Byron Laferriere

IT-209

Dr. Ostrowski

Analysis Paper

# Applicability of Control Systems

## Open Loop

## The open loop control system involves providing a robot with a world map that it is to use to navigate its way through. This type of control system requires that the robot operate in an unchanging environment. Because of these limitations, an open loop controller is seen as the simplest form of control system. The programmer issues a command and hopes that the system obeys it (Jones & Roth, 2004). In an open loop controller, the robot only pays attention to what it is told to do and not what it encounters in its environment or what the robot actually does in response to the command.

## Closed Loop

A closed loop controller is used in situations where the programmer needs the robot to perform as commanded, even when the conditions change (Jones & Roth, 2004). These types of controllers succeed in dynamic environments that require the robot to adapt to moving objects and changing locations. Instead of blindly following the commands given, the robot is also aware of its own personal actions and whether or not it is completing the task it was assigned. This allows the robot to make changes to the actuators in order to complete the task accordingly.

## Open Loop w/ Parameters and State

To address the weaknesses originally stated with open loop controllers, this alternative controller was also mentioned. In an open loop controller, a programmer can add complexity to the original controller, that can make it perform in a more dynamic manner than simply completing a task. Adding additional parameters to an open loop can increase the appearance of dynamic behavior in robots but would still require a static setting to support the dynamic behaviors.

## Bang-Bang Controllers

## This controller is commonly used in combination with mobile robotics and is also associated with the title of proportional control (Jones & Roth, 2004). Simply designed, the bang-bang controller needs only one bit of input to be effective (Jones & Roth, 2004). These controllers are best used with situations that require hysteresis to be used. The controller has two simple states which control things like the gain parameter and the arc the robot follows (Jones & Roth). Using these state controls with hysteresis to keep a median line and never deviate too far, the bang-bang controller offers simple robot task management capabilities.

# Applicability of Arbitration Approaches

## Fixed-Priority

Fixed-Priority arbitration involves establishing a behavioral hierarchy in robotic programming, which defines the behavior that is to be listened to when two behaviors are demanding resources simultaneously. If the commands given by the behaviors contain contradictory messages to the robot, arbitration is necessary. Fixed-priority arbitration requires that there be a behavior that is labeled as the most important behavior to follow, then from there, other behaviors can be labeled according to importance. When multiple behaviors send their commands to the arbiter at once, the pre-determined order of importance can easily be followed by the robot. This arbitration method essentially declares a permanent “winner” when behavioral conflict occurs (Jones & Roth, 2004).

## Variable-Priority

Variable-Priority arbitration requires the definition of many different scenarios and how the robot should behave differently in each. Instead of having one pre-determined “winner” when behavioral conflict occurs, variable-priority assigns a different winner to each scenario. While these schemes are workable, in robotics, they are rarely ever worth the trouble (Jones & Roth, 2004). This method of arbitration requires much more coding to be done and is very time consuming. One thing to be noted also, is the way a system works is not continuous and can shift from moment to moment, making what the robot is doing and why out to be very challenging to pre-determine (Jones & Roth, 2004).

## Subsumption Architecture

This arbitration technique was inspired by the development of the brain over the course of evolution (Jones & Roth, 2004). There are lower, primitive behaviors assigned to the robot that are much alike human motor functions and are never lost (Jones & Roth, 2004). From here, higher-level functions can be added to these foundational behaviors in a building block format. The primitive behaviors are always programmed first and then subsumed by new behaviors if modifications need to be made for functionality (Jones & Roth, 2004). Subsumption architecture can be used to address the issue of connecting behaviors using the explicit constructions known as wires (Jones & Roth, 2004).

## Motor Schema

Known for its inclusive way to combine conflicting behavior outputs, the motor schema arbitration method uses compromise to navigate its way through conflict issues (Jones & Roth, 2004). In this schema, vector fields are used by the robot to identify obstacles possessing a repulsive potential in the field. By casting all constraints on the robot in a similar language, the motor schema arbitration makes the discovery of optimal solutions easier for the robot in the field (Jones & Roth, 2004)

## Least Commitment

Rather than allowing a single behavior to determine what the robot should not do, least commitment arbitration calculates what the robot should do by detecting unsafe actions (Jones & Roth, 2004). It does not allow one behavior to simply take over control of the robot when reacting, instead it picks the safest translations and rotations to be used that are closest to the motions desired by the highest-priority goal-seeking behavior (Jones & Roth, 2004).

# Recommendation

This cat toy robot would need to possess a closed loop control system that would allow it to adapt to the dynamic environment surrounding it (Jones & Roth, 2004). With the unpredictable movements of the cat and how furniture can be rearranged, or new obstacles like shoes, the robot will need to possess a control system that allows it to overcome ever-changing disturbances. It seems that sticking to a simple fixed-priority arbitration scheme would be the best route. In the end, no matter what the scenario the robot is to encounter, the same behavioral guidelines could be followed, as long as they were well-defined in advance. Sticking to a simple foundational design, will allow for me to adopt a subsumption architecture in my design to update behaviors as necessary.

# 

# Challenges

Some challenges I foresee occurring during this project involve the random combination of features/functionality that I hope to include. I hope to combine the three features available with the Finch robot: lights, noises, variable bursts, in a random order continuously when in use. This design choice was chosen in hopes of keeping the cat entertained for a longer period of time. If the same squeak and light happened every time the cat approached the toy from the right, the animal could quickly become uninterested in the toy. I’m thinking that I should be able to create a loop, however, that randomly assigns light color, speed, and noise to each encounter with the cat. I do think that this could be an issue because of my limited knowledge in the commands that the Finch robot can receive. However, I hope to be able to overcome this challenge for the final project.

# 

# Reference

Jones, J. L., & Roth, D. (2004). *Robot programming: A practical guide to behavior-based robotics*. New York: McGraw-Hill.