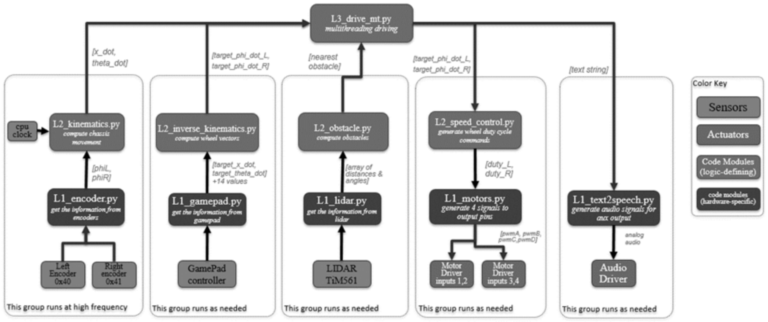
Scuttle Software Guide, Kinematics & Obstacle Avoidance

revised 2019.08.06

TEXAS A&M UNIVERSITY

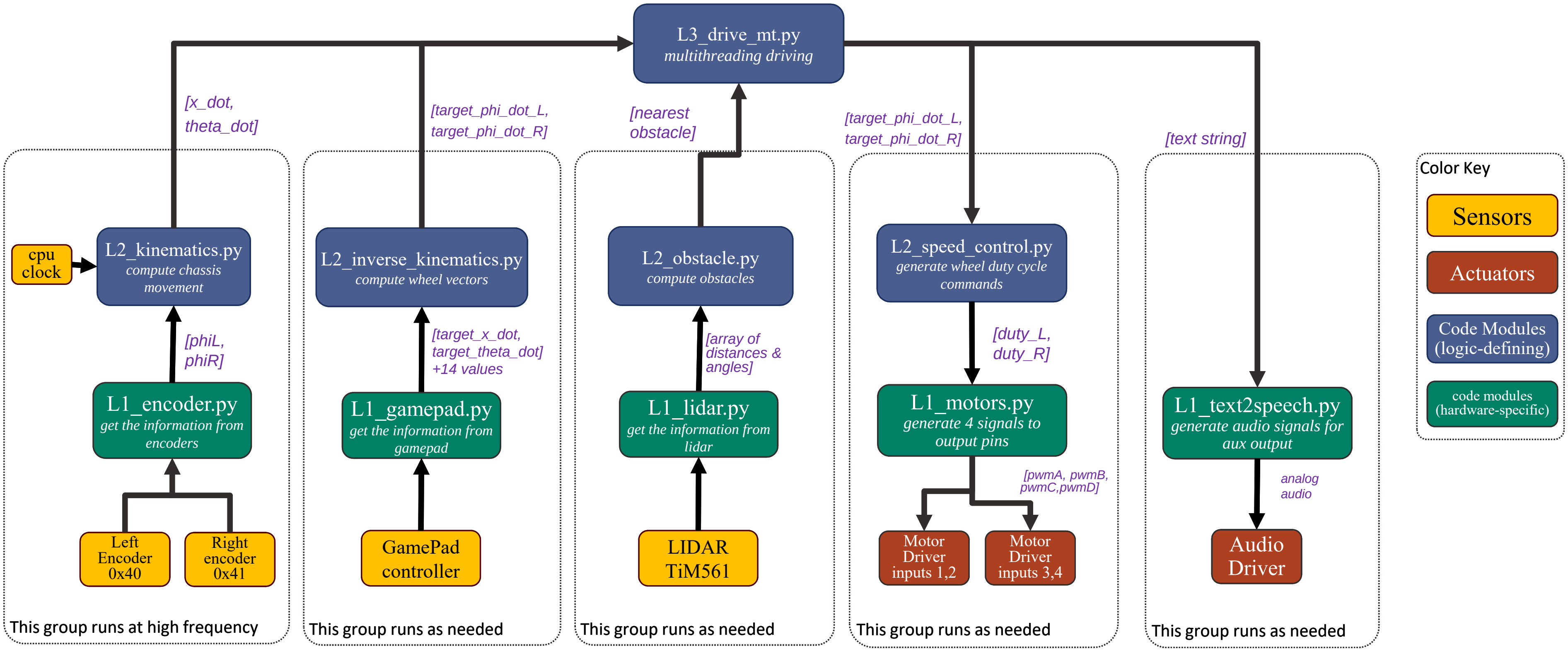
Software Architecture

- This section covers:
  - The parts of each software file
  - How the programs interact with each other
  - How the programs interact with hardware
  - Sensor software vs actuator software



L3_drive_mt.py
multithreading driving
[x_dot, theta_dot]
[target_phi_dot_L, target_phi_dot_R]
[nearest obstacle]
[target_phi_dot_L, target_phi_dot_R]
[text string]
cpu clock
L2_kinematics.py
compute chassis movement
[phiL, phiR]
L1_encoder.py
get the information from encoders
Left Encoder 0x40
Right encoder 0x41
This group runs at high frequency
L2_inverse_kinematics.py
compute wheel vectors
[target_x_dot, target_theta_dot] +14 values
L1_gamepad.py
get the information from gamepad
GamePad controller
This group runs as needed
L2_obstacle.py
compute obstacles
[array of distances & angles]
L1_lidar.py
get the information from lidar
LIDAR TiM561
This group runs as needed
L2_speed_control.py
generate wheel duty cycle commands
[duty_L, duty_R]
L1_motors.py
generate 4 signals to output pins
[pwmA, pwmB, pwmC,pwmD]
Motor Driver inputs 1,2
Motor Driver inputs 3,4
This group runs as needed
L1_text2speech.py
generate audio signals for aux output
analog audio
Audio Driver
This group runs as needed
Color Key
Sensors
Actuators
Code Modules (logic-defining)
code modules (hardware-specific)

Software Architecture - Overview



L3_drive_mt.py
multithreading driving

[x_dot, theta_dot]

L2_kinematics.py
compute chassis movement

cpu clock

[phiL, phiR]

L1_encoder.py
get the information from encoders

Left Encoder 0x40

Right encoder 0x41

This group runs at high frequency

[target_phi_dot_L, target_phi_dot_R]

L2_inverse_kinematics.py
compute wheel vectors

[target_x_dot, target_theta_dot] +14 values

L1_gamepad.py
get the information from gamepad

GamePad controller

This group runs as needed

[nearest obstacle]

L2_obstacle.py
compute obstacles

[array of distances & angles]

L1_lidar.py
get the information from lidar

LIDAR TiM561

This group runs as needed

[target_phi_dot_L, target_phi_dot_R]

L2_speed_control.py
generate wheel duty cycle commands

[duty_L, duty_R]

L1_motors.py
generate 4 signals to output pins

[pwmA, pwmB, pwmC,pwmD]

Motor Driver inputs 1,2

Motor Driver inputs 3,4

This group runs as needed

[text string]

L1_text2speech.py
generate audio signals for aux output

analog audio

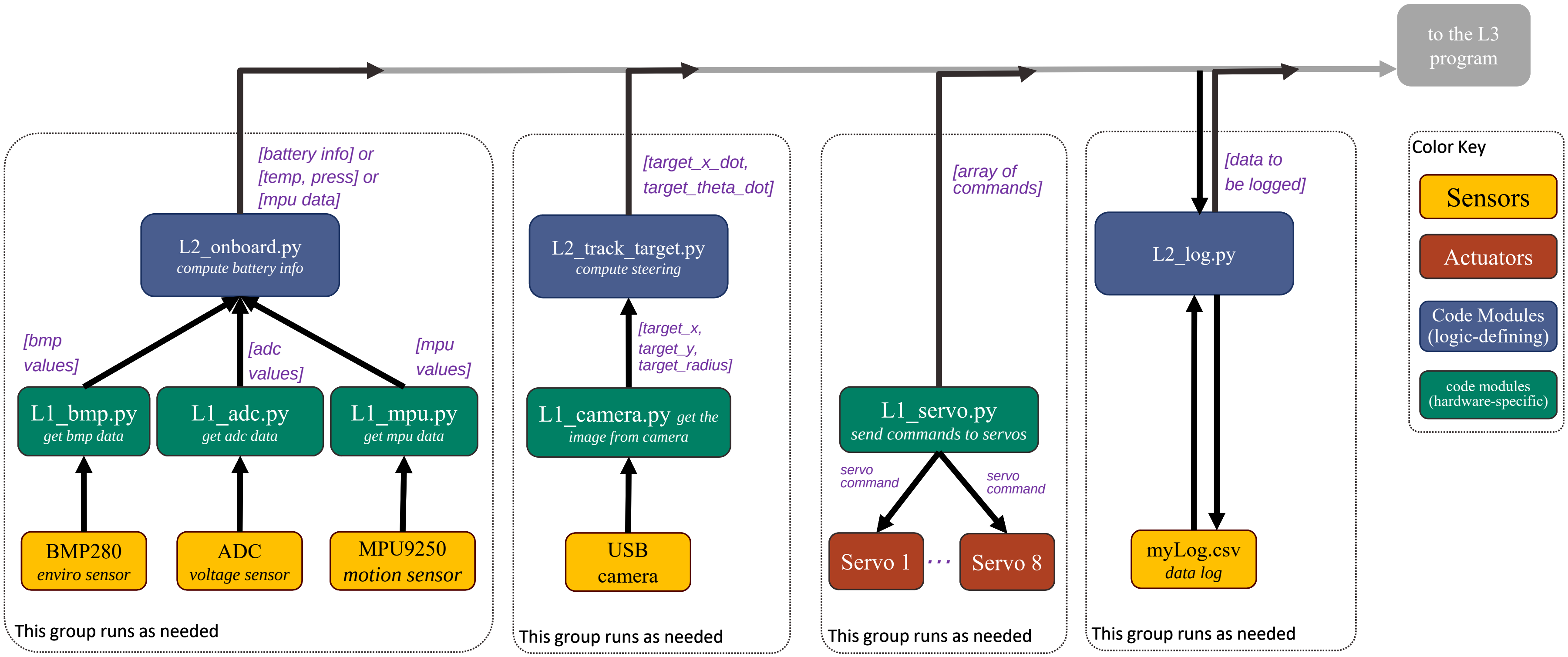
Audio Driver

This group runs as needed

Color Key

- Sensors
- Actuators
- Code Modules (logic-defining)
- code modules (hardware-specific)

Software Architecture – Overview (continued)



Software Architecture - Level1 Program Example

Explanation of what the code does

Import External programs, then internal (if applicable)

Define constants and
take initialization actions

Define functions.
In some cases, make functions that
combine other functions in sequence.

Offer a simplified, minimal
loop for testing the code

Software Architecture: SCUTTLE Threading Purpose

- Threading offers better control over **timing of code execution**.
- Each thread should contain **actions that are related** and that should be executed within a specific time window.
- The user should avoid passing data between threads because it reduces robustness. Instead, **call the level 2 program as needed in each thread**, even if you need to communicate with the same device (ie, retrieve gamepad commands for driving and retrieve in parallel for speaking commands)

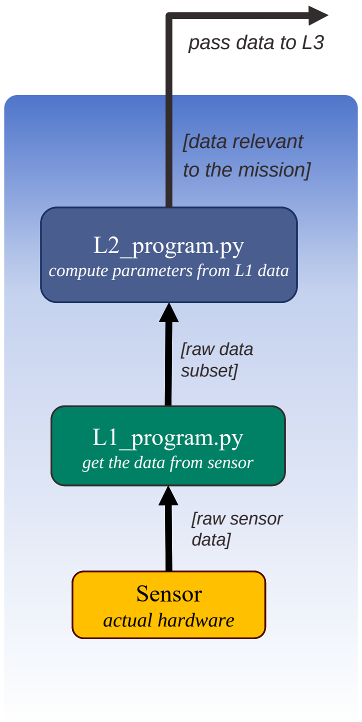
	Thread 1 (driving)			Thread 2 (obstacle detect)	Thread 3 (speaking)
Do What?	Drive the robot	AND	Log the speed	Indicate obstacles using speaker	speak via speaker
When?	After sampling the controller		After sending drive command	when obstacle is detected by lidar	user presses button
	20ms cylcle			10ms cylcle	5ms cylcle

Software Architecture: Tiers Guidelines

Importing
Files & Libraries

Only import L1 programs directly below. Do not import L3 program. As needed, import external libraries.

Do not import L2 or L3 programs. As needed, import external libraries.



pass data to L3

[data relevant to the mission]

L2_program.py
compute parameters from L1 data

[raw data subset]

L1_program.py
get the data from sensor

[raw sensor data]

Sensor
actual hardware

Testing
Code and Hardware

This file has a loop to test the whole group (L1 + L2 + sensor).

This file has a loop to test sensor function.

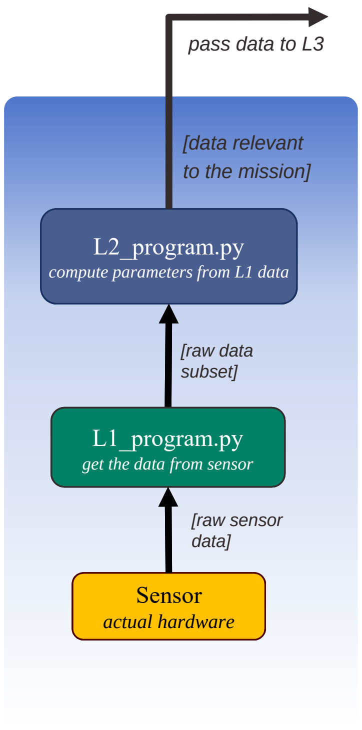
Software Architecture: Sensors vs Actuators

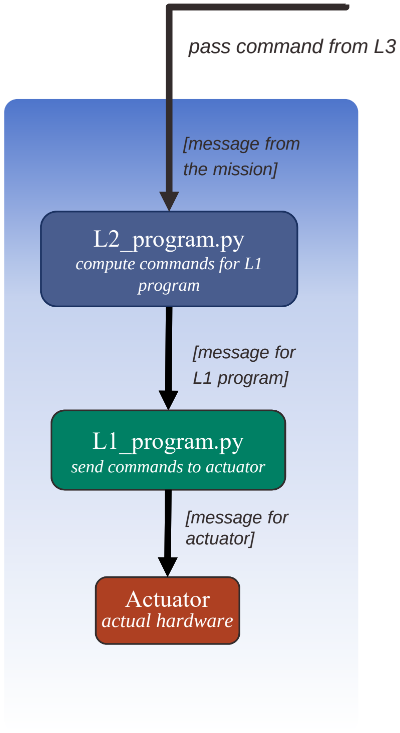
Sensor and Actuators have the same architecture except for **data direction**.

For **sensors**, the data is generated at the hardware and sent UP.

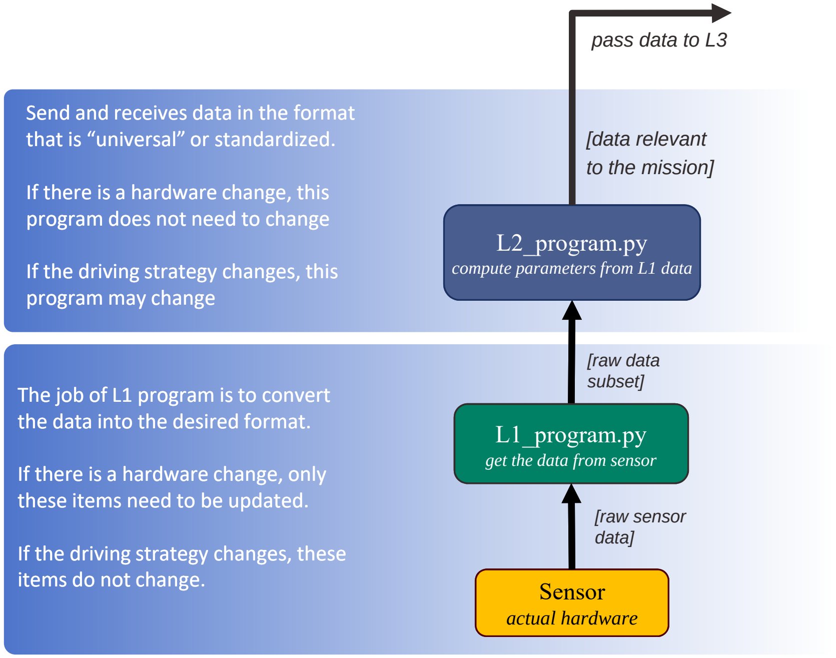
For **actuators**, the data is generated at the top and sent DOWN to hardware.

Some sensors and actuators have feedback and preset commands, so data may flow both ways.





Software Architecture: Modularity & Robustness



Send and receives data in the format that is “universal” or standardized.

If there is a hardware change, this program does not need to change

If the driving strategy changes, this program may change

The job of L1 program is to convert the data into the desired format.

If there is a hardware change, only these items need to be updated.

If the driving strategy changes, these items do not change.

pass data to L3

[data relevant to the mission]

L2_program.py
compute parameters from L1 data

[raw data subset]

L1_program.py
get the data from sensor

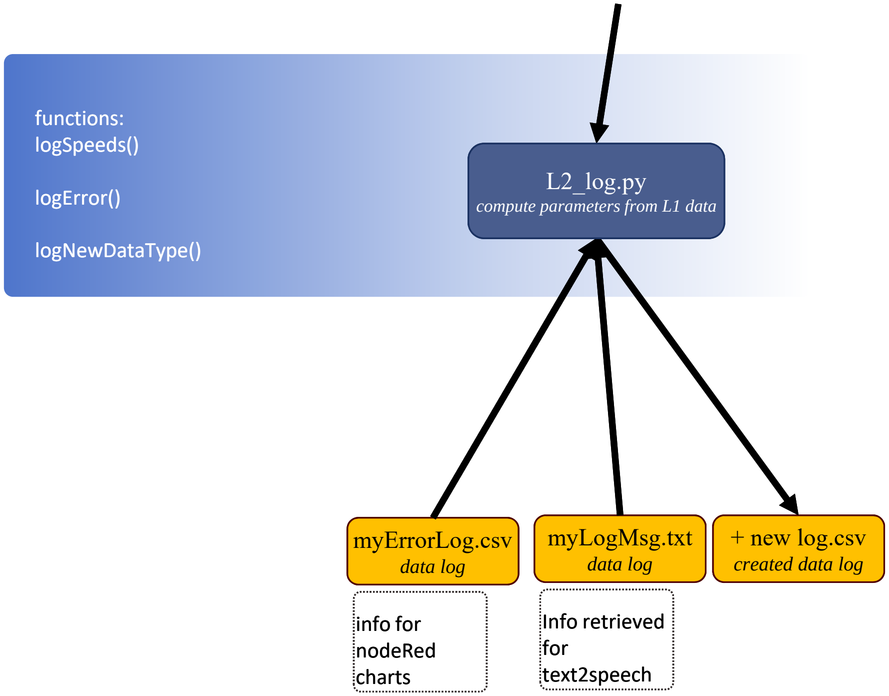
[raw sensor data]

Sensor
actual hardware

Software Architecture: Editing Log Files

Rather than interacting with hardware, the L2_log program will interact with files on the linux machine. It **acts as a sensor** in that it retrieves recorded data and it **acts as an actuator** in that it can receive data and perform an action with it (store in a file).

As of 2019.08, the L2_log.py program is not written but the intended functionality is shown. There will be ability to access different files and to write new files, by calling the functions from L2_log.py.



functions:
logSpeeds()
logError()
logNewDataType()
L2_log.py
compute parameters from L1 data
myErrorLog.csv
data log
myLogMsg.txt
data log
+ new log.csv
created data log
info for nodeRed charts
Info retrieved for text2speech