UNIVERSITAT POLITÈCNICA DE VALÈNCIA
Antonio Martí Campoy
amarti@disca.upv.es

Complex movements

Methods and Tools for Software Quality

ESIGELEC

Introduction.

In previous exercises you built functions to perform basic movements. Now it is time to create an array to store and then execute a concatenation of different basic movements. Using this array will allow us to simplify the main loop. We will work only with non-blocking movements.

The first problem you have to deal with is where and how to store the list of movements you want to concatenate. Of course it will be an array but, what kind of elements would be stored in this array? Yes, it must be some kind of structure.

First of all, you need a movement identifier. That is, a code to say which kind of movement (straight, spin by steps, spin by bearing, turn by steps or bearing, spiral or stop) the robot has to do.

Then, you need to store together with the identifier the parameters related to the movement: distance to run, bearing to match, and even time the robot must be stopped.
Not all the movements need the same parameters, so you have to choose between a unique, big structure with as many fields as parameters for all the movements, or a *union* of structures, with one structure for each movement.

Do you know the difference between structure and union? Which do you think is better?

If you use a union, you will save memory. It is a nice idea for embedded systems because in several cases the amount of memory of the microcontroller is small. However your code will be a bit complex, mainly when you want to access a parameter inside a structure inside a union inside and structure being an element of an array. Did you need to put the union inside a structure?

If you use a structure, you will waste memory of the microcontroller, but your code will be simpler, because you will have to access just to a field belonging to an array.

Choose the option you prefer and create, in a new .h file the structures you need. Start working with two or three movements, for example straight move, spin_steps and spin_bearing.

Finally, in function main, declare an array of movements and initialize it with three or four movements. You will need to reuse the infinite loop created in previous exercises.