Chapter 7

In-Class Activity Day 4: Rainbow Road Kick-Off

Today we're going to be starting our first module that introduces some fundamental quantitative representations that we will use in this class to describe physical systems. To ground our work together, we'll be considering the context of *mobile robotics* and developing a controller that will allow a robot to autonomously drive along complex geometric trajectories. The framing of this entire module is introduced in the **Rainbow Road Challenge**, which we invite you to have a look at now!

Learning Objectives

- Reinforce the concept of a sensory-motor loop in robotics.
- Deploy a simple blackbox controller for a Neato.
- Review the definition of a vector.
- Introduce vector representations.
- Practice arithmetic vector operations.

7.1 Meeto Your Neatokart

In the last week, you've seen some of the data collected from a Neato and reviewed the mathematical relationships between position, velocity, and acceleration. Today, you will get to be in the driver's seat as we walk through the steps to collecting your own data with a Neato and performing basic control through MATLAB.

7.1.1 Sensory-Motor Loops Revisited

Recall from Day 1 that robots can be thought of through a framework of sensory-motor loops (see Figure 7.1).

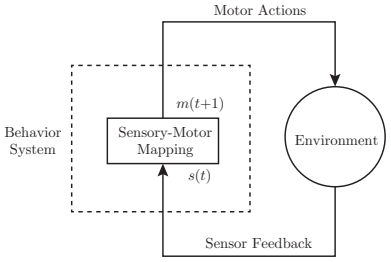

Figure 7.1: A schematic of a robot controlled by a sensory-motor loop.

Your Neatokart is equipped with several sensors with which it can parse the world (see Figure 7.2):

1. A LIDAR laser range finder (a range value between the sensor and an object is provided 360°around the vehicle)
2. 4 Bump sensors (a binary measurement of whether the sensor, which is a switch, is pressed)
3. 2 Encoders (a rotary sensor attached to the wheels which track the linear rotational distance that the wheel has traveled)

And it interacts with the world through its two actuated wheels.

The Neatokart's "brain" is a Raspberry Pi, and it actually serves more as a conduit for information – your laptop will be doing most of the heavy intellectual lifting. The Raspberry Pi accesses the sensor data from the vehicle through a USB serial connection, transmits the data to your laptop over a Wifi network connection, listens for commands sent from your laptop over the Wifi network, and then passes those commands to the wheel motors.

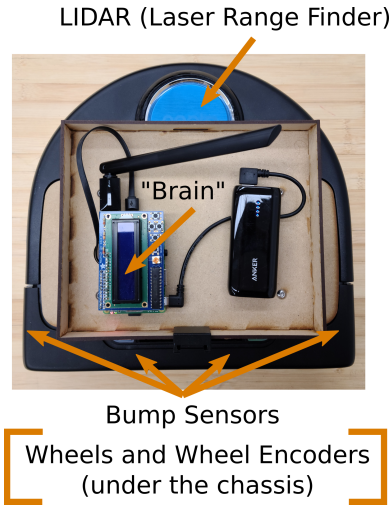

Figure 7.2: A standard Neatokart.

Exercise 7.1

Conceptual Check-In
Draw a conceptual sensory-motor schematic for the Neato. Be specific on what sensors, motors, and computers are being used.

7.1.2 Let's Drive!

At your tables, please each read through the Meeto Your Neato documentation page **in its entirety** before grabbing a Neato for your table. Taking turns, make sure you each practice getting your Neato running, connecting, driving around with a joystick, and shutting down your connection. Members of the teaching team will be floating around if you run into any challenges or bugs along the way!

Exercise 7.2

Document Your Drive
This may be a good time to consider how you want to document your in-class engagement – the Neatos are awfully charismatic after all! Take a picture or generate one of the example plots (or your own) listed on the Meeto Your Neato page.

7.2 [Conceptually] Navigating Rainbow Road

If you haven't had a look at the **Rainbow Road Challenge**, please take a moment to read through it now. Over the next several classes, we'll be building the tools to get our Neatokarts across the finish line. This

section provides a brief conceptual overview to use as a reference as we start digging into our technical material.

On Day 1, we briefly previewed the idea of *differential drive* – for a two-wheeled robot, the difference in velocities between the wheels controls its heading, speed, and turning rate. In our challenge, we are given a particular path for our Neatokart to drive, and have the ability to set the velocities of our wheels. To be successful, we need to precisely control each wheel's velocity at a given time, in order to keep the Neatokart from falling off the sides of Rainbow Road.

At the very fundamental level, this entire module utilizes **vectors** as an object through which we'll reason about our Neatokart's motion. Our topic schedule for the next several classes will be:

1. Week 3, Day 5: **Vectors and Vector-Valued Functions (Parametric Functions)** – How do we parse the Rainbow Road trajectory? How do we relate a trajectory with motion?
2. Week 3, Day 6: **Velocity and Odometry** – Given information about how much our wheels have traveled over time, how do we reconstruct where our Neatokart has been?
3. Week 4, Day 7: **Basis Vectors and Unit Vectors** – How do we connect "where we've been" with "how did we get here"?
4. Week 4, Day 8: **Frames of Reference and Coordinate Systems** – What is the formal motion model of our Neatokart, and how do we translate that to moving along the Rainbow Road?
5. Week 5, Day 9: **Parametric Paths and Encoders** – How do we design a good open-loop controller for our Neatokart?

Exercise 7.3

Clarifying Questions
What questions do you have about the Challenge or the Roadmap ahead? Discuss with your table, and grab a teaching team member to clarify any points of confusion.

7.3 So...What is a Vector?

Precisely, a vector is a quantity with a magnitude (length) and a direction. A vector is usually depicted graphically as an arrow between two points (composing the head and the tail), with the length and direction of the arrow corresponding to the vector's magnitude and direction, respectively.

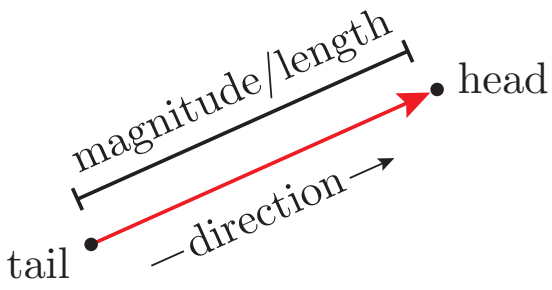

Figure 7.3: A vector can be graphically represented as an arrow between two points (the tail and the head).

Typically, a vector is denoted symbolically using either an overhead arrow $\vec{v}$ or with boldface $\mathbf{v}$ (though other notations exist). The magnitude/length of $\vec{v}$ is written as $|\vec{v}|$ or $||\vec{v}||$. The direction of $\vec{v}$ is given by $\hat{v}$.

We use a **coordinate system** to assign a numerical value to a vector. In a two-dimensional Cartesian coordinate system, $\vec{v} = (v_x, v_y)$ or $\begin{pmatrix} v_x \\ v_y \end{pmatrix}$ denotes the vector whose tail to head displacement has x and y coordinates of v_x and v_y respectively. If we place the tail of the vector such that it coincides with the origin of the coordinate frame, then (v_x, v_y) will denote the position of the head (see Figure 7.4).

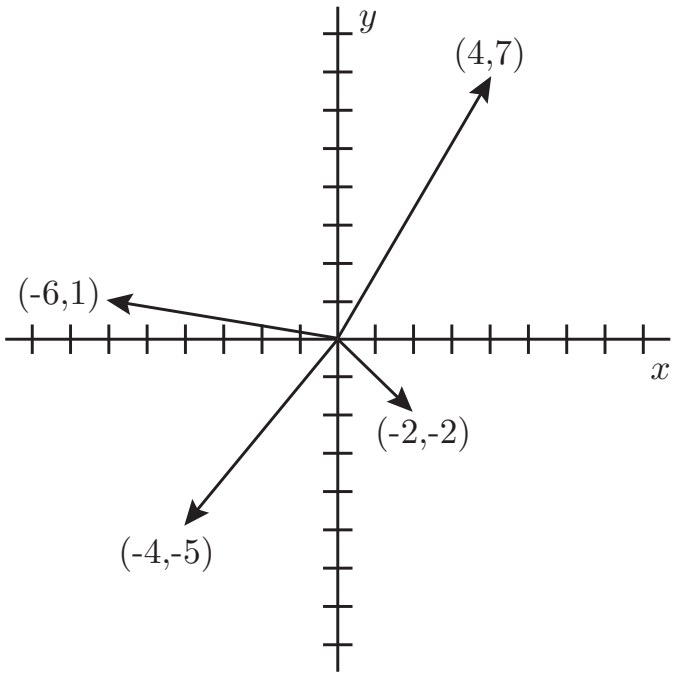

Figure 7.4: The Cartesian coordinates of a vector describe the horizontal/vertical displacement between the tail and head of a vector.

A few things to note:

- Vectors can have any number of dimensions (one, five, or even infinity).
- A **fixed vector** is a vector whose tail and/or head is assigned (affixed) to a particular point (for example, a vector describing the position of an object in space). A **free/floating vector** is a vector that is not associated with any point in particular.
- Vectors are objects that are independent from their numerical representation. The numerical value of a vector will depend on your choice of coordinates.

7.4 Vector Arithmetic

Let's review some of the arithmetic operations that we can perform using vectors.

7.4.1 Scalar Multiplication

The quantity $a\vec{v}$ denotes the product of scalar a and vector $\vec{v}$. Graphically, this corresponds to scaling the length of $\vec{v}$ by a factor of a (see Figure 7.5). Scalar multiplication preserves the direction of the input vector (up to a $+/-$).

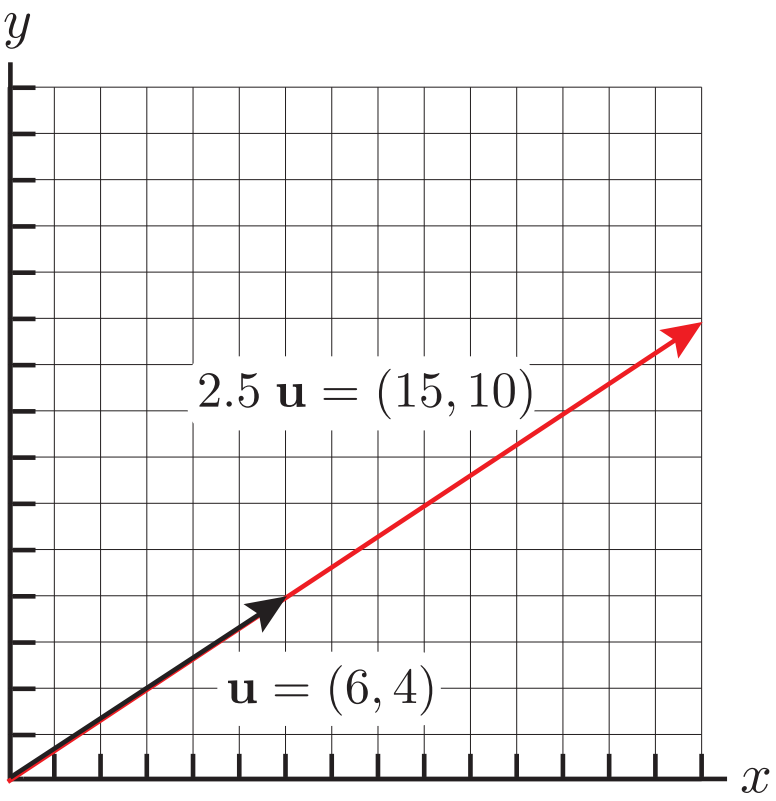

Figure 7.5: Graphically, scalar multiplication corresponds to growing/shrinking the vector by some scalar factor.

To evaluate the product $a\vec{v}$ in Cartesian coordinates, we multiply each component of $\vec{v}$ by a. For example, in 2D, we get:

$$a\vec{v} = (av_x, av_y) \tag{7.1}$$

7.4.2 Addition and Subtraction

The quantity $\vec{v} + \vec{u}$ denotes the sum of vectors $\vec{v}$ and $\vec{u}$. Graphically, we can depict this sum by placing the tail of $\vec{u}$ such that it coincides with the head of $\vec{v}$ (or vice versa), with their sum being the vector connecting the start and end of the combined path (see Figure 7.6).

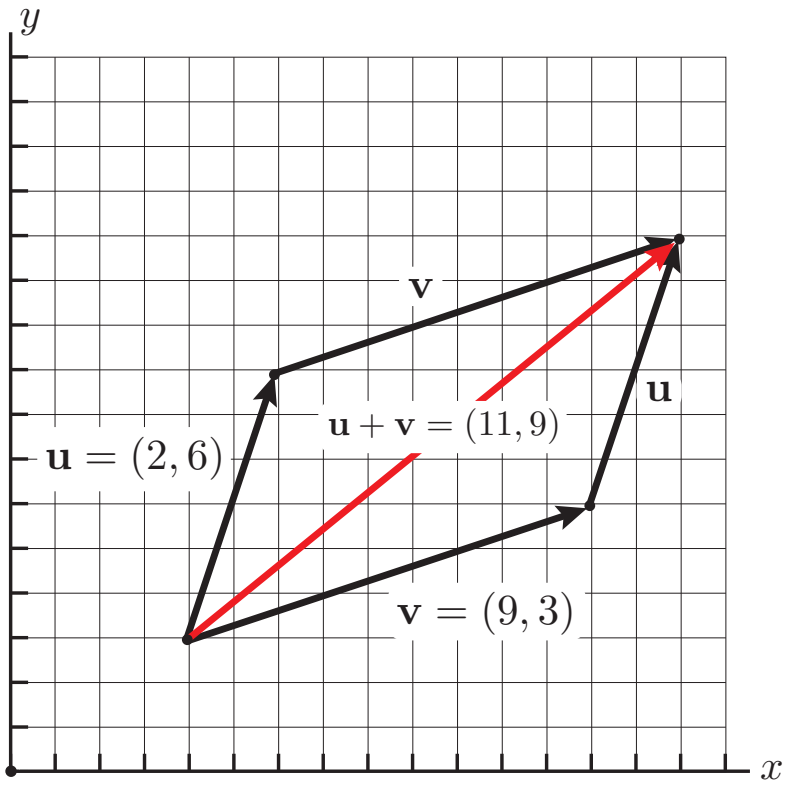

Figure 7.6: Graphically, vector addition corresponds to combining the vectors from head to tail to form a single path.

To evaluate the sum $\vec{v} + \vec{u}$ in Cartesian coordinates, we add each component of $\vec{v}$ to the corresponding component of $\vec{u}$. For instance, in 2D, we get:

$$\vec{v} + \vec{u} = (v_x + u_x, v_y + u_y) \tag{7.2}$$

Vector subtraction is similar to addition. The quantity $\vec{u} - \vec{v}$ denotes the difference of vectors $\vec{u}$ and $\vec{v}$. We can think of vector subtraction as a composition of two operations. First, we multiply $\vec{v}$ by the scalar -1, and then we add the result with $\vec{u}$. As such, we can depict this graphically as the sum of the vectors $\vec{u}$ and $-\vec{v}$ (see Figure 7.7).

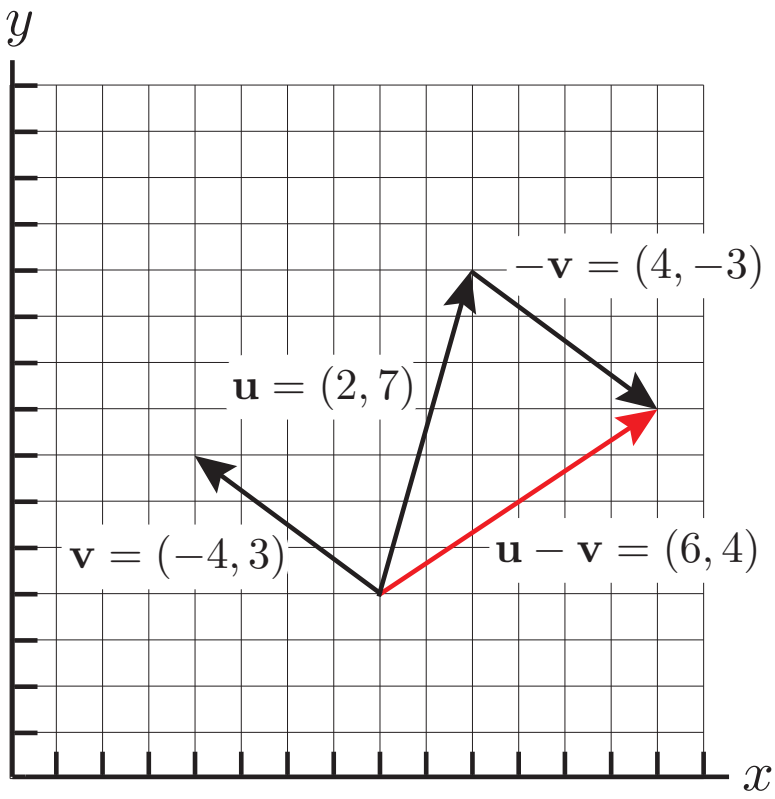

Figure 7.7: Graphically, vector subtraction is the same as addition, except that the subtracting vector has been reversed.

To evaluate the difference $\vec{u} - \vec{v}$ in Cartesian coordinates, we subtract each component of $\vec{v}$ from the corresponding component of $\vec{u}$. For instance, in 2D, we get:

$$\vec{u} - \vec{v} = (u_x - v_x, u_y - v_y) \tag{7.3}$$

7.4.3 Euclidean Norm (magnitude)

The Euclidean norm is how we compute a vector's magnitude/length. In 2D, we can think of the vector $\vec{v} = (v_x, v_y)$ as the hypotenuse of a triangle with base v_x and height v_y. As such, the Euclidean norm of $\vec{v}$ would be given by:

$$||\vec{v}|| = \sqrt{v_x^2 + v_y^2} \tag{7.4}$$

This can be generalized to higher dimensions. If $\vec{v}$ is an N-dimensional vector, then the length of $\vec{v}$ would be given by:

$$||\vec{v}|| = \sqrt{\sum_{i=1}^{N} v_i^2} \tag{7.5}$$

Exercise 7.4

Notation Check-In
We have shown several examples here of 2D vector arithmetic. How would you write the general formulas shown here for vectors in a 3D coordinate system?