01 - Outline, Ch1 Preview

Vinicius Prado da Fonseca, Ph.D.
vpradodafons@mun.ca

December 18, 2025

Outline

Course objectives

- The primary objective: introduction to the mathematical formulation and practical aspects of robotic manipulators.
- It will present kinematics, dynamics, control, and programming vital to effectively using/designing robotic arms.
- Topics covered:
  - Configuration space;
  - Rigid-body motions;
  - Forward and inverse kinematics;
  - Trajectory generation;
  - Motion planning;
  - Manipulator control.
- A complete yet straightforward robotic manipulator model will be developed to demonstrate these concepts using high-level languages and frameworks.

Course details

- Instructor: Dr. Vinicius Prado da Fonseca
  - E-mail: **vpradodafons@online.mun.ca (Brightspace email) include [COMP3766] in the subject line**;
    - Send email within Brightspace. Does not receive emails from outside brightspace;
    - Use ONLY brightspace email.
  - Office: EN-2012;
  - Office Hours: Friday after class or by appointment.
- Course Prerequisites: COMP2001, COMP2002, MATH2000, MATH2050, STAT2500 or STAT2550
- Lectures (**EN 1054, M-W-F 14:00 - 14:50**) and course notes on Brightspace;
  - Lectures are recorded on Online Rooms (Bongo).

Course details (cont.)

- Labs
  - Lecture room, lecture time, some Fridays;
  - Quiz on Brightspace;
  - Course repository: `https://github.com/vncprado/COMP3766`.
  - More details next slides.
- Textbook
  - Modern Robotics Mechanics, Planning, And Control, Kevin M. Lynch and Frank C. Park, 2019 (Cambridge University Press);
  - Book website: `https://hades.mech.northwestern.edu/index.php/Modern_Robotics`
  - Online notes and slides will also be available.

Evaluation

- The final grade in the course will be determined as follows:
  - Quizzes (2) - 25%;
  - Lab exercises (4) - 20%;
  - Assignments (5 or 6) - 30%;
  - Final Project - 25%.
- Course outline on Brightspace:
  - Evaluation, lecture dates, lab schedule, assignment deadlines and textbook units.

Assignments

- Assignments (5 or 6) may have written and/or programming portions
- Submit written portions by hand in a scanned document or annotated PDF.
- Programming portions will be mostly in Python. Use the provided files as a template for your code.

Labs

- Labs (4) will be **IN PERSON**
  -
  - Lecture room, Lecture time, some Fridays, see schedule on Brightspace;
  - Repository with ROS workspace will be provided;
  - Bring laptop with **Docker** and **VSCode** installed;
  - D2L Quiz will be open until 23:59 on lab day.
- NO EXTENSIONS!
- You must do the **pre-lab section** before arriving at the lab time.
- After completing the lab you must answer the **quiz on brightspace**.
- On Brightspace you will find:
  - Lab schedule;
  - Lab manual;
  - Support files;
  - Lab quiz.

Final Project

- Repository (INCLUDE REPO LINK HERE)
- Final weeks starting after second quiz.
- Deadline last day of classes.

Robotics is a relatively young discipline with highly ambitious goals, the ultimate one being the creation of machines that can behave and think like humans.

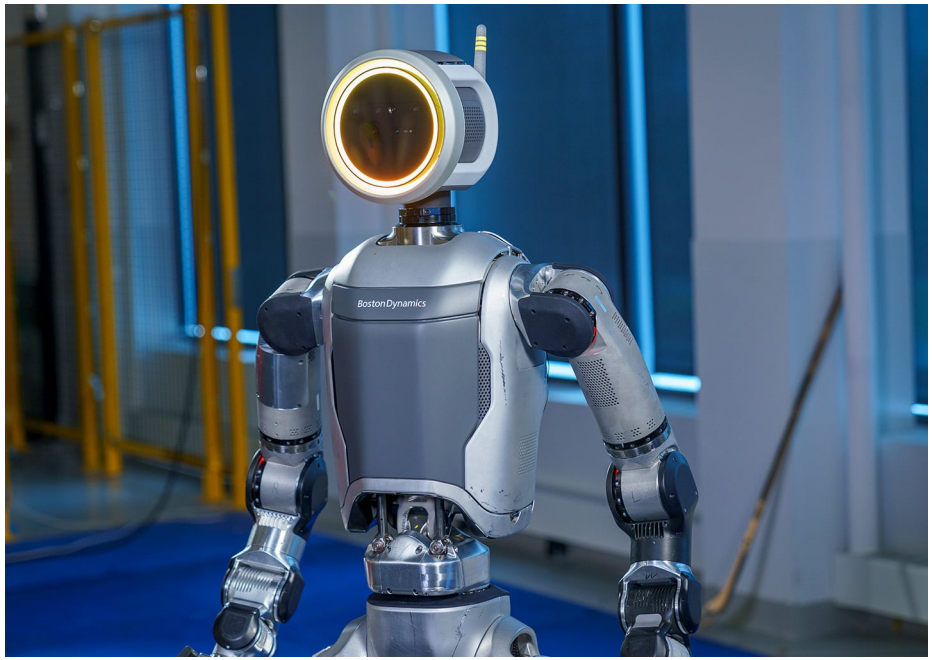
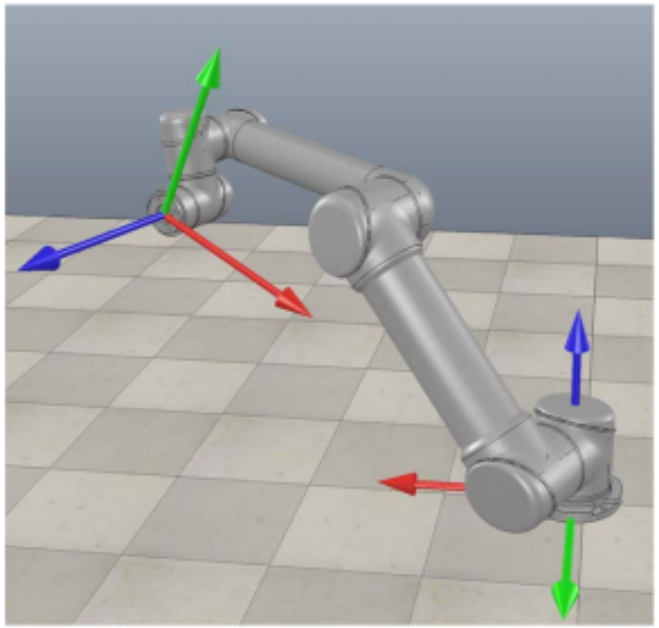


Figure: Boston Dynamics Atlas Robot.

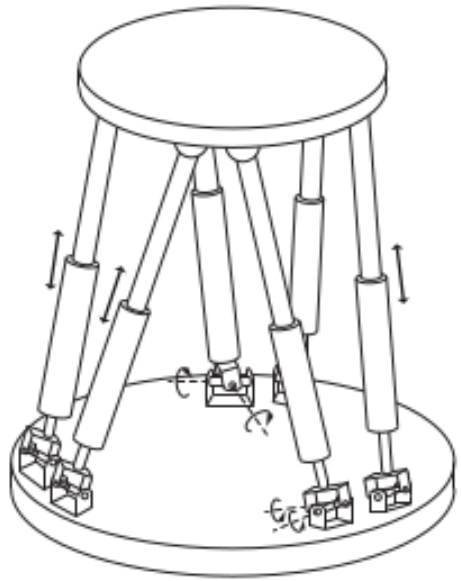
Ch1 Preview

Our focus in this course is on mechanics, planning, and control for robot mechanisms, such as robotic arms. Basically, a mechanism is constructed by connecting rigid bodies, called **links**, together by means of **joints**, so that relative motion between adjacent links become feasible. **Actuation** of the joints then causes the robot to move and exert forces in desired ways.

- Robot mechanisms can be arranged in a serial fashion like the open-chain arm in Figure 1.1
- Or forming closed loops, such as the Stewart-Gough platform.
- Open-chain, all the joints are actuated.
- Closed loops, only a subset of the joints may be actuated.



(a) An open-chain industrial manipulator, visualized in V-REP [154].



(b) Stewart–Gough platform. Closed loops are formed from the base platform, through the legs, through the top platform, and through the legs back to the base platform.

Figure 1.1: Open-chain and closed-chain robot mechanisms.

Chapter 2: Configuration Space

Here we focus on representing the configuration of a robot system, a specification of the position of every point of the robot. The robot consists of a collection of rigid bodies connected by joints. The configuration of a rigid body in the plane can be describe using 3 variables while in the space the rigid body can be described using 6 variables. The number of variables is the number of degrees of freedom of the rigid body which is also the dimension of the configuration space. The dof of a robot hence the dimension of its configuration space is the sum of the dof of its rigid bodies minus the number of constraints provided by the joints. Knowing the dof of a rigid body and the constraints we can derive Grübler's formula for calculating the dof of general mechanisms.

Chapter 2: Configuration Space

Other configuration space concepts of interest include topology and its representation. Two configuration spaces of the same dimension may have different shapes.
Example:

- two-dimensional plane;
- the two dimensional surface of a unit sphere.
  - latitude and longitude, or
  - x, y, z subject to the constraint $x + y + z = 1$.

The former is an explicitly parametrization whole the latter is an implicitly parametrization. We will use implicit representations of configurations of rigid bodies. A robot arm is typically equipped with a hand, gripper, or other tools, more generally called and end-effector. The space of positions and orientations of a frame attached to this end-effector is called task space. The workspace is the subset of the task space that the end-effector can reach.

Chapter 3: Rigid-Body Motions

This chapter addresses the problem of how to describe mathematically the motion of a rigid body moving in three-dimensional physical space. One convenient way is to attach a reference frame to the rigid body and to develop a way to quantitatively describe the frame's position and orientation as it moves. We introduce a 3×3 matrix representation for describing a frame's orientation; such a matrix is referred to as a rotation matrix.

Chapter 3: Rigid-Body Motions

The most natural and intuitive way to visualize a rotation matrix is in terms of its exponential coordinate representation. Given a rotation matrix R, there exists some unit vector $\hat{\omega} \in \mathbb{R}$ and some angle $\theta \in [\theta, \pi]$ such that the rotation matrix can be obtained by rotating the identity frame (that is, the frame corresponding to the identity matrix) about $\hat{\omega}$ by θ. The exponential coordinates are defined as $\omega = \hat{\omega}\theta \in \mathbb{R}^3$, which is a three-parameter representation.

Chapter 3: Rigid-Body Motions

There are several other well-known coordinate representations, e.g., Euler angles, Cayley–Rodrigues parameters, and unit quaternions, which are discussed in Appendix B.

Chapter 3: Rigid-Body Motions

Another reason for focusing on the exponential description of rotations is that they lead directly to the exponential description of rigid-body motions. The latter can be viewed as a modern geometric interpretation of classical screw theory. Keeping the classical terminology as much as possible, we cover in detail the linear algebraic constructs of screw theory, including the unified description of linear and angular velocities as six-dimensional twists (also known as spatial velocities), and an analogous description of three-dimensional forces and moments as six-dimensional wrenches (also known as spatial forces).

Chapter 4: Forward Kinematics

For an open chain, the position and orientation of the end-effector are uniquely determined from the joint positions. The forward kinematics problem is to find the position and orientation of the reference frame attached to the end-effector given the set of joint positions. In this chapter we present the product of exponentials (PoE) formula describing the forward kinematics of open chains. As the name implies, the PoE formula is directly derived from the exponential coordinate representation for rigid-body motions. Aside from providing an intuitive and easily visualizable interpretation of the exponential coordinates as the twists of the joint axes, the PoE formula offers other advantages, like eliminating the need for link frames (only the base frame and end-effector frame are required, and these can be chosen arbitrarily).

Chapter 4: Forward Kinematics

In Appendix C we also present the Denavit-Hartenberg (D-H) representation for forward kinematics. The D-H representation uses fewer parameters but requires that reference frames be attached to each link following special rules of assignment, which can be cumbersome. Details of the transformation from the D-H to the PoE representation are also provided in Appendix C.

Chapter 5: Velocity Kinematics and Statics

Velocity kinematics refers to the relationship between the joint linear and angular velocities and those of the end-effector frame. Central to velocity kinematics is the Jacobian of the forward kinematics. By multiplying the vector of joint-velocity rates by this configuration-dependent matrix, the twist of the end-effector frame can be obtained for any given robot configuration. Kinematic singularities, which are configurations in which the end-effector frame loses the ability to move or rotate in one or more directions, correspond to those configurations at which the Jacobian matrix fails to have maximal rank. The manipulability ellipsoid, whose shape indicates the ease with which the robot can move in various directions, is also derived from the Jacobian.

Chapter 5: Velocity Kinematics and Statics

Finally, the Jacobian is also central to static force analysis. In static equilibrium settings, the Jacobian is used to determine what forces and torques need to be exerted at the joints in order for the end-effector to apply a desired wrench. The definition of the Jacobian depends on the representation of the end-effector velocity, and our preferred representation of the end-effector velocity is as a six-dimensional twist. We touch briefly on other representations of the end-effector velocity and their corresponding Jacobians.

Chapter 6: Inverse Kinematics

The inverse kinematics problem is to determine the set of joint positions that achieves a desired end-effector configuration. For open-chain robots, the inverse kinematics is in general more involved than the forward kinematics: for a given set of joint positions there usually exists a unique end-effector position and orientation But, for a particular end-effector position and orientation, there may exist multiple solutions to the joint positions, or no solution at all.

Chapter 6: Inverse Kinematics

In this chapter we first examine a popular class of six-dof open-chain structures whose inverse kinematics admits a closed-form analytic solution. Iterative numerical algorithms are then derived for solving the inverse kinematics of general open chains by taking advantage of the inverse of the Jacobian. If the open-chain robot is kinematically redundant, meaning that it has more joints than the dimension of the task space, then we use the pseudoinverse of the Jacobian.

Chapter 9: Trajectory Generation

What sets a robot apart from an automated machine is that it should be easily reprogrammable for different tasks. Different tasks require different motions, and it would be unreasonable to expect the user to specify the entire time-history of each joint for every task; clearly it would be desirable for the robot's control computer to "fill in the details" from a small set of task input data. This chapter is concerned with the automatic generation of joint trajectories from this set of task input data. Formally, a trajectory consists of a path, which is a purely geometric description of the sequence of configurations achieved by a robot, and a time scaling, which specifies the times at which those configurations are reached.

Chapter 9: Trajectory Generation

Often the input task data is given in the form of an ordered set of joint values, called control points, together with a corresponding set of control times. On the basis of this data the trajectory generation algorithm produces a trajectory for each joint which satisfies various user-supplied conditions. In this chapter we focus on three cases:

i point-to-point straight-line trajectories in both joint space and task space;

ii smooth trajectories passing through a sequence of timed "via points"; and

iii time-optimal trajectories along specified paths, subject to the robot's dynamics and actuator limits. Finding paths that avoid collisions is the subject of the next chapter on motion planning.

Chapter 10: Motion Planning

This chapter addresses the problem of finding a collision-free motion for a robot through a cluttered workspace, while avoiding joint limits, actuator limits, and other physical constraints imposed on the robot. The path planning problem is a subproblem of the general motion planning problem that is concerned with finding a collision-free path between a start and goal configuration, usually without regard to the dynamics, the duration of the motion, or other constraints on the motion or control inputs. There is no single planner applicable to all motion planning problems. In this chapter we consider three basic approaches: grid-based methods, sampling methods, and methods based on virtual potential fields.

Chapter 11: Robot Control

A robot arm can exhibit a number of different behaviors depending on the task and its environment. It can act as a source of programmed motions for tasks such as moving an object from one place to another, or tracing a trajectory for manufacturing applications. It can act as a source of forces, for example when grinding or polishing a workpiece. In tasks such as writing on a chalkboard, it must control forces in some directions (the force pressing the chalk against the board) and motions in other directions (the motion in the plane of the board). In certain applications, e.g., haptic displays, we may want the robot to act like a programmable spring, damper, or mass, by controlling its position, velocity, or acceleration in response to forces applied to it.

Chapter 11: Robot Control

In each of these cases, it is the job of the robot controller to convert the task specification to forces and torques at the actuators. Control strategies to achieve the behaviors described above are known as motion (or position) control, force control, hybrid motion-force control, and impedance control. Which of these behaviors is appropriate depends on both the task and the environment. For example, a force-control goal makes sense when the end-effector is in contact with something, but not when it is moving in free space.

Chapter 11: Robot Control

Most practical control schemes compensate for these uncertainties by using feedback control. After examining the performance limits of feedback control without a dynamic model of the robot, we study motion control algorithms, such as computed torque control, that combine approximate dynamic modeling with feedback control.

Questions?

Next:

- 02 - Ch2 Intro, 2.1, 2.2