This slide deck was distributed in 2018-2020 for students in the course MXET300, Mobile Robotics at Texas A&M University. It's a template to begin your project planning as well as the final presentation.

[Key Value] Students reported this was most helpful for subdividing the robotic mission into routines and subroutines, where the student-built software will implement a discrete set of functions for each subroutine. Then, all cases for real-world scenarios are addressed with a robotic reaction, including errors in any moment during the final demonstration.

SCUTTLE Team Project:
Santa's Helper

Example Project by William Wonka and Dora Winnifred

Team 10

2020.09.18

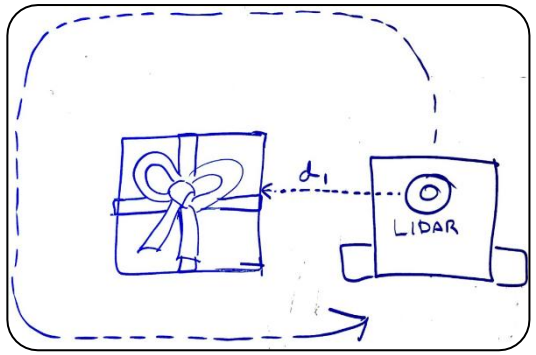
[Instructions] (delete this slide before submission)

- These are your instructions. The purpose of this task is to:
  - Produce slides for your final presentation
    - To be directly used. Just update if you have a change.
  - Break down your problem into subroutines
    - And assign yourself the first subroutines to accomplish.
  - Generate the important variables for your mission.
  - Identify all areas requiring development:
    - Sensing, Actuating, Computation
    - Identify log files you will create
  - Surface any gaps between current needs & current capability
    - Can I compute everything I need to compute?
    - Can my sensors produce all Information needed?
    - Is my hardware suitable to achieve actuation necessary?

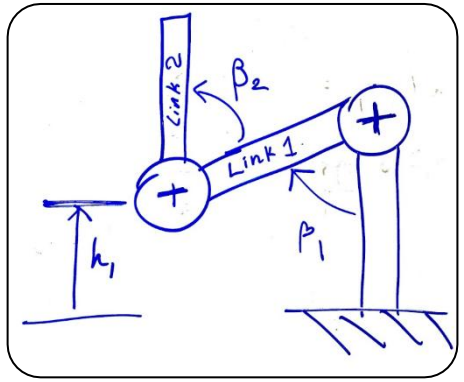
Mission: Wrap a Gift!

Wrap a gift: Encircle a gift by driving, and dispense a wrapper on the gift.

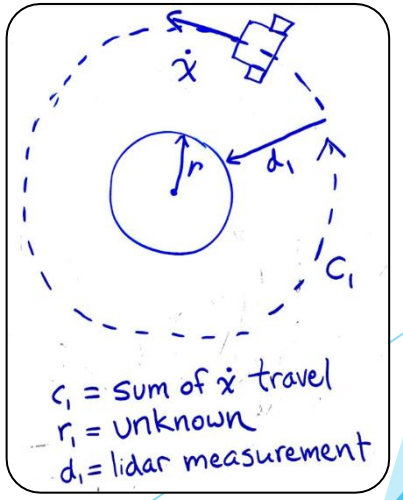
Drive around the gift



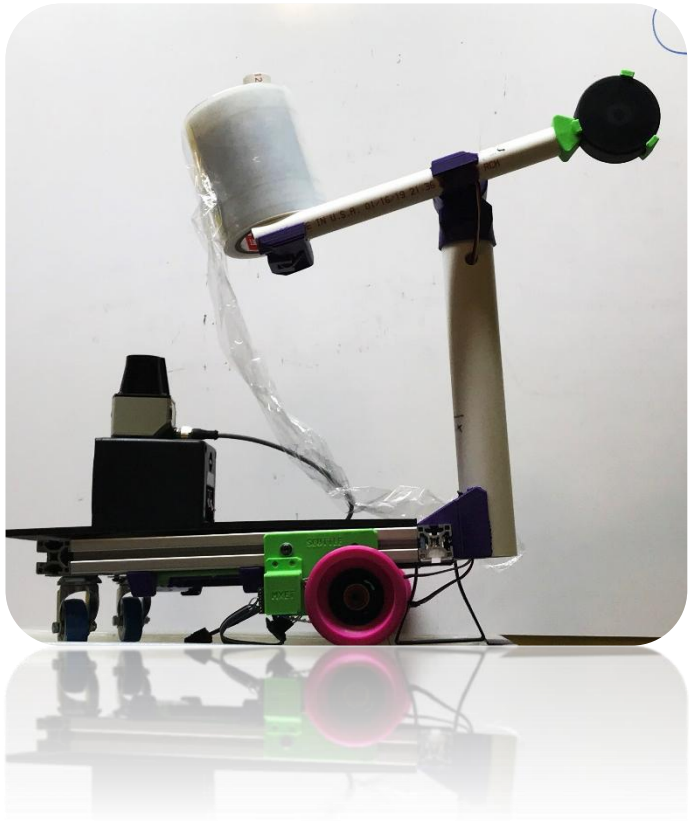
Lift the spool



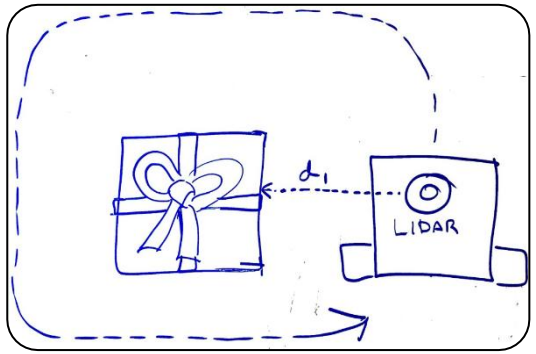
Calculate the Motion

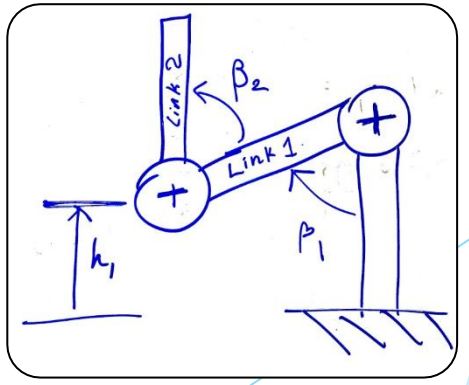


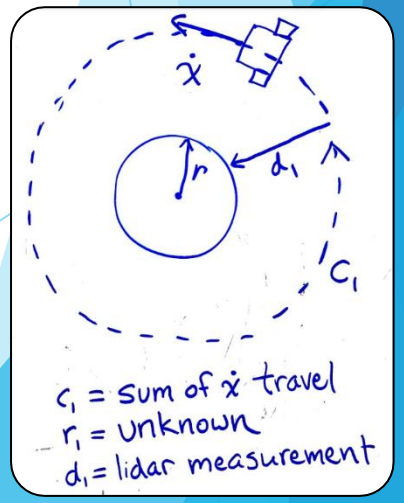
Mission:



- Scuttle will carry a spool of wrapping material for packaging items for shipment.
- The user will use the GamePad to drive the robot to the location of a package (on the right or left hand side, at a distance no more than 40cm
- The User pins the wrapping to the package, manually
- The User indicates to begin wrapping by a button on gamepad.
- SCUTTLE drives around the package to wrap it up, while raising the wrapper from minimum height to full height, at 10cm per rotation.







Routines Diagram

Routine 1

1.1
Receive the signal to start "wrapping" from the Gamepad.

Sensing: check gamepad buttons A & B
Actuating: none
Feedback to the user: condition of the buttons

button A pressed: - R2
button B pressed - R1

Routine 2

2.1
Circle around the package and maintain a constant distance.
Sensing: Lidar distance to package
Actuating: wheel speeds
Computation: proper turning rate to circle the object.
Feedback: too far, too close

2.2
Raise the wrapper spool at a rate of 10cm per revolution
Sensing: Check the compass for each increment of 90 degrees
Actuating: move the servo up by 2.5cm and adjust servo2 to stay vertical
Computation: calculate the
Feedback: spool height

2.3
Make a log file containing distance traveled intermittently in one column1 and the distance from SCUTTLE to the package in column2.
Sensing:
Actuating:
Computation: estimate radius of spool
Feedback:

Spool is lifted to final height h2: R3

Routine 3

3.1
Estimate the consumed material & output to the user

Sensing: Sum the logged distances from within a file.
Computation: Estimate the consumed material
Feedback

Estimation is completed - R1

Routine 4

4.1
Monitor the "flags.txt" file for changes. When the status of the flag changes, speak the associated phrase over the speaker.

Sensing: Collect the status indicators from .txt file
Feedback: Pass the phrases to the audio output

Estimation is completed - R1

Color Key

One routine is carried out together in a loop. This may be one L3 python file or one thread.

describes the goals of a subroutine

conditions which trigger the next Routine, & which routine is next

[Instructions:] Routines

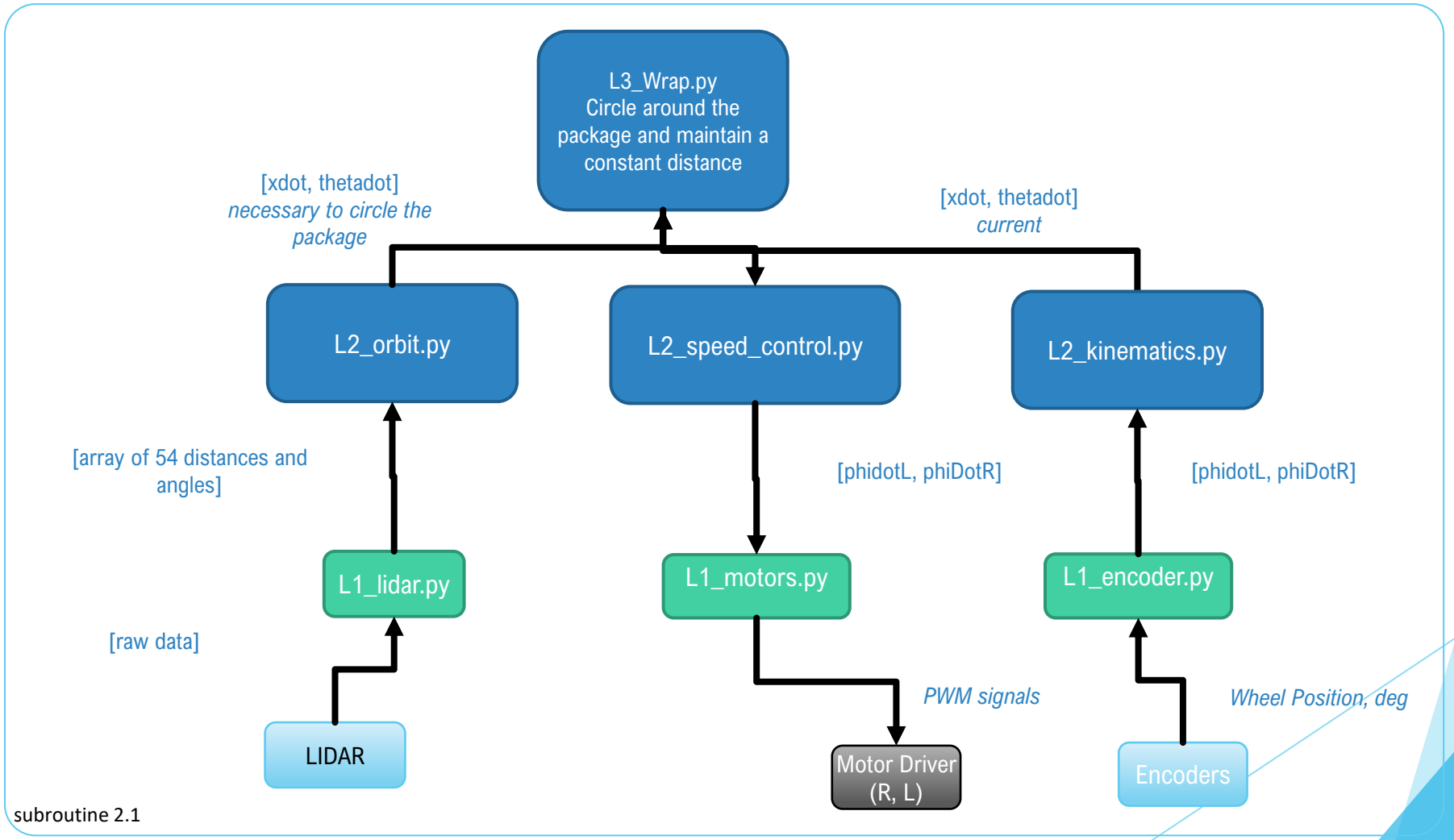
A subroutine should be something you can create and test independently. It may be an L2 code or a small loop in an L3 code which calls a couple of L2 codes.

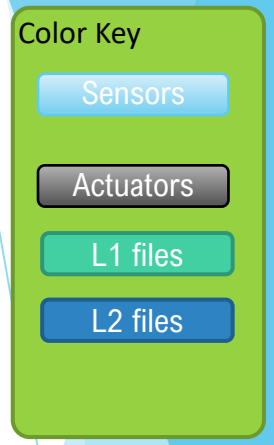
Try to simplify your subroutine into one of these:

- Sensing and interpretation
- *Calculating and Actuating*
- *Calculation and passing feedback to user*

conditions which trigger the next Routine should be easily converted into a single "true" or "false" statement

Data Flow (subroutine 2.1)





Routine 2.1 Details

Routine 2.1
Circle around the package and maintain a constant distance. Drive with constant x-dot and choose theta-dot to keep a distance from the object.

Color Key

Created By Team

Existing in SCUTTLE platform

Sensing Required:

Get distance d1, meters from lidar to the nearest object

Computation:

Compute target theta-dot by checking the angle of d1 vector, and attempt to maintain it along SCUTTLE'S y-axis

If d vector is forward of the robot's left, theta dot should decrease.
If d vector is too small, theta dot should decrease.

Outputs to the User:

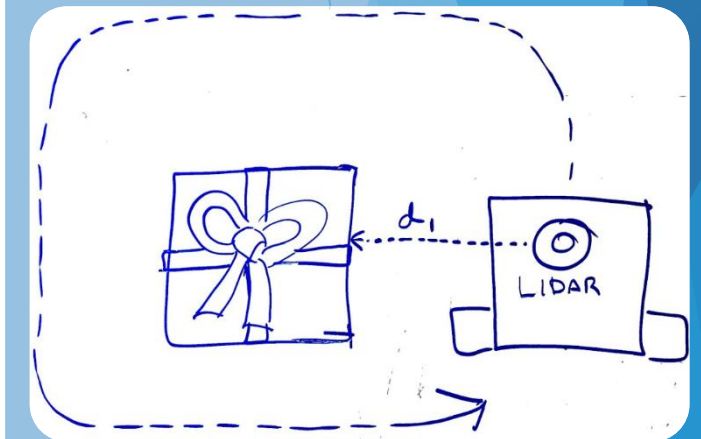
status of the d vector, current xdot, and current theta dot.

error 2.1, if d goes out of range

Actuation:

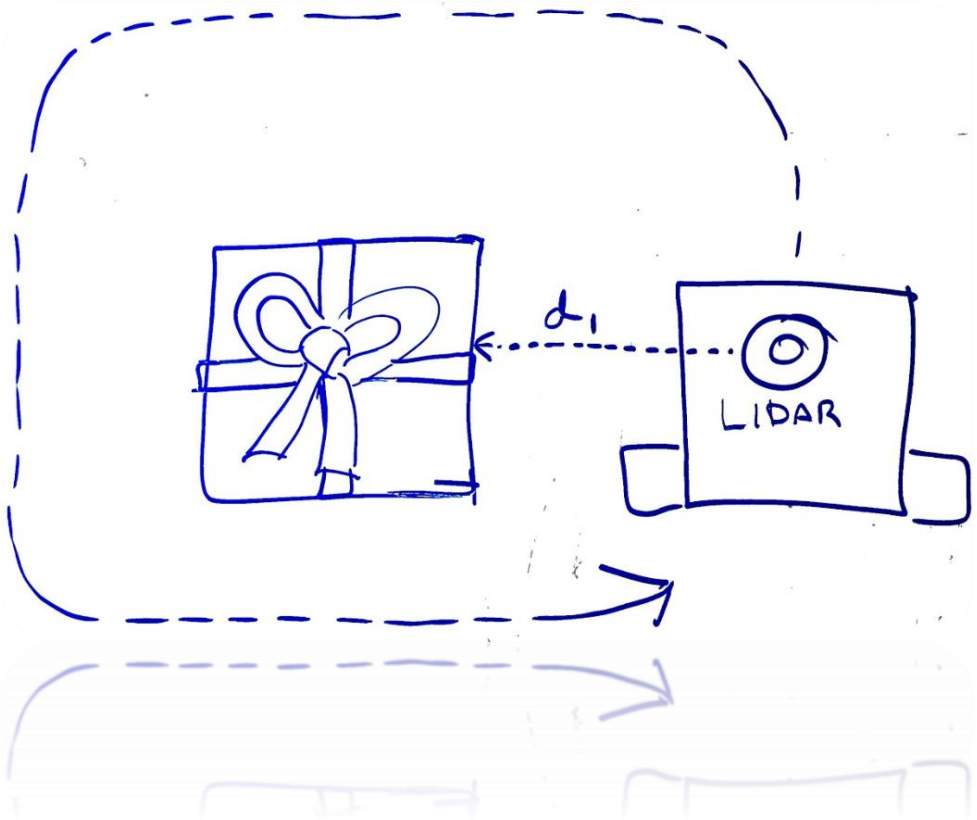
Drive the wheels with xdot & thetadot with closed-loop control

Figure for Computations



Routine 2.1 Demo Video

- In this video, the lidar collects an angle and a distance for the shortest obstacle and makes the d1 vector.



Routine 2.2 Details

Routine 2.2
Raise the wrapper spool at increments of 10cm per revolution. *The spool height will be h. When the compass increments more than 90 degrees, h will increment by driving the servo1 and servo2*

Color Key

Must Be Created By Team

Existing in SCUTTLE platform

Sensing:

L1_compass.py
Check when the compass movement

Computation:

L2_spoolSystem.py

Compute the angle beta1 of the servo1 to increase the height h by 2.5cm

Convert the angles to pass as L1_servo commands.

Computer the angle of servo 2 which will maintain a vertical position on the spool

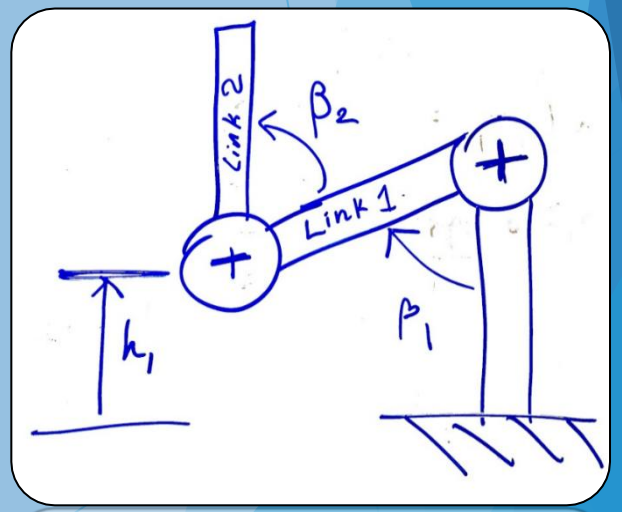
Outputs to the User:

L2_log.py
Log errors and status of the height increments

Actuation:

L1_servo.py
Passes servo commands to the servos

Figure for Computations



Routine 2.2 Demo Video

- In this video, the robot is turned (by hand) by 90 degrees and the servos controlling beta1 and beta2 are incremented by the proper amount to raise h by 2.5cm. When h reaches the top position, the cycle stops.