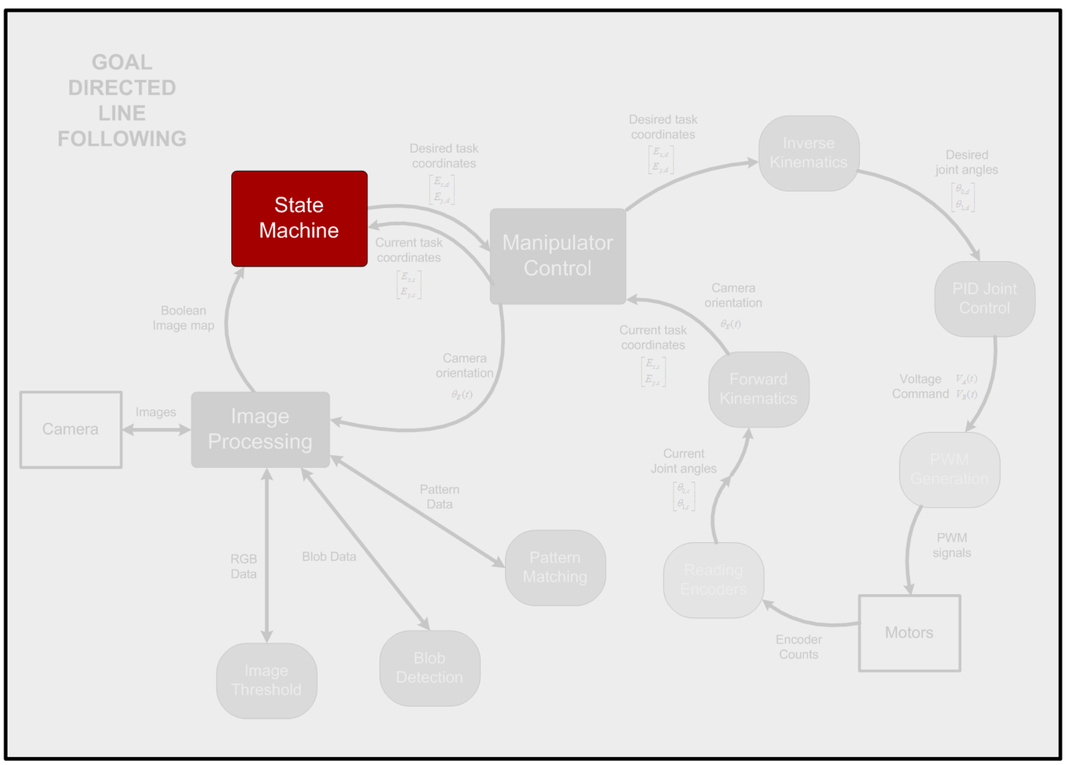
State Machines

Topics Covered

- Using State Machines for autonomous program execution.
- Using a State Machine with the QNET Mechatronic Systems.

Prerequisites

- The QNET Mechatronic Systems is set up according to the Quick Start Guide.



1 Background

A state machine is an abstract machine, that can be in one of many but finite defined states. The next state depends on the current state, in-state calculations or the current value of it's inputs. The state machine should always have a beginning state, and an end state. The remaining states should always have at least one exit state other than the same state itself. A state flow diagram can represent the behaviour of a state machine, as shown in Figure 1.1.

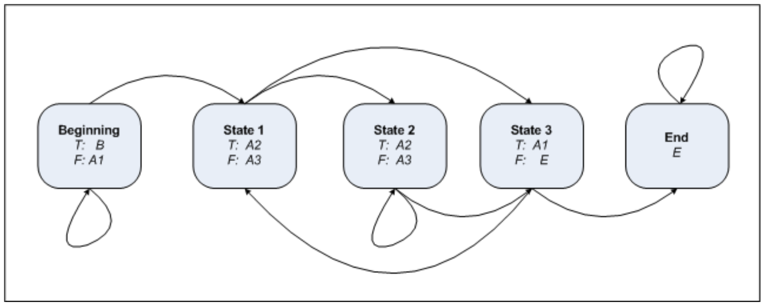


Figure 1.1: State Flow Diagram

A Beginning state can take the system to State 1 if an internal condition is TRUE, or keep it in the Beginning state otherwise. Similary, State 1 can proceed to State 2 or State 3. State 2 can proceed to State 3 if a condition is FALSE, or remain in State 2 if TRUE, the latter being a case in which either a pause is required, or the system waits for an event to occur. State 3 either loops back to State 1 or proceeds to the End State, remaining there.

State machines are used in logic circuits, describing and designing neurological systems, traffic lights, combination locks, washing machines, vending machines, etc., in all of which, there are predefined states that a system must autonomously traverse through. Figure 1.2 shows an example of a simplified traffic light state machine.

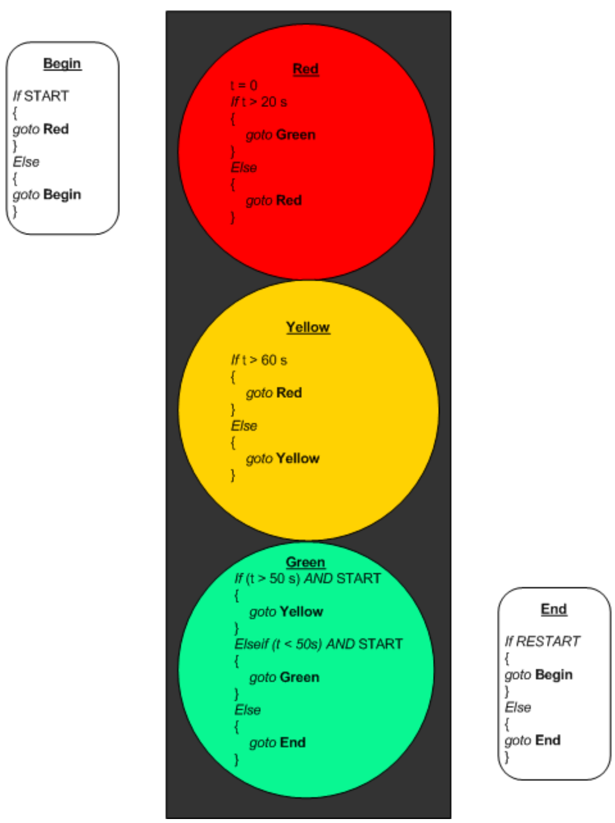


Figure 1.2: Traffic Light State Machine

2 State Machine Challenge

In this lab, a simple light modification state machine is presented which obeys the following rules,

- All traffic light LED groups should light up either green or red at any given time.
- All traffic light LED groups should light up red initially.
- When the manipulator moves within range of a traffic light and detects a red LED, the traffic light in question should turn green and remain there.
- When the manipulator moves within range of a traffic light while all the traffic lights are green, all the traffic lights should turn red.
- A traffic light that has changed colour while the manipulator is in it's range, cannot change colour again until the manipulator moves out of range of all traffic lights, and re-enters.

Thus, when all the third traffic light LED turns green, the manipulator must be moved away from it, and brought back into the range of one of the traffic lights, to turn them all red. Following this change, the traffic light in question cannot turn green according to the normal operation. The manipulator must be moved away and brought back into range to resume normal operation.

This state machine is designed using 4 states, including the Initialize (begin) and End states, as shown by Figure 2.1.

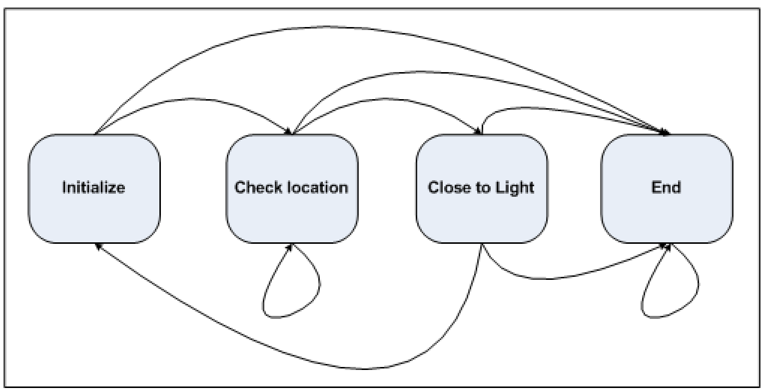


Figure 2.1: Light modification state machine

The conditions under which the states change depend on three major variables, STOP, IN_RANGE and ALL_GREEN. When the user presses the STOP button, the state machine should immediately go to the End state. IN_RANGE is TRUE if the manipulator is in range of a traffic light on the QNET Mechatronic Systems silk, and FALSE otherwise. Lastly, ALL_GREEN is TRUE if all the traffic lights are green, and FALSE otherwise. The state change is summarized in Table 2.1.

↓Current | Next→	Begin	Check location	Close to light	End
Begin	X	STOP	X	STOP
Check location	X	STOP, IN_RANGE	STOP, IN_RANGE	STOP
Close to light	ALL_GREEN	ALL_GREEN	X	STOP
End	X	X	X	⊗

Table 2.1: Light modification state changes, variables in green are TRUE, and red are FALSE

In LabVIEW™, state machines are implemented as a case structure within a while loop. Inside each condition of the case structure, the next state is passed as a state register. This is read at the next iteration and so on.

2.1 Stage 1: Proximity to a Traffic Light

1. Open `Mechatronic Systems.lvproj`, and under `Quanser ELVIS RIO | Subsystems`, open `State Machine.vi`. Run the VI. After the Calibration bar is full, move the manipulator manually till one of the traffic light LEDs is at the centre of the image. Note the joint angle coordinates. Repeat this for the other two traffic lights shown in Figure 2.2.

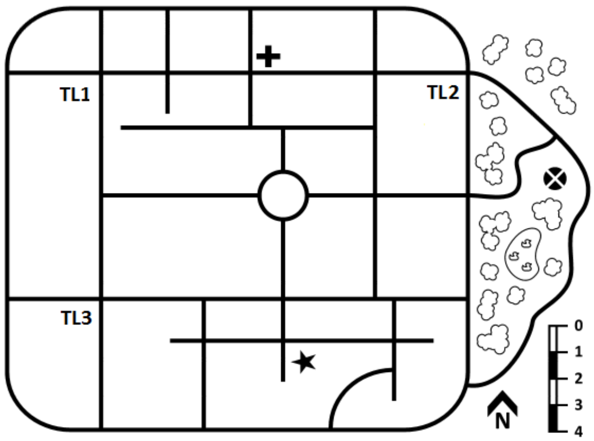


Figure 2.2: Traffic light locations on QNET Mechatronic Systems silk

2. Open the block diagram in `State Machine.vi`. In the while loop labelled as State Machine, browse to the case Check location. This state takes the current joint position vector from the Manipulator Control loop, and checks if it is within range of the traffic lights. Use your answer for the joint positions from the previous step with ± 0.15 rad to fill in the upper and lower range for each traffic light (labelled TL1 - L, TL1 - U, etc.).

2.2 Stage 2: Turning Traffic Lights Green

Browse to the case Close to light. Within this case structure, there are two more case structures nested within each other, which provide the required functionality. Note that the LED lights on the QNET Mechatronic Systems correspond to the convention shown in Table 2.2. The outer structure See a red light? depends on whether a red light is seen by the QNET Mechatronic Systems camera. The inner case structure Which red light? depends on which of the three traffic lights is detected.

	Red	Yellow	Green
TL 1	0	1	2
TL 2	3	4	5
TL 3	6	7	8

Table 2.2: Traffic light LED index on the QNET Mechatronic Systems

1. In the three cases of the case structure Which red light?, the green boolean array input refers to the state of the QNET Mechatronic Systems LEDs, as described in Table 2.2. In this case structure, the correct red light must be turned off, and then the corresponding green light turned on. Modify the three cases to enable this functionality by modifying the 0 constant inputs to the correct indices from Table 2.2.

Note that the code shown in Figure 2.3 takes every third LED light (i.e. the green LEDs) from the array of the QNET Mechatronic Systems's LEDs, and checks if they are all green. In case they are green, the next state is selected as Initialize, where all the LEDs are turned back to red. Note that if the third traffic light is just turned Green, Close to light will not register this change in the current iteration, and will still proceed to the Check location state. Check location checks not only whether or not the manipulator is in range of a traffic light, but also if it was during the previous iteration, and won't proceed to Close to light unless the previous state was not in range of a traffic light.

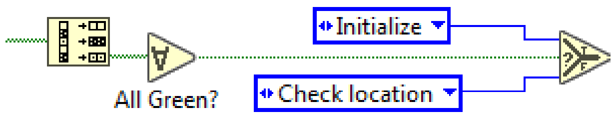


Figure 2.3: Checking if all of the QNET Mechatronic Systems's traffic light LEDs are green

2.3 Stage 3: Full Functionality of the State Machine

1. Run the VI with the correct joint positions in place within the Check location state, and the correct lights turned ON or OFF in the Close to light state. Does the system behave as expected?

Quanser Inc.
119 Spy Court
Markham, Ontario
L3R 5H6
Canada
info@quanser.com
Phone: 1-905-940-3575
Fax: 1-905-940-3576

Printed in Markham, Ontario.

For more information on the solutions Quanser Inc. offers, please visit the web site at:
http://www.quanser.com