Administrivia

- MP1 is released yesterday. Due 03/05.
- Project Pitches are on due on 02/23. You will get 5 minutes to present and 2 minutes for questions from audience.
  - Software track
  - Hardware track
- Asynchronous students must send course staff a video the day before
  - Means before 11:59 PM CST
- https://publish.illinois.edu/safe-autonomy/projects-spring-2021/
- Signup sheet in Discord:
  https://docs.google.com/spreadsheets/d/1ExPJB_k32eS30z607XDBdSceGTWisBD0t7jZfhjLXMU/edit?usp=sharing
Basic Autonomous Vehicle Example

- **Actuators**
  - Motor
  - Servo (Steering)

- **Sensors**
  - Camera

- **On-Board Computer**
  - Raspberry Pi Zero
  - STM32F030 Microcontroller
Task: Detect Distracted Human and Stop
Solution A (The Trivial Solution)
But... Can Raspberry Pi Zero Run Realtime ML?

- Short answer: Not really
- Possible Solutions:
  - A Bigger car with a Larger GPU
  - A Remote Large GPU
Solution B (ROS)
What is ROS?

- Operating System for Robotics
  - Hardware Abstraction
  - Low-Level Device Control
  - Common Libraries/Packages
  - Communication between Processes

- Use Cases
  - Autonomous Vehicle Research (GEM Platform)
  - Autonomous Vehicle Simulator (MP)
  - Collaborative Robots (Robot Arms)
  - Research Robots (Turtlebot)

http://wiki.ros.org/kinetic
ROS Computation Graph

- Nodes: Processes (Python/C++)
- Master: Server
- Messages: Data Structures
- Topics: Message Buses
- Services: Request/Reply
- Bags: Datasets
- Publisher: Publish Message to Topic
- Subscriber: Subscribe Message from Topic

http://wiki.ros.org/ROS/Concepts
ROS Computation Graph: MP0 Example
ROS Nodes and Master

- **Node**: ROS Processes (Camera, Lidar, ML Algorithms)
- **Master**: Let the Nodes Know Each Other/ Keep Parameters
- **roscore**: Start ROS Master Server (Invoked during first roslaunch automatically)
- **rosnodename list**: List all ROS nodes
- **rosnodename info [node_name]**: Tells you more about specific node
- **rosrunch [ros_package] [node_name]**: Run a ROS node
- **python [node_name.py]**: Run a ROS node (More for Debugging)

http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes
Yet Another Example

- One of My Favourite Sensor: Intel Realsense RGB-D Camera

https://www.intelrealsense.com/depth-camera-d435/
Realsense Node

- As an RGB-D camera, it provides:
  - RGB Image
  - Depth Image
  - RGB Point Cloud
  - And a lot of other things...

- Part of results after running `rosnode info`:
  - Publications:
    - /camera/color/image_raw [sensor_msgs/Image]
    - /camera/depth/image_raw [sensor_msgs/Image]
    - /camera/depth/color/points [sensor_msgs/PointCloud2]
    - ...

ROS Topics

- Topics: Message Buses
- Publisher(s) stream message through topic to subscriber(s).
- `rostopic list [-v]`: List all topics (-v for more information)
- `rostopic info [topic_name]`: Print info about topic (message type)
- `rostopic echo [topic_name]`: Print messages to screen
- `rostopic echo [topic_name] -n 1`: Print 1 message to screen
- `rostopic hz [topic_name]`: Print publish frequency
- `rostopic bw [topic_name]`: Print topic bandwidth

http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics
ROS Messages

- Messages are data structures with typed fields defined by .msg files.
- To read/write message, simply use the following notation:
  - `msg1.fieldA = 1`
- The above works only on primitive types.
  - `bool`
  - `int8/uint8/int16/uint16/int32/uint32/int64/uint64`
  - `float32/float64`
  - `string`
  - “time/duration”

```c
# This message contains an uncompressed image
# (0, 0) is at top-left corner of image
#
Header header
  # Header timestamp should be acquisition time of image
  # Header frame id should be optical frame of camera
  # origin of frame should be optical center of camera
  # +x should point to the right in the image
  # +y should point down in the image
  # +z should point into to plane of the image
  # If the frame id here and the frame id of the CameraInfo
  # message associated with the image conflict
  # the behavior is undefined
uint32 height
  # image height, that is, number of rows
uint32 width
  # image width, that is, number of columns
string encoding
  # Encoding of pixels -- channel meaning, ordering, size
  # taken from the list of strings in include/sensor_msgs/image_encodings.h
int8 is bigendian
  # is this data bigendian?
uint32 step
  # Full row length in bytes
uint8[] data
  # actual matrix data, size is (step * rows)
```

Realsense... again

- Results after running `rostopic list`
  - `/camera/color/image_raw`
  - `/camera/depth/image_raw`
  - `/camera/depth/color/points`
  - ...

- `rostopic info /camera/color/image_raw`:
  - Type: `sensor_msgs/Image`
  - Publishers:
    - * `/camera/realsense2_camera_manager` (http://localhost:39161/)
  - Subscribers: None <--- Note this

- `rostopic echo`?
  - We need a way to visualize image.
Your Friend RQT

- ROS’s official 2D GUI
- **rqt**: Topic Monitor + Node Graph Visualizer + Transformation Tree+
- **rqt_image_view**: Specialized in displaying image messages
Publishers and Subscribers

- Now run `rostopic info /camera/color/image_raw`:
  - Type: sensor_msgs/Image
  - Publishers:
    - * /camera/realsense2_camera_manager (http://localhost:39161/)
  - Subscribers:
    - * /rqt_gui_cpp_node_25437 (http://localhost:44713/)  
- Publisher: Publishes messages to topic (source)
- Subscriber: Subscribes messages from topic (sink)
- Nodes can have multiple publishers and subscribers.
ROS Computation Graph Revisited

Master

Register

Publisher

Topic

Strong Typed Message

Register

Subscriber
ROS Services

- ROS Topic Model: Good for Many-to-Many One-Way Transport
- What if you want a request/reply interaction in a distributed system?
- ROS Service: One node requests, and another node replies
  - Spawning models
  - Setting parameters
  - ...

File: `dynamic_reconfigure/Config.msg`

Raw Message Definition

```plaintext
BoolParameter[] bools
IntParameter[] ints
StrParameter[] strs
DoubleParameter[] doubles
GroupState[] groups
```

Compact Message Definition

```plaintext
dynamic_reconfigure/BoolParameter[] bools
dynamic_reconfigure/IntParameter[] ints
dynamic_reconfigure/StrParameter[] strs
dynamic_reconfigure/DoubleParameter[] doubles
dynamic_reconfigure/GroupState[] groups
```


http://wiki.ros.org/Services
ROS Bags

- ROS Bags: Record Messages from Topics and Replay Later
- Like a video but with more information
- `rosbag record -a`: Record everything
- `rosbag info [*.bag]`: Summary of contents
- `rosbag play [*.bag]`: Play bag once
- `rosbag play -l [*.bag]`: Loop playback

http://wiki.ros.org/rosbag/Commandline
ROS Workflow in MP0

- **catkin_make**: Build ROS catkin workspace (similar to make)
- **source devel/setup.bash**: Execute a set of commands to setup the workspace (location of ROS packages, nodes, etc.)
- **roslaunch mp0 mp0.launch**: Launch set of nodes with parameters for running MP0
- **python main.py --d_sense 15 --v_0 5 --a_b 5 --t_react 0.00**: Launch main MP nodes
- **python set_pos.py --x 0 --y 0**: Launch a node that sets the position of the car
Example: ROS Publisher

```python
#!/usr/bin/env python
# license removed for brevity
import rospy
from std_msgs.msg import String

def talker():
    pub = rospy.Publisher('chatter', String, queue_size=10)
    rospy.init_node('talker', anonymous=True)
    rate = rospy.Rate(10)  # 10hz
    while not rospy.is_shutdown():
        hello_str = "hello world \%s \%s . %s" % (rospy.get_time(), hello_str)
        rospy.loginfo(hello_str)
        pub.publish(hello_str)
        rate.sleep()

if __name__ == '__main__':
    try:
        talker()
    except rospy.ROSInterruptException:
        pass
```

http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29
Example: ROS Subscriber

```python
#!/usr/bin/env python
import rospy
from std_msgs.msg import String

def callback(data):
    rospy.loginfo(rospy.get_caller_id() + " I heard %s", data.data)

def listener():
    # In ROS, nodes are uniquely named. If two nodes with the same
    # name are launched, the previous one is kicked off. The
    # anonymous=True flag means that rospy will choose a unique
    # name for our 'listener' node so that multiple listeners can
    # run simultaneously.
    rospy.init_node('listener', anonymous=True)
    rospy.Subscriber("chatter", String, callback)

    # spin() simply keeps python from exiting until this node is stopped
    rospy.spin()

if __name__ == '__main__':
    listener()
```

http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29
ROS Launch Files

- **roslaunch**: A Tool for Easily Launching Multiple ROS Nodes
  - Remember how nodes are separated and “kinda” independent?
- **What it also does**: Setting Global Parameters on Server
  - Robot Model
  - Robot Name
  - ...
- **Launch files**
  - XML Format
  - Can Find Packages and Pass Arguments
  - Mapping Topics
  - ...

http://wiki.ros.org/roslaunch/XML
## ROS Packages

- **Package:** Collection of Node Files, Launch Files, CMake List, Meta Information, and Other Things
- **Git Packages:** Put under src folder (MP packages, Lidar, GEM...)
- **apt-get Packages:** Gazebo, Controllers, Drivers...
  - Install by apt-get/apt package manager

---

http://wiki.ros.org/Packages

https://subscription.packtpub.com/book/hardware_and_creative/9781788478953/1/ch01lv1sec13/understanding-the-ros-filesystem-level
Transformation (TF2)

- In real life, cars and robots are not a single point.
- For example:
  - Where is the camera frame with respect to the robot base? (Fixed)
  - Where is the robot base with respect to the starting point? (Dynamic)
- TF2 Listener (“Subscriber”)
- TF2 (Static) Broadcaster (“Publisher”)
Realsense... The Third Time

- Realsense Tracking Camera

https://www.intelrealsense.com/tracking-camera-t265/
What does it do?

- As an tracking camera, it provides:
  - IMU Data
  - Odometry
  - Transformation (Static/Dynamic)
  - Optional Fisheye Image
  - And a lot of other things...

- Part of results after running `rosnode info`:
  - Publications:
    - `/camera/odom/sample [nav_msgs/Odometry]`
    - `/tf [tf2_msgs/TFMessage]`
    - `/tf_static [tf2_msgs/TFMessage]`
    - `...`
Your Other Friend RViz

- rqt: 2d visualizer
- RViz: 3d visualizer
- Provided in the MP (Launch file)
- Can also be launched using “rviz”
- Supports common ROS messages
- Especially useful for:
  - Robot Model
  - Transformation (TF)
  - Point Cloud
  - LaserScan (2D Lidar)
Gazebo

- Multi-Robot Simulator
- ROS Simulator
- Simulates:
  - Robot Motion (Physics)
  - Robot Model
  - Sensor (Camera/Lidar)
  - Custom Plugins
- MPs use Gazebo for simulating the vehicle and the environment.
How to make things run faster?

- Gazebo is demanding (like a game).
  - Physics Engine
  - Rendering
  - Sensor Simulation
- Algorithms
  - Code with efficiency in mind
  - Some algorithms just run slowly
- VMWare
  - Increase CPU count
  - Increase memory
  - Increase VGPU memory
End of ROS Intro: Q&A

- Can you use ROS1 on Windows/macOS/WSL?
  - Yes, but I wouldn’t recommend you do that.
- What about ROS2?
  - It will probably be better than ROS1 but lacks community support at current stage.
- My Gazebo crashes!
  - Restart/ Reboot
- How do I record video demo?
  - OBS Studio
Computer Vision

- OpenCV
  - MP1: Lane Detection
  - Pre-Processing
  - Camera Configurations
  - Post-Processing
  - Matching
- NumPy

https://opencv.org/
Machine Learning

- PyTorch
- TensorFlow
- NumPy
- We have 2x 2080Ti on GEM

https://github.com/MaybeShewill-CV/lanenet-lane-detection