Robot Documentation

Alexander Greus, Kenneth Thompson, Dylan Zemlin, Spencer Smith

September 26, 2025

1 Reactive Architecture

1.1 Chosen Architecture

Our project implements the subsumption architecture. This architecture follows the reactive control paradigm and is designed around a series of layers that the robot will follow. Each layer can be considered a behavior, where each behavior has some priority such that the behavior list can be simply declared as a priority list. So, the higher priority behaviors are capable of overriding the controls of lower priority behaviors. This was a clear choice for this project as the entire concept of it was behavior based design and priority. Additionally, we chose this because we did not need anything other than reactive due to no need of mapping (at least no need to use the map that we generated).

1.2 Subsumption Architecture Code

The implementation of subsumption can be found in a few different places

- **Behavior Priority List** In the main loop of the primary, a list of behaviors is checked in the order listed by the assignment. The first behavior that triggers a non-zero movement command (e.g. it has a velocity or rotates) takes control of that loop iteration and will prevent any further behaviors from triggering. For example the BumperBehavior prevents motion immediately if a collision is detected and prevents any further behaviors from running.
- **Object Oriented Behaviors:** Each behavior is represented as a class (BumperBehavior, EscapeBehavior, AvoidBehavior, RandomTurnBehavior, ForwardBehavior), inheriting from a base Behavior class. This design makes it incredibly simple to implement the chosen architecture as we can easily put a list of behaviors together that can be ran using the same line of code (behavior-¿run()).
- **Sensor Data:** Each behavior has access to a context objet that contains the latest sensor reading and global state information such as whether or not the robot is escaping, etc.

2 Algorithms and Behaviors

The behaviors as listed below are in order from highest priority to lowest priority (e.g. bumper is highest).

2.1 Bumper Behavior

The bumper sensor detects direct collisions with obstacles. If activated, this behavior issues an immediate stop command and disables teleoperation.

2.2 Escape Behavior

When the robot is blocked both left and right, it performs an escape routine that is implemented by selecting a random turn angle (using uniform sampling as per the instructions) to rotate approximately 180 degrees away from the obstacle. The escape behavior remains active until the robot completes its turn.

2.3 Avoidance Behavior

If an obstacle is detected on one side or directly in front, the avoidance algorithm determines an angular velocity to steer the robot away. This is done by comparing distances measured by the laser scanner on the left and right sides, then generating a turn speed.

2.4 Random Turn Behavior

To prevent the robot from becoming stuck in repetitive left/right movements the random turn behavior occasionally interrupts forward motion. After traveling a certain distance a random small adjustment which prevents the robot from getting stuck.

2.5 Forward Behavior

When no other behavior is triggered the robot defaults to forward motion.