Introduction

B(E)3M33MRS — Aerial Multi-Robot Systems

Ing. Tomáš Báča, Ph.D.

Labs: Ing. Tomáš Báča, Ph.D

Multi-Robot Systems group, Faculty of Electrical Engineering
Czech Technical University in Prague

Lab Tasks

01 Control

- Pass = 15 points

02 Formation

- Pass = 15 points
- + optional 20 points

03 Swarm

- Pass = 15 points
- + optional 15 points

Exam + Labs = Final mark, 100 points = A, ECTS

Max 70 points from the lab towards your exam.

13 lab weeks

- 2nd week is a state holiday :-(
- Deadlines are listed at `https://cw.fel.cvut.cz/b221/courses/mrs/start`

Lab Tasks

01 Control

You will solve

- UAV control,
- UAV state estimation.

You will learn

- PID control,
- UAV dynamics,
- Linear Kalman Filter.

You will practice:

- Programming in C++,
- Simulating in Gazebo,
- Visualizing in Rviz.

02 Formation

You will solve

- Formation control,
- Mission state machine,
- Multilateration (optional).

You will learn

- multi-UAV planning,
- UAV coordination,
- multilateration.

You will practice:

- More Programming,
- Simulating in Gazebo,
- Visualizing in Rviz.

03 Swarm

You will solve

- Swarm control,
- Consensus.

You will learn

- swarming and flocking,
- consensus.

You will practice:

- More Programming
- Simulating in Gazebo
- Visualizing in Rviz

PC requirements for the labs

1. Own Linux laptop

- "any" Linux OS
- $\approx$ 5 GB of free space
- + install Singularity

2. Computer in the lab

- should work out of the box

3. Own Windows laptop

- there is a chance it will work
- there is a chance it won't
- WSL (Windows System for Linux)

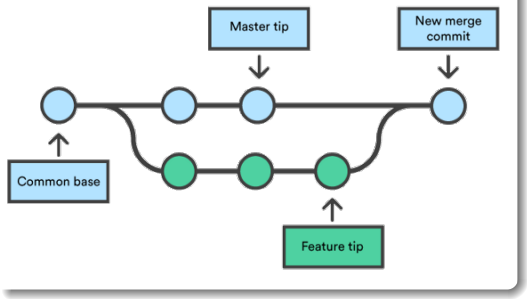
Work in Linux terminal

- `cd <path>` — changes directory to path
- `cd ..` — goes one directory higher
- `cd` — goes to `HOME`
- `./script.sh` — will execute `script.sh` in current directory
- `nano file.txt` — edits `file.txt` using nano
- `vim file.txt` — edits `file.txt` using vim
- `exit` — exits shell session
- `zip -r archive.zip folder` — zips `folder` to `archive.zip`
- `unzip archive.zip` — unzips the archive
- `tar -xvzf archive.tar.gz` — unpack the `archive.tar.gz`

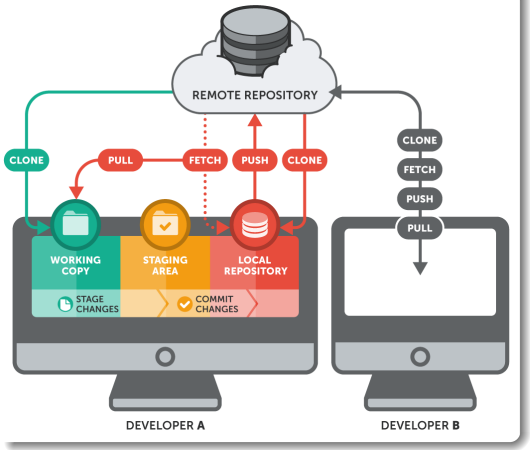
GIT — Version control system of the Open Source world

- Ability to synchronize changes between computers.
- Keeping track of own changes.
- Maintaining full history.

Git commits organized in branches



Decentralized software version control system



GIT — Version control system of the Open Source world

Git hosting servers — where to host your projects

- `http://github.com` — free public and private repositories
- `http://gitlab.com` — free public and private repositories
- `http://gitlab.fel.cvut.cz` — study-related projects

Tips for setting up

- Setup SSH keys to push/pull without password: (`https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account`)
- Make your Lab task repository private, to avoid others from stealing your unique solution ;-)
- For the start, use Github's desktop client.

GIT — Version control system of the Open Source world

Often-used git commands

- `git clone <url>` — clones repository from `<url>`
- `git status` — shows the status of the repository
- `git diff` — shows uncommited changes in the repository
- `git add file.txt` — adds changes in `file.txt` to the next commit
- `git add -A` — adds all changes to the next commit
- `git commit -m "fixed bugs"` — commits changes with message “fixed bugs”
- `git commit -am "fixed all bugs"` — commits all changes with message “fixed bugs”
- `git reset --hard` — reverts all changes (dangerous)
- `git clean -fd` — removes new uncommited files (dangerous)
- `git pull` — pulls current branch
- `git push` — pushes current branch

C++

- Vanilla C++17
- Standard libraries
- Eigen library

Visual Studio Code

- provided within the container
- code completion should work

SublimeText

- provided within the container
- code completion might work

Vim

- provided within the container
- code completion will work

Standard libraries

Output:

- std::iostream

Utilities:

- std::pair
- std::tuple

Containers:

- std::vector
- std::list

Eigen library

- Eigen::Matrix
- Eigen::Vector

MRS UAV System on Robot Operating System [1]

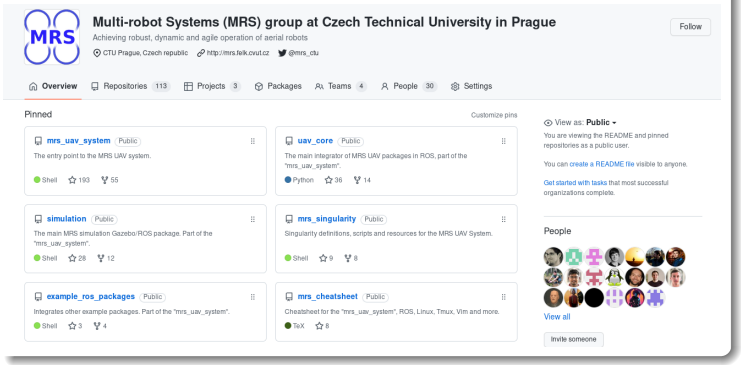
Do NOT install it!

- Used by the MRS lab to do research.
- Students use to do Bc/Msc theses.
- Contained within a Singularity container for the lab assignments

What does it do?

- UAV control
- State estimation
- UAV deployment
- Motion planning
- Sensor processing

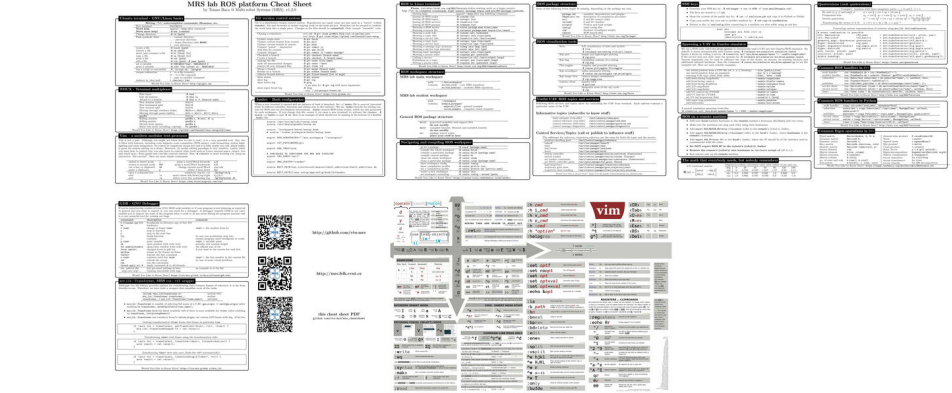
MRS UAV System



https://github.com/ctu-mrs/mrs_uav_system

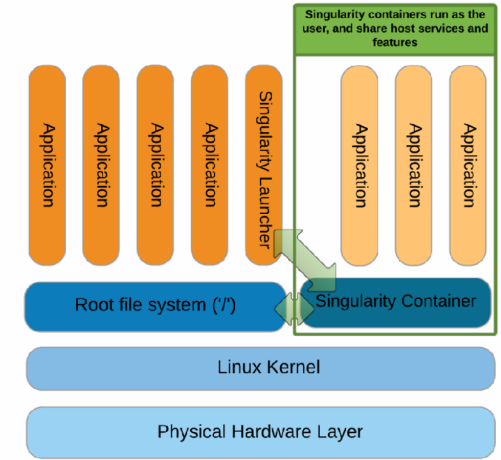
https://github.com/ctu-mrs/mrs_cheatsheet

5 pages of tips for Linux, bash, ROS, C++, Python, Vim and more



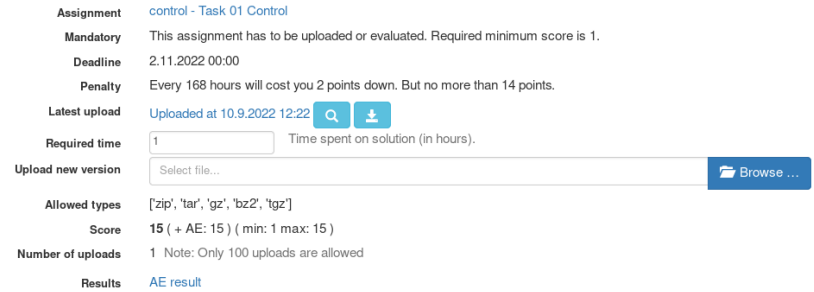
Singularity Container

- Minimal-invasive option for personal PC
- ≈ 4 GB file docker image
- Includes all build and run dependencies
- "Guaranteed" compatibility during the semester
- **No need to learn singularity**
- Students are shielded from it by our scripts



On the lab's computer

1. Go to `https://cw.felk.cvut.cz/brute/`
2. Login as a student
3. Locate the Task 01 Control



4. Download the zip file
5. Unpack the archive file (`tar -xvzf task_01_control.tar.gz`)
6. Follow the instructions in the `assignment.pdf` (section *2.3* Starting the simulation)
   - **skip the step with running** `install_singularity.sh`

On personal computer with Linux

1. Go to `https://cw.felk.cvut.cz/brute/`
2. Login as a student
3. Locate the Task 01 Control
4. Download the zip file
5. Unpack the archive file (`tar -xvzf task_01_control.tar.gz`)
6. Follow the instruction in the `assignment.pdf`
7. Follow the instruction in the `assignment.pdf` (section *2.3 Starting the simulation)*
   - **For Ubuntu-compatible OS**: follow the step with running `install_singularity.sh`
   - **For other Linux OS**: Install Singularity according to the official documentation:

     `https://docs.sylabs.io/guides/3.0/user-guide/index.html`

On personal computer with Windows

Windows system on Linux (WSL)

- Allows running Ubuntu "natively" under Widows
- It's great when it works
- Might not work due to your GPU, GPU driver, Windows version, etc.
- Works on average in 66% of time

1. Make sure your windows are up-to-date.
2. Install WSL (`wsl --install` in power shell)
3. Install Ubuntu 20.04 using the Microsoft Store
4. Download and install VcXsrv (an X-server client for Windows used to see the GUI from within Ubuntu)
5. Start the `vcxsrv.exe` by running: `vcxsrv.exe -ac -multiwindow`
6. Run the Terminal from the Start menu and launch a new terminal with Ubuntu 20.04.
7. Run `sudo apt update && sudo apt upgrade && sudo apt install gedit`
8. Run `gedit` to verify that GUI will show up
9. If the GUI does not show up, add the following line to the `$HOME/.bashrc` file in the Linux subsystem:

   ```
   export DISPLAY=$(grep -m 1 nameserver /etc/resolv.conf | awk 'print $2'):0.0
   ```

10. **Follow the Linux instructions on the previous page**

Virtualization

- It is possible, e.g., using VMWare, VirtualBox.
- Will be slow.
- Will need approx. 20 GB of space.

Apple's macOS

- Will not work with A1 and A2 chips due to different CPU architecture.
- Probably will not work on virtual machine either.
- It might work with Intel under virtual machine.
- We don't know, we don't have MACs around.
- We offer no support on MAC.

Automatic evaluation

BRUTE server

- All points earned through BRUTE
- Simulation in BRUTE is serialized
- Be patient, it might take time.
- Approx. 3 minutes for task 01, if there is no one in the queue.
- Approx. 8 minutes for task 02, if there is no one in the queue.

On local machine

- The same evaluators as in BRUTE
- Follow the instructions in the assignment pdf

Info result page in BRUTE

MRS — Task 01 Controller — bacatoma (1)

Submission

Login:	bacatoma
Year:	2022Z
Submission no.:	1
Seconds after deadline:	0

Results

Score: 15.0

Control test: PASSED, position RMSE 0.29 (limit 0.50), tilt RMSE 0.10 (limit 0.50), additional info: finished

Kalman test: PASSED, position RMSE 0.03 (limit 0.05), velocity RMSE 0.02 (limit 0.05)

Control test log:

```
student@8e514fdc840d:/tmp/ros_workspace/src$ source /tmp/ros_workspace/devel/setup.bash && roslaunch task_01_evaluation controller_test.launch file:=/tmp/tmpoe
... logging to /home/student/.ros/log/cdc6a4d6-30f2-11ed-b45f-251b71095d21/roslaunch-8e514fdc840d-3381.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.
[2;/tmp/ros_workspace/src/ros_packages/evaluation/launch/controller_test.launch
started roslaunch server http://127.0.0.1:46493/

SUMMARY
========

PARAMETERS
 * /rosdistro: noetic
 * /rosversion: 1.15.14
 * /uav1/controller_test/file: /tmp/tmpoekz4dpe
 * /uav1/controller_test/main_timer//rate: 100
 * /uav1/controller_test/path: [0.0, 0.0, 1.5, 0...
 * /uav1/controller_test/relax_heading: True
 * /uav1/controller_test/requirements/rmse_position: 0.5
 * /uav1/controller_test/requirements/rmse_tilt: 0.5
 * /uav1/controller_test/start_timer/rate: 1
 * /uav1/controller_test/stop_at_waypoints: False
 * /uav1/controller_test/use_heading: True
 * /uav1/controller_test/version: 1.0.0
```

Communication

MS Teams

- You should be added to the B3M33MRS course.
- Post general questions rather then sending personal messages.

References

[1] T. Baca, M. Petrlik, M. Vrba, V. Spurny, R. Penicka, D. Hert, and M. Saska, "The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles," *Journal of Intelligent & Robotic Systems*, vol. 102, no. 26, pp. 1–28, 1 May 2021.