# ME179P W22 - INTRODUCTION TO ROBOTICS: PLANNING AND KINEMATICS

## SUBMISSIONS GUIDELINES

FRANCESCO SECCAMONTE
THIS VERSION: JANUARY 4, 2022

Welcome to ME179P, Winter 2022!
This guide illustrates the submission procedure for the course.

### 1. Homework assignments

From the course website:

> Homework is typically due on Wednesday 5pm of the week following the corresponding lectures (e.g., Homework for Week 1 is due on Wednesday of Week 2). Warning: The week-by-week schedule below may be modified during the quarter to include a few exceptions. Please check Gradescope regularly.
> Handwritten or typed answers are to be submitted via Gradescope.
> Note: You must use the template files provided.
> Programming assignments are to be submitted on Gradescope (if you use Python), or on MATLAB Grader (if you use MATLAB). In case you do not have access to one of these, please email the TA. Note: You must use the template files provided.

Homework template files are available on the course website.

### 2. Programming Assignments

At the beginning of the course, you are required to pick your programming language (MATLAB or Python3) on Gradescope. Please note that your codebase will build incrementally, that is, assignments might require you to use code you wrote for previous assignments. This means that, unless you want to write code twice, you are going to stick with one programming language throughout the course.

You are required to use the provided templates (also called APIs), as they serve multiple purposes:

- APIs are the standard way to write code in any university lab/industry, as they represent a contract between the programmer and the user
- APIs allow to work with other people; if you don't want to use them because you feel they limit your freedom, found your own company and be the only one writing code
- APIs allow to follow standard and well established programming patterns, such as Test Driven Development (TDD)
- APIs allow to implement automatic testing (in industry, Continuous Integration)
- In the context of this class, APIs allow to provide you *continuous and real-time feedback*, without the need to wait for the TA to grade your submissions
- APIs allow a consistent and fair grading scheme, without the need for the TA to adapt the tests to the submitted code, thus avoiding the chance of introducing (additional) errors

*Date*: January 4, 2022.

The ultimate goal is to increase students' engagement and improve their learning outcomes, as well as keep the TA's life sane. The price to pay is for students to stick to the provided templates (small), and for the TA to setup two systems in advance and really make sure that tests are well written (not so small, but well worth it).

2.1. **Picking a programming language.** At the beginning of the course, you are required to choose your programming language on a Gradescope survey. If you choose MATLAB, you will be added to the MATLAB Grader course page right after the end of the survey. In case you don't receive an email invite, please contact the TA.

2.2. **Python3 submissions.** Python submissions are to be done in Gradescope.
Each programming assignment will contain instructions (typically a PDF file) and one or more template files. The template files are available on the course website. The assignments might also contain sample code to run your submission, and check it works in one or more basic test cases.
Every time you submit your solution, it will be run automatically and you will receive your score. The score of your last submission corresponds to your grade for such programming assignment.
In case your code requires some external dependencies (very unlikely), you can list them in the `add_requirements.txt` file, and upload it as part of your submission. Such dependencies will be managed automatically on the submission server via `pip`.

2.3. **MATLAB submissions.** MATLAB submissions are to be done in MATLAB Grader.
Each programming assignment will contain instructions (typically a PDF file) and *only one* template file. The template files are available directly on MATLAB Grader. The assignments might also contain sample code to run your submission, and check it works in one or more basic test cases.

To deal with the 1 function = 1 file MATLAB limitation, the templates will consist of a class containing all the functions as static methods. This object-oriented workaround is not necessarily the optimal solution, but it serves our purpose.

You can find an example below.

```
% Using a class with static methods is
% a workaround to include multiple functions
% in only one file.

% Please fill the static methods as normal
% MATLAB functions. If you want to call another
% function you wrote, simply invoke it by
% out = ALLFUNCS.otherfcn(in1, in2);

classdef ALLFUNCS
    methods(Static)

        function y = add2(x)
            y = x+2;
        end

        function y = add3(x)
              y = ALLFUNCS.add2(x)+1; % Calling the method add2 in the same class
        end

    end
end
```

LISTING 1. File ALLFUNCS.m .

```
% Example on using static methods
% in MATLAB.

x = 1;
y2 = ALLFUNCS.add2(x) % Calling the method add2 in the class ALLFUNCS
y3 = ALLFUNCS.add3(x) % Calling the method add3 in the class ALLFUNCS

% y2 should be 3, y3 should be 4
```

LISTING 2. Example main.m .

Every time you submit your solution, it will be run automatically and you will receive your score. The score of your last submission corresponds to your grade for such programming assignment.

## 3. FAQ

3.1. **Which packages/toolboxes am I allowed to use?** In principle you can use whichever package you want, except from symbolic computation and automatic differentiation packages/toolboxes.

- In Python, `numpy` should suffice, and it is already imported in all the template files.
- In MATLAB, a basic installation should suffice. On MATLAB Grader, the following toolboxes are installed (but very likely not required): Control System Toolbox, Global Optimization Toolbox, Optimization Toolbox, Robotics System Toolbox, System Identification Toolbox. No additional toolboxes can be installed.

3.2. **My solution works on the sample code provided, but fails on (some of) the tests.** The sample code is only a subset of the tests that are run to evaluate your submission. Make sure to account for all possible edge-cases, and check the inputs provided are valid. If inputs are not valid, raise an exception (in Python) or give an error message (in Matlab).

3.3. **The submission server got stuck. What do I do?** Testing sample solutions requires only a few seconds, so it is likely your solution has some bugs (infinite loops, missing return statements, etc.). In case you really think there is something wrong with the submission server, please email the TA.

3.4. **How many times can I submit my solution?** As of now, there is no limit to the number of submissions you can upload. However, this should not be seen as a trial-and-error procedure. Please only submit your code once you are confident it implements the required features.

We reserve the right to limit the number of submissions if we realize this option is being misused by the students.

3.5. **Do I need to always use Gradescope/MATLAB Grader to check my solution works?** You are encouraged to write your code offline, and test it on the sample code provided. Once you get it working, feel free to submit it and get it scored.

## 4. Regrade request

Lastly, even an automatic grading scheme can fail for a number of reasons, such as software version mismatch, server issues, weird package dependencies, wrong tests written by the TA, etc. The probability of these things happening is way lower than a human committing mistakes, but nonzero. If you feel your submission has suffered from such issues, you can submit a Regrade request, via Gradescope for Python3 submissions, TBD for MATLAB submissions (either via Gradescope or by emailing the TA). In this case, the TA will manually run the tests on your submissions and try to spot potential issues that might have happened in the automatic procedure.

We reserve the right to suspend such possibility if we realize this option is being abused by the students.

## 5. Final words

This is only the second time we implement such a systematic approach for the programming assignments. Last year, after some limited ramp-up time, students were extremely happy with it. It will hopefully be smooth sailing, but there might be a few hiccups here and there.

If you find something odd, or you want to provide any kind of feedback, please email the TA: fseccamonte@ucsb.edu .

We hope you will all have a profitable learning experience!

Prof. Francesco Bullo, Instructor
Francesco Seccamonte, TA