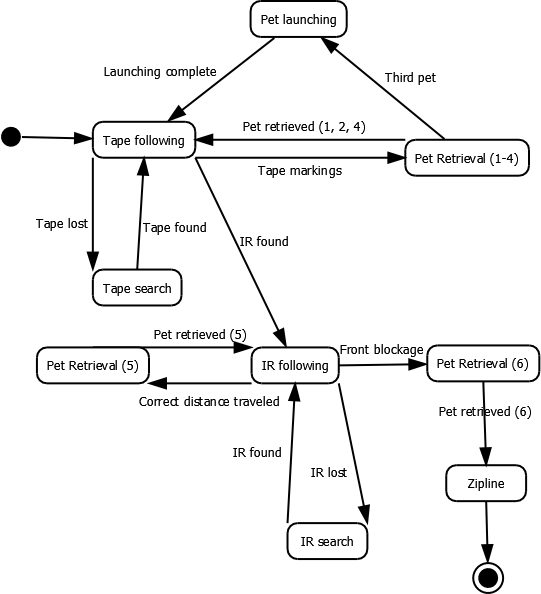
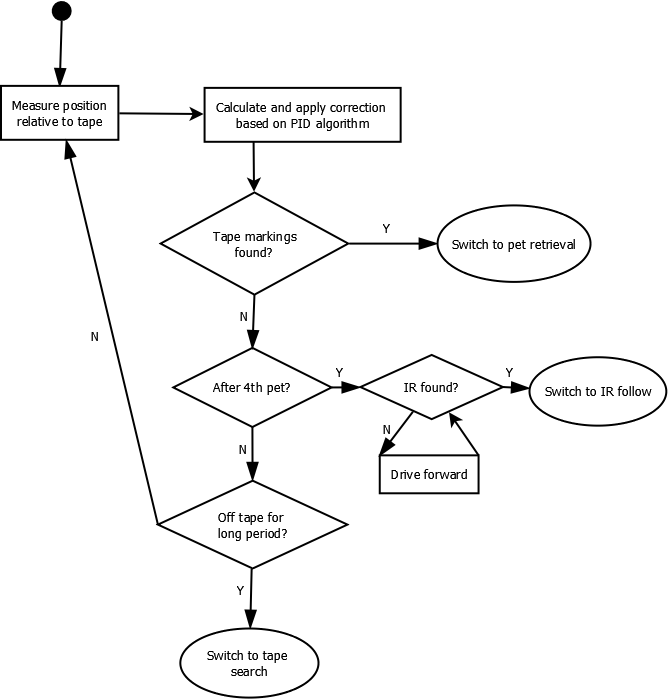
Our strategy for the software design is to make it modular, tracking the current state of the robot (including parameters like pets retrieved and approximate location on the track) in order to aid in decision-making.

6.1 High-Level State Diagram

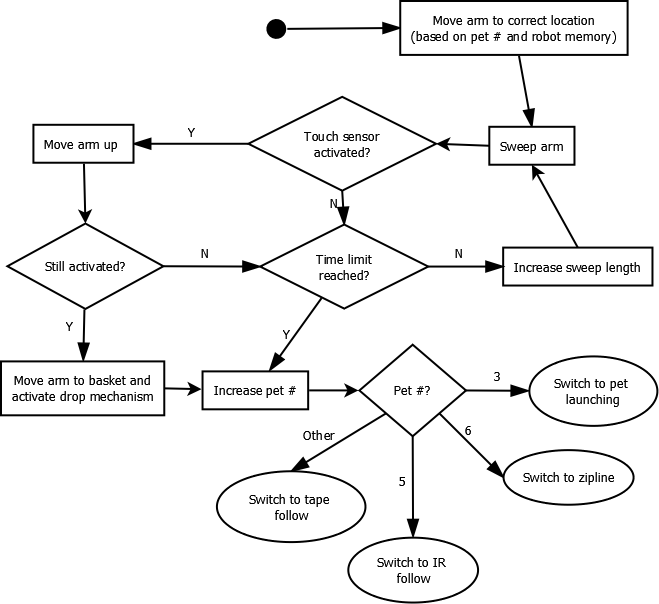


The three most important stages of the flow are the tape following, IR following, and ziplining (with pet retrieval interspersed). Each of the individual states is documented below.

6.2 Tape Following Control Flow

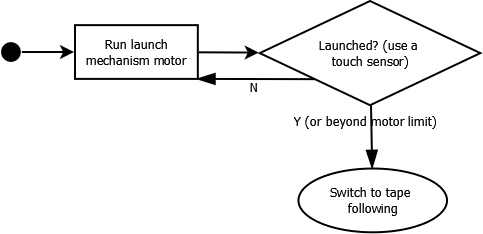
Tape following will be handled using the PID control algorithm described in class: this is not actually shown in the flowchart, but is wrapped into the correction calculation step. The robot will check for tape markings often (as often as required to always catch them after testing): it will check for IR and tape less often in order to make the algorithm run faster.

6.3 Pet Retrieval Control Flow



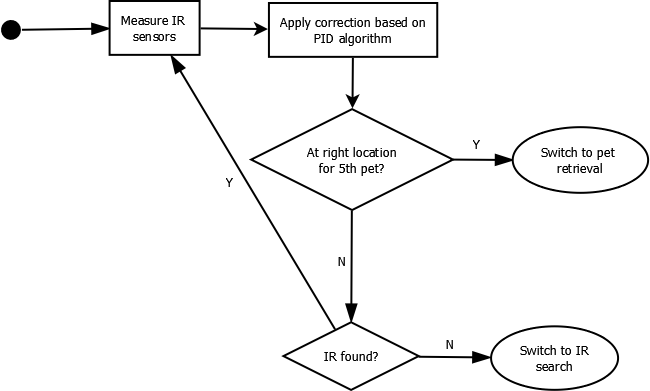
For the first step, the correct location of the arm for each pet will be hardcoded, and then the robot will search around this location using arm sweeps. If the pet isn't caught after each sweep, the robot will continue sweeping (in larger and larger areas) until a time limit (TBD) is reached, in which case the robot will just skip the pet. The robot will also make sure that any sensor activations aren't spurious, especially in the case of the pet covered in foam, by lifting the arm and rechecking the sensor. An important factor is the tracking of how many of the pets have been picked up: this number may be modified (since the robot may miss a pet altogether) by the robot's knowledge of its location on the course, and of the differing characteristics of each pet.

6.4 Pet Launching Control Flow



Pet launching is intended to be as simple as possible: the launch mechanism motor will simply run until it detects that launching has occurred (or that it has run more than a hardcoded limit, in which case it will assume launching has occurred).

6.5 IR Following Control Flow



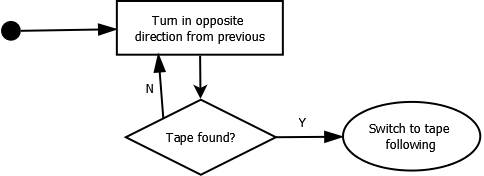
The IR algorithm is fairly similar to the Tape following algorithm, with the key difference that the inputs are analog instead of digital: this simplifies the algorithm somewhat as keeping track of the derivative is simplified with a more-or-less continuous function. In this case, the error function is based on the difference of the values between the IR sensors.

6.6 Zipline Control Flow

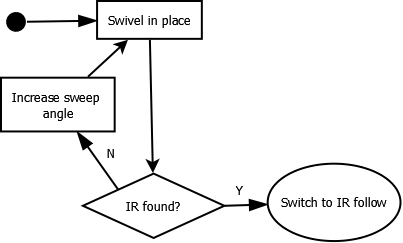
C:\Users\Gregory\Documents\GitHub\enph253-2015-team13\Robot\State Diagrams\Zipline State Diagram.png

Ziplining is fairly simple, as there is no need to control a zipline arm: the basket will clip itself onto the zipline as it is lifted. The robot simply fully lifts the accordion flexure and then lowers it again.

6.7 Error Handling



If the robot hasn't found tape for a while, it will guess that it missed a crossing of the tape and will turn in the opposite direction that it was turning in at a slow speed until it finds tape again. In order to avoid this problem altogether, proportional gain will be prioritized over speed so that the robot corrects itself quickly (as the robot doesn't need to be very fast).



If the robot loses the IR signal, it will make successively larger sweeps in place until it finds IR.