containers. Containers allow you to package up an application with all of the parts it needs, such as libraries and other dependencies, and deploy it as one package. By doing so, thanks to the container, competitors and organizers can rest assured that the application will run on any other Linux machine regardless of any customized settings that the machine might have that could differ from the machine used for writing and testing the code.

Thus, a Docker container was chosen as a starting point for the teams participating in this competition. To ease the start-up, a complete guide for installation and running the container is provided below.

For a better understanding of what Docker is and how to use it, some basic concepts will be described hereafter.

- **Docker Images**: the blueprints of our application which form the basis of containers. This image will not be modified by any of the containers, unless they save (commit) their changes into a new version of the image.
- **Docker Container:** it is created from Docker images and runs the actual application. It contains everything you need to run it - code, runtime, system tools, system libraries, and settings. Once you run an image and create a container, you should start and attach to the same container in order to access the modifications you make. Containers are not erased unless you consciously remove them. **Caution** with commands as `docker container prune` or `docker container rm <container-id-or-name>`, you can lose all your changes!!
- **Host Machine**: it is the computer where the guest operating system, in this case a Docker container, runs on. The resources of the host machine are shared with the guest machine, so no hardware needs to be virtualized.

2.2 Host Requirements

Docker host can be run on multiple hosts OS (Windows, macOS, Linux), but Ubuntu is recommended in order to connect easily your **GPU** (if you want to use Machine Learning solutions) or your **X server** (if you want to directly run GUIs from the Docker container). For those teams which will not use any of the highlighted features, feel free to choose your favourite OS. For those who plan to use the Docker container as a regular Ubuntu (despite of the desktop environment), please refer to the following recommendations.

3. Getting Started

In this section, the dependencies installation, the initial execution of the Docker image, as well as the useful container's commands to reopen and save changes will be discussed.

3.1 Dependencies

Installing Docker is really straightforward. You can go to the official web-page (https://www.docker.com/get-started) and follow the instructions found here (https://docs.docker.com/engine/install/) based on your host OS.

Pull Docker image

Once Docker is installed and working, the next step is to download the competition base Docker image. In order to get it, you only have to type the next command:

```
$ docker pull ramimarinerobots/virtual-competition-2023
```

4. Basic Usage

Once the steps of the previous sections are done, we are ready to start using the METRICS RAMI Docker image. In this section, the basic Docker usage needed to run and work with the created container will be covered.

4.1 Running the Docker Image

To create a container from the *virtual-competition-2023* image provided, the docker run command will be used. This only needs to be run the first time, though you may want to create different containers to try different things.

User: rami

and no password is needed.

Note that each time you run the docker image, a new container completely independent from the rest will be created, and changes performed to one of them will not be visible to other containers, unless you copy those changes via `docker cp` command.

The first step will be configuring out which is the image identifier, or image id. Run on a terminal the following command and you will see something like this:

```
$ docker images
```

```
la
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ramimarinerobots/virtual-competition-2023	latest	c283ad977004	4 days ago	650MB

Now copy your image id and you can directly run this image with the next command, and see a new bash session as below:

```
$ docker run -it <image-id>
```

```
metrics@1c45e4bfc539:~$
```

As far as you want to use different features such as running GUIs, using the host GPU on the Docker container, or connect host and guest to the same network in order to share roscore, different arguments must be passed to `docker run` cmd.

4.1.1 Connecting host's GPU

Please refer to https://metricsproject.eu/wp-content/uploads/2021/07/MetricsGuideline.pdf. on how to use GPU.

4.2 Connect to your Docker container

As mentioned before, you will normally run the Docker image only once, you should select which features you would like to use and run accordingly the given Docker image. If you have tested every subsection's commands, you will have many containers at this point. You can check how many containers you have initialized by typing `docker ps -a`. If you want to remove any of those containers, you can run `docker container rm <container-id-or-name>`, where the container id can be also the name or also a list of multiple space separated ids/names.

If otherwise you only have your desired container created and you want to reopen it, then you can just type:

```
$ docker start -a -i <container-id-or-name>
```

or

```
$docker start <container-id-or-name>
```

```
$ docker attach <container-id-or-name>
```

If you have already started the container, but you want to open a new terminal on the same container, you can do:

```
$ docker exec -ti <container-id-or-name> /bin/bash
```

Note that if you attach to the container, and you exit this session, either via closing the terminal, or via typing exit, or pressing ctrl + D , every bash launched via docker exec will also terminate immediately. So you may prefer to do:

```
$ docker start <container-id-or-name>
```

```
$ docker exec -ti <container-id-or-name> /bin/bash
```

4.3 Saving containers/images

Once you have finished working, or as a backup, you may want to save your container as a Docker image and save this image as a file.

To save your container as a new Docker image, you would have to type:

```
$ docker container commit -m Optional msg <container-id-or-name>

repo_name:optional tag
```

For example:

```
$ docker container commit -m Added final solution <container-id-or-name>

cmre_team
```

or

```
$ docker container commit <container-id-or-name> cmre_team:example_tag
```

Now you can run docker images and check that the new image is created with your repository name and your tag if specified. You should go to Docker hub https://hub.docker.com/ webpage, create your account and create a repository, so you can later upload images by pushing them. In this case, the repository name must be of the type `username/repo name[:optional tag].`

```
$ docker login -u <username> -p <password> #Only once
```

```
$ docker push username/repo\_name[:optional\_tag]
```

If you would like to export a Docker image as a file instead of uploading it, you can simply run:

```
$ docker save -o filename.tar repo_name[:optional_tag]
```

For loading the image saved to a file:

```
$ docker load -i filename.tar
```

4.4 Other useful commands

For tons of useful commands and options for shown commands, please refer to Docker reference https://docs.docker.com/reference/

5. Docker image for RAMI23 cascade evaluation marine robots

The use of GPUs and related APIs (e.g. CUDA) (see https://metricsproject.eu/wp-content/uploads/2021/07/MetricsGuideline.pdf) are allowed during the work on the training set and the development of the team software.

However, in the Docker image uploaded to Docker hub for the evaluation the software cannot use of GPUs and relative APIs.

Please also consider that the evaluation environment will not have an available Internet connection.

For any further question about the Docker image to produce please contact the organizers at

im@metricsproject.eu.