**Autonomous Mode Tutorial**

The autonomous timed movement tutorial starts by examining the example VI provided in LabVIEW and then providing insight into implementing this mode for the robot project.

We begin by going over the example VI that shows two methods of implementing autonomous movement. Before we look at the two different methods, we see the left side of the block diagram which sets the stage for the program. The code here determines the type of movement to execute and also provides information being used as references or as debugging tools to monitor the functionality of the VI.

This includes

- Initializing the joystick and giving the joystick value (which is often used as the throttle)
- Configuring movement pattern by reading in what is selected. In this case, the autonomous control was selected from the dashboard
- Showing the driver position and the alliance color
- Confirmation that this autonomous VI is running

![Diagram of Autonomous Movement Tutorial Example: Alternative 1](image)

**Figure 1. Autonomous Movement Tutorial Example: Alternative 1**

Based on the selected moving pattern- the case structure presents two types of movement control. The first alternative uses a For loop to run an array of desired robot movements. The robot will move in the pattern based on the elements in the array. This allows for easier integration with large amounts of movement and outputs current movement for debugging.
The second alternative is a much simpler movement that curves in one direction and then the other. By using the Drive VIs, the robot is instructed step by step to turn in both directions for a certain amount of time. This is a great way to implement simple movement, but is less effective if you have long sequence of motions.

**Figure 2. Autonomous Movement Tutorial Example: Alternative 2**

This example is just the starting point for configuring an autonomous movement. In the previous years’ documents and in the next section of this tutorial, we discuss modularizing the movements by creating subVIs that combines repeating operations of movement into one entity.
Timed Movement VIs

For some background information, the Safety Config VIs were introduced in LabVIEW FRC 2011 to replace the Watchdog Timer used in previous years. When the safety feature is enabled, it is necessary to constantly update the values written to the RobotDrive, MotorControl, PWM, Relay, and Solenoid within the timeout interval (typically 100ms) or the safety mechanism will be triggered and the device will be stopped.

In autonomous mode, it is common to want the device to operate with the same value for a period of time longer than the timeout interval. In this case it is necessary to repeatedly update the device with the same value for the duration of that time. Here is an example program that simply tells the robot to turn right, turn left, and then stop:

As you can see, each movement is accomplished by updating the RobotDrive with the same value in a while loop. The delay time and the number of loop iterations determine how long the RobotDrive will operate with that value. In a simple three-step process like this example, having a loop for each movement is fine. As our code grows in complexity, however, programming the movements like this will quickly become cumbersome. Operations that will likely be repeated throughout your code, such as turning left or turning right, should be turned into subVIs. The image below shows how we took the previous example and modularized it to create our own VIs to interface with the RobotDrive:
These custom VIs simplify the overall programming process by allowing you to simply pass the required parameter(s) to a single VI, in this case the number of seconds to turn left/right or stand still. Alternatively, you could create a more generic movement VI with three inputs for X-axis value, Y-axis value, and the amount of time to drive those values as shown below:

It is up to you to determine whether it makes sense to create individual VIs for each type of movement or the more generic movement VI for your robot project. For simple operations (like turning and stopping) these custom VIs may seem like more work than they are worth, but as your code grows in size and the operations grow in complexity you will quickly realize the value of creating these VIs for specific operations.