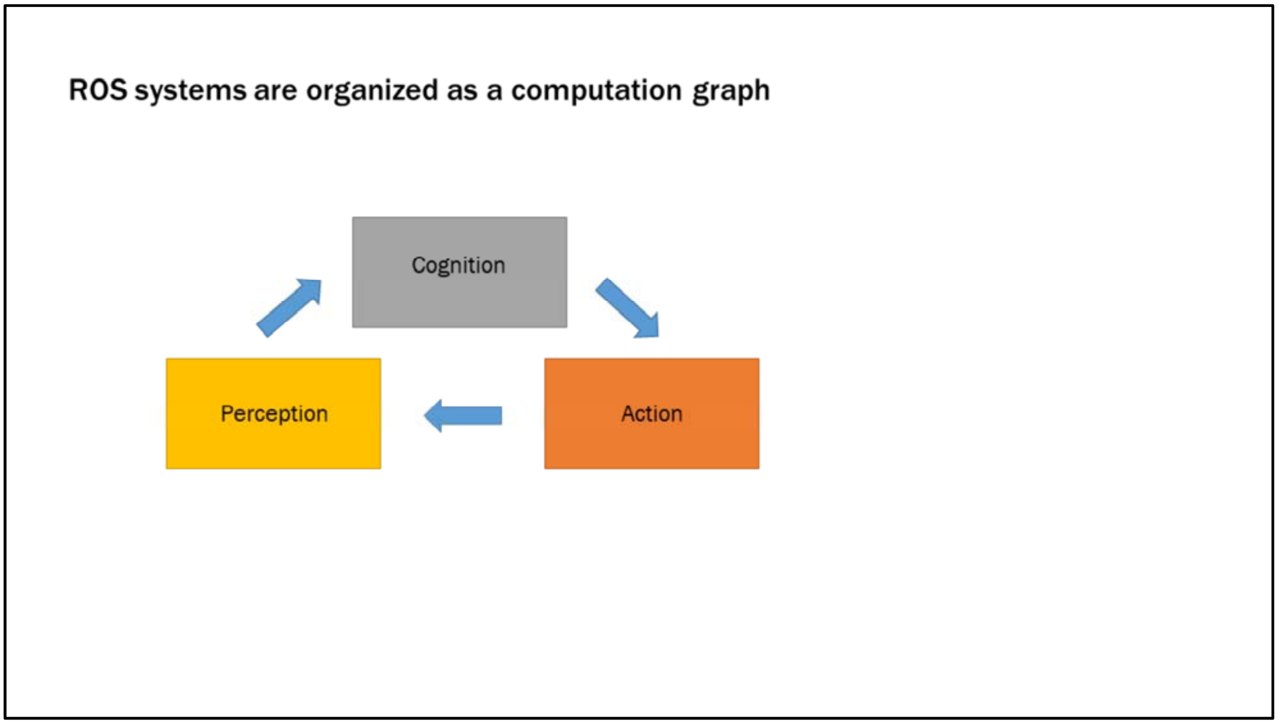
--
Introduction to ROS for Office Hour 1 in week 2
CS189, 2018
Julia Ebert and Florian Berlinger
--



ROS systems are organized as a computation graph

Cognition

Action

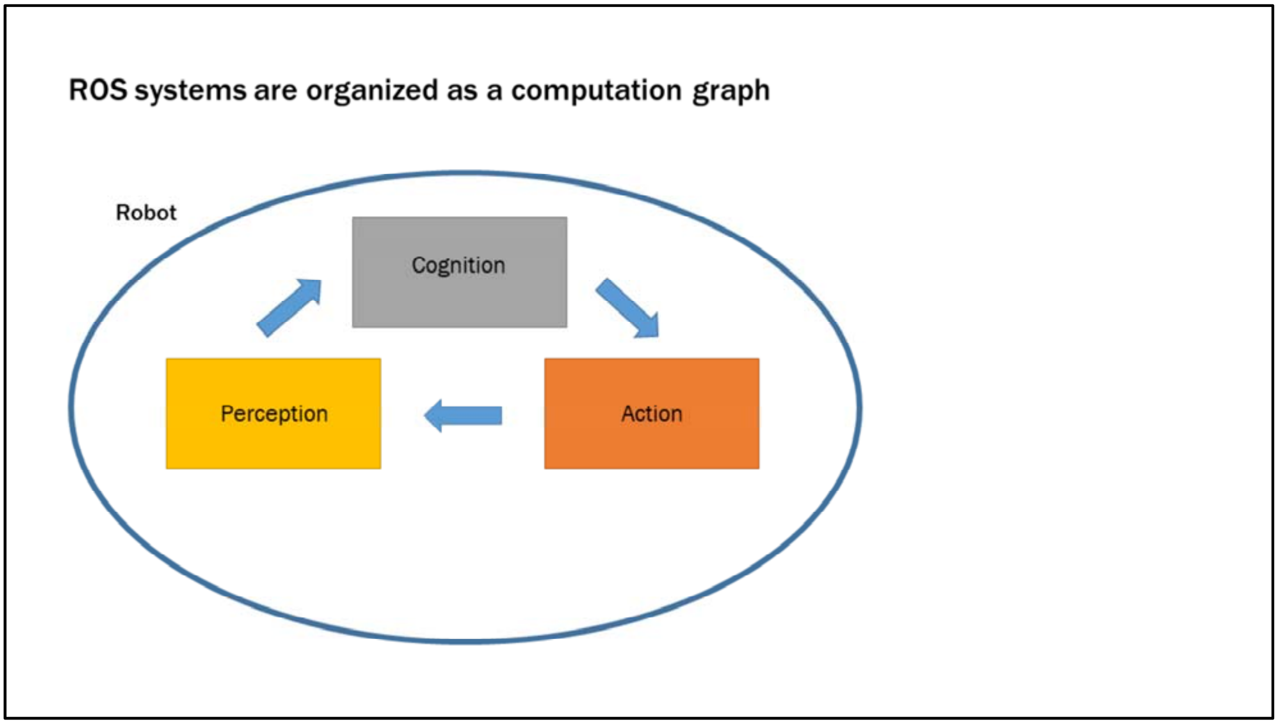
Perception

A number of independent programs each perform some piece of computation, passing data and results to other programs, arranged in a pre-defined network.

Task can be decomposed into many independent subsystems.

Each subsystem is called a node, which is usually a single-purpose program.

Event-driven system.



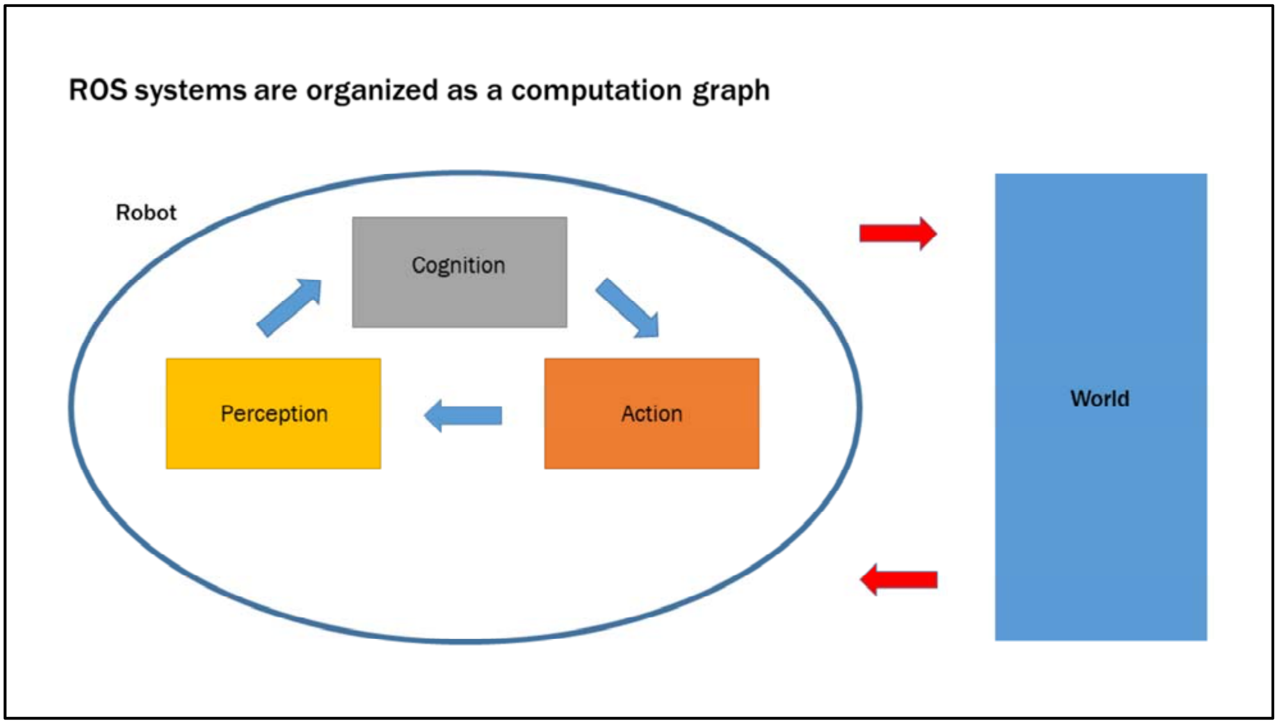
ROS systems are organized as a computation graph

A number of independent programs each perform some piece of computation, passing data and results to other programs, arranged in a pre-defined network.

Task can be decomposed into many independent subsystems.

Each subsystem is called a node, which is usually a single-purpose program.

Event-driven system.

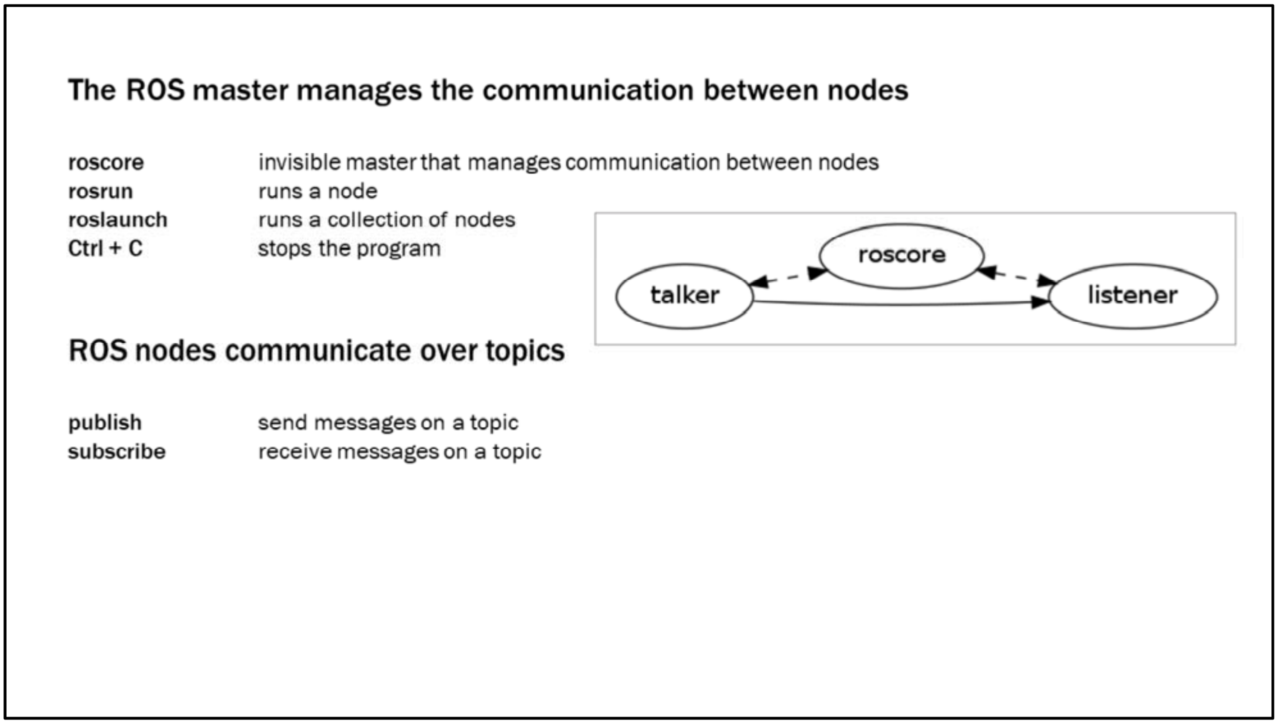


A number of independent programs each perform some piece of computation, passing data and results to other programs, arranged in a pre-defined network.

Task can be decomposed into many independent subsystems.

Each subsystem is called a node, which is usually a single-purpose program.

Event-driven system.



The ROS master manages the communication between nodes

roscore	invisible master that manages communication between nodes
rosrun	runs a node
roslaunch	runs a collection of nodes
Ctrl + C	stops the program

roscore
talker
listener

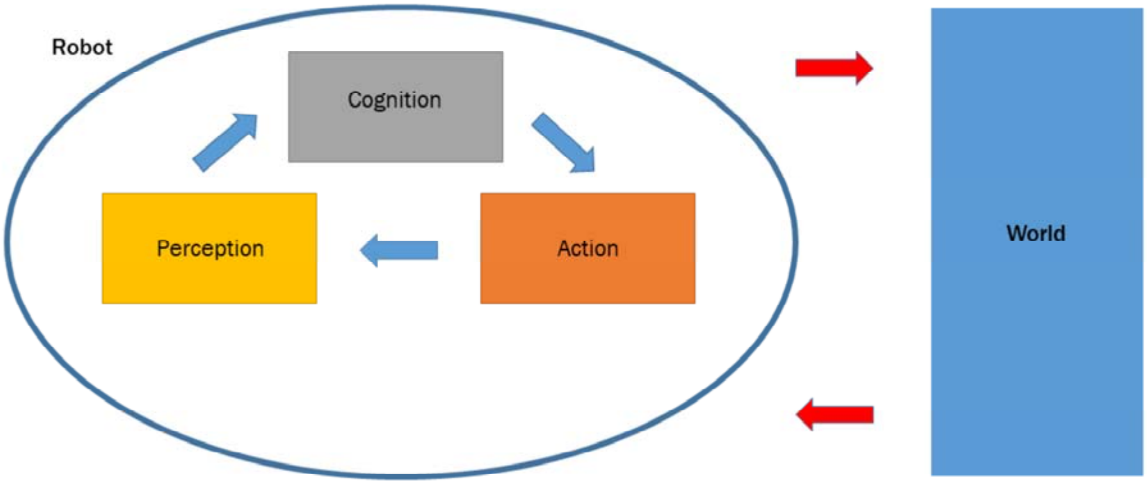
ROS nodes communicate over topics

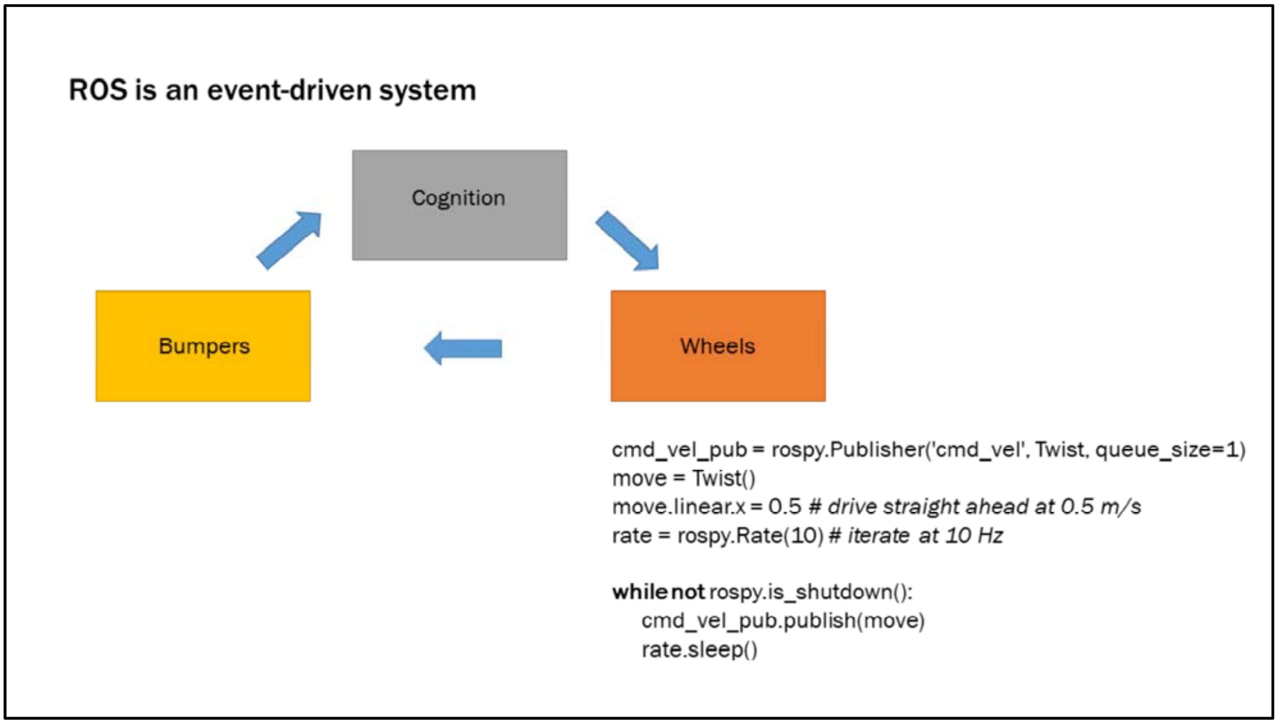
publish	send messages on a topic
subscribe	receive messages on a topic

Topic = stream of messages

Publish or subscribe to a topic (typically 1 publisher, n subscribers)

ROS systems are organized as a computation graph





ROS is an event-driven system

Cognition

Bumpers

Wheels

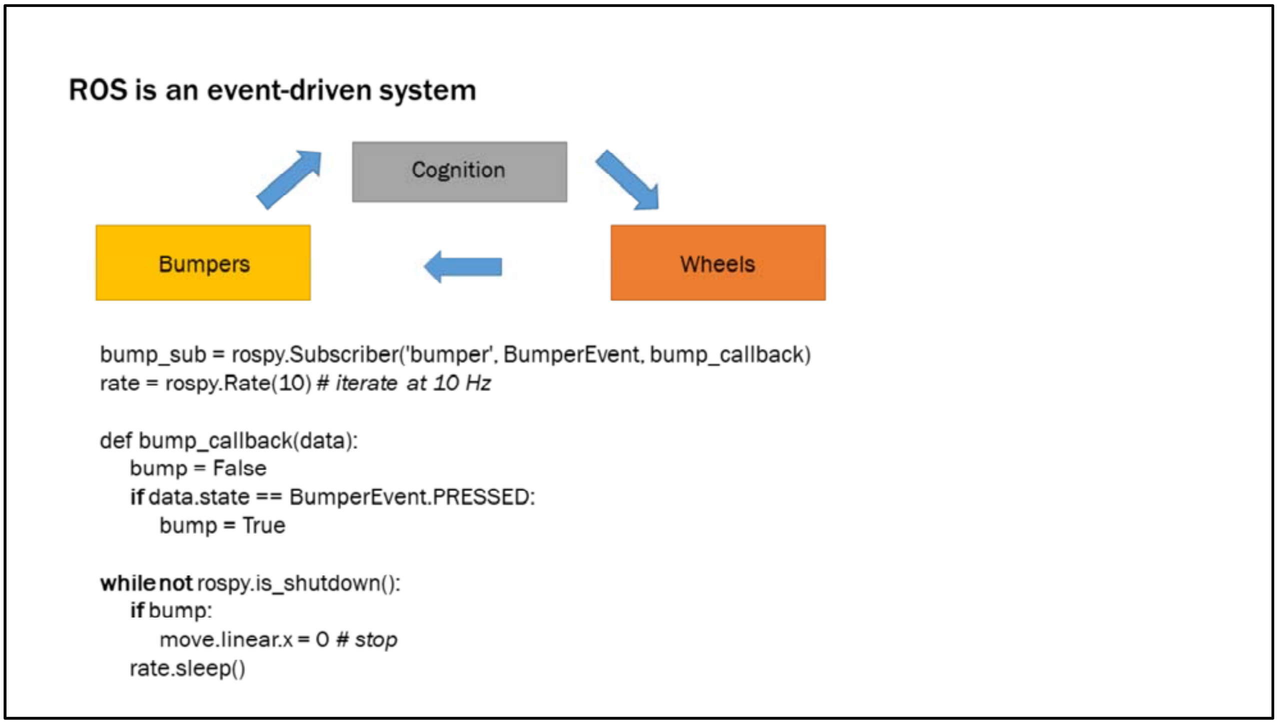
```
cmd_vel_pub = rospy.Publisher('cmd_vel', Twist, queue_size=1)
move = Twist()
move.linear.x = 0.5 # drive straight ahead at 0.5 m/s
rate = rospy.Rate(10) # iterate at 10 Hz

while not rospy.is_shutdown():
    cmd_vel_pub.publish(move)
    rate.sleep()
```

Wheels for action.

Bumper sensors for perception.

Physical world imposes events on the robot’s sensors, e.g., robot might bump into a wall.



ROS is an event-driven system

Cognition

Bumpers

Wheels

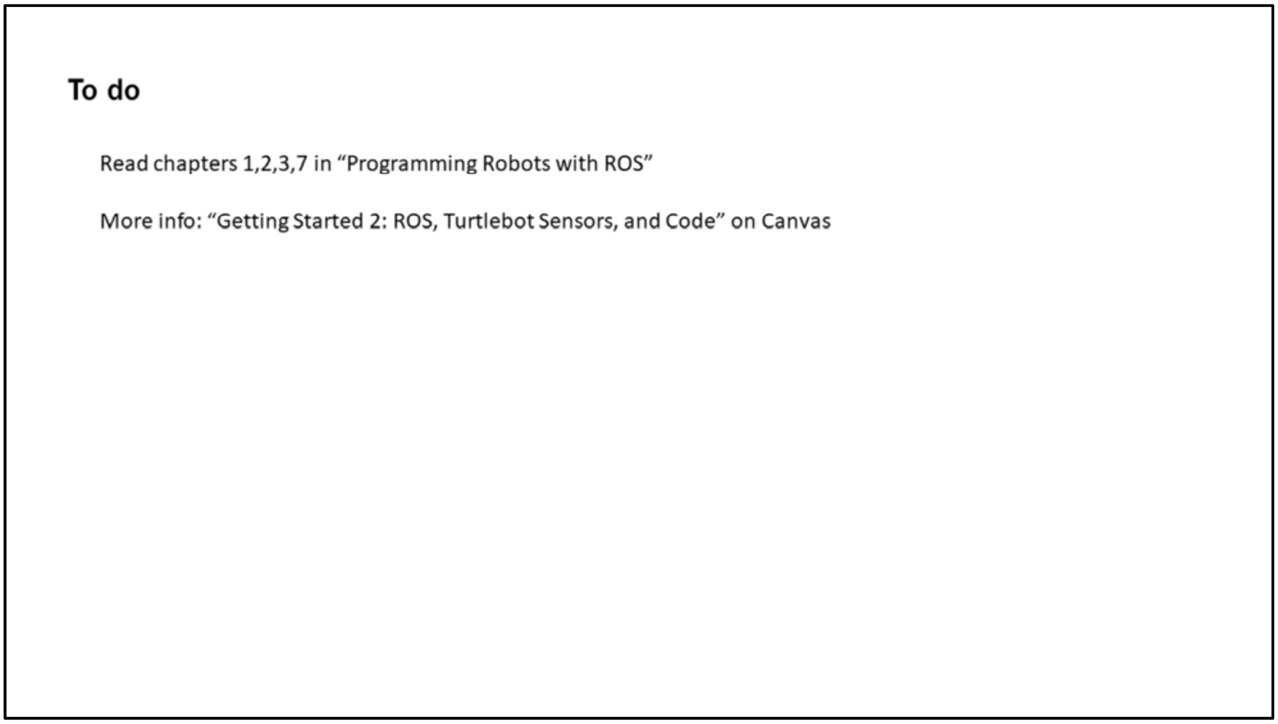
```
bump_sub = rospy.Subscriber('bumper', BumperEvent, bump_callback)
rate = rospy.Rate(10) # iterate at 10 Hz

def bump_callback(data):
    bump = False
    if data.state == BumperEvent.PRESSED:
        bump = True

while not rospy.is_shutdown():
    if bump:
        move.linear.x = 0 # stop
    rate.sleep()
```

Stop at bumper event.

Event-driven! No need to call bump_sub in each iteration of the while loop like it was the case for cmd_vel_pub.



To do

Read chapters 1,2,3,7 in “Programming Robots with ROS”

More info: “Getting Started 2: ROS, Turtlebot Sensors, and Code” on Canvas