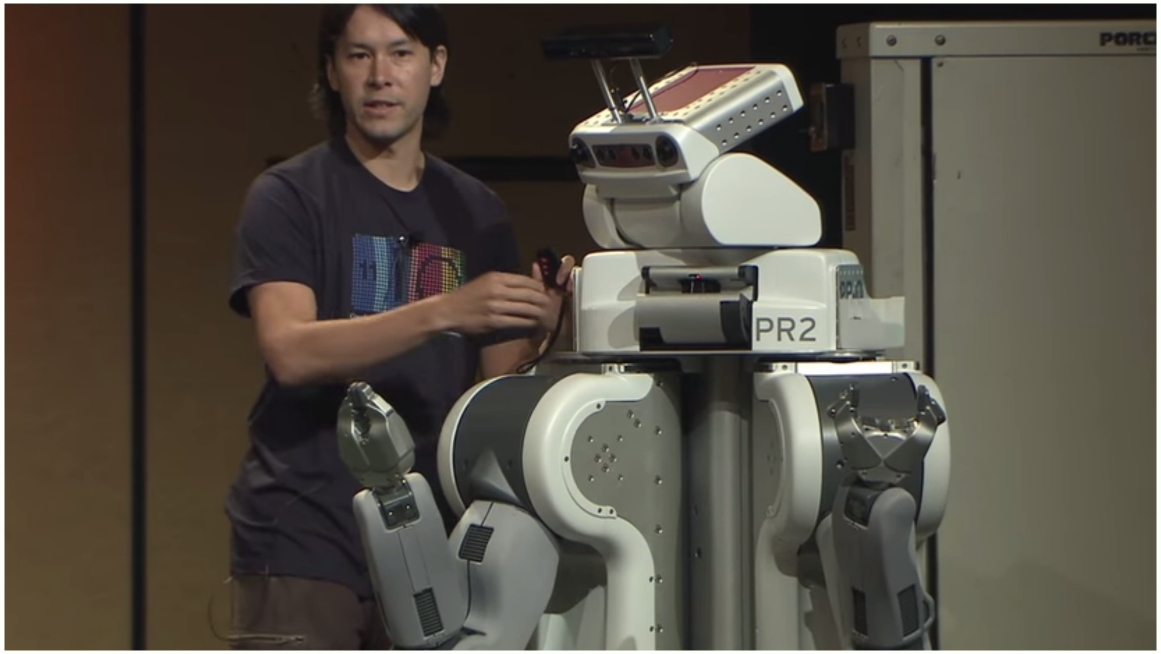
Rapid prototyping with rosh

Dan Lazewatsky (dlaz)
Personal Robotics Lab
School of Mechanical, Industrial
and Manufacturing Engineering
Oregon State University

rosh history

- Originally written by Ken Conley in 2011
- Demoed at Google I/O



- Mostly forgotten since then

What is rosh?

- ros h?
- ro sh?
- rosh...ambo?
- python scripting environment for ROS
    - interactive shell
    - scripts

What does rosh do?

- Eliminates boilerplate code
- Eases introspection
  - topics, services, nodes, actions, transforms, …
- Enables interaction with entire installed ROS ecosystem
  - packages, messages, bags, …

What is rosh…

good for

- interactive debugging
- short, linear scripts
- glue

not good for

- long scripts
- high performance
- multi-threaded code

rosh basics

Looping forever:

```
while ok():
  …
```

Getting info:

```
info(<almost anything>)
```

Visualization:

```
show(<almost anything>)
```

Useful aliases:

rosh	**rospy**
`now()`	`rospy.Time.now()`
`Time, Duration, Sleep`	`rospy.<Time, Duration, Sleep>`
`Header`	`rospy.Header`

Ways to use rosh

- interactive shell
  - tab completion of
    - topics, services, message names, lots more
- script
  - rosh is on your path!
    - `#!/usr/bin/env rosh`

Oregon State UNIVERSITY

rosh basics: packages

Accessing package info

```
packages.<pkg_name>
```

Available information

- name: package name
- path: full path to package
- depends1: direct package dependencies (build_depend+run_depend)
- launches: launch files in package
- manifest: manifest.xml or package.xml
- msg: messages defined in package
- srv: services defined in package
- nodes: nodes defined in package

Example

```
In [1]: packages.tf.depends1.angles.path
Out[1]: u'/opt/ros/hydro/share/angles'
```

rosh basics: messages

Accessing message info

```
msg.<pkg_name>
```

Equivalent to `packages.<pkg_name>.msg`

Instantiating messages

```
msg.std_msgs.ColorRGBA()
```

With positional arguments

```
msg.std_msgs.ColorRGBA(195,69,0,0)
```

With keyword arguments

```
msg.std_msgs.ColorRGBA(r=195,g=69)
```

Getting Message Definition

```
 [1]: show msg.std_msgs.ColorRGBA
-------> show(msg.std_msgs.ColorRGBA)
float32 r
float32 g
float32 b
float32 a
```

rosh basics: topics (basics)

Subscribing

Get the last message:

```
    topics.topic_name[0]
```

Get the next message:

```
    topics.topic_name[1]
```

Get all future messages:

```
    for msg in topics.topic_name[:]:
        …
```

Publishing

Publish a message object:

```
    topics.topic_name(msg)
```

Publish and create a new object (e.g. ColorRGBA):

```
    topics.topic_name(r=195, g=69, b=0)
```

rosh basics: topics (advanced)

Get all future messages:

```
for msg in topics.topic_name[:]:
    …
```

Get M through Nth messages on topic:

```
for msg in topics.topic_name[M:N]:
    …
```

Pipe one topic to another (topic_tools/mux)

```
topics.topic_name = topics.other_topic_name
```

Publish on a new topic

```
rostype(topics.new_topic, msg.std_msgs.ColorRGBA)
```

rosh basics: topic introspection

Get message definition for topic

Get nodes publishing topic

Get nodes subscribing to topic

```
topics.color
Out[0]:
float32 r
float32 g
float32 b
float32 a
```

rosh basics: topic introspection

Get message definition for topic	`ni = info(topics.color)`
Get nodes publishing topic	`ni.pub_nodes()`
Get nodes subscribing to topic	`Out[1]: /color_pub_node`

rosh basics: topic introspection

Get message definition for topic

Get nodes publishing topic

Get nodes subscribing to topic

```
ni = info(topics.color)
ni.sub_nodes()
Out[2]:
/rostopic_13197_1408484321
 530
```

rosh basics: services

Calling services

Call a service with a request object

```
resp = services.a_srv(req)
```

Call a service while creating a new request

```
resp = services.rosout.set_logger_level('ros',
 'warn')
```

rosh basics: topics + services example

Turn a service into a topic

rosh

```
rostype(topics.set_camera_info, msg.sensor_msgs.CameraInfo)
for info_msg in topics.set_camera_info[:]:
  services.camera_driver.set_camera_info(info_msg)
```

rospy

```
import rospy
from sensor_msgs.msg import CameraInfo
from sensor_msgs.srv import SetCameraInfo
def info_cb(msg, info_proxy):
    info_proxy[0](msg)
rospy.init_node('topicify_camera_info')
info_proxy = rospy.ServiceProxy('set_camera_info',
 SetCameraInfo)
rospy.Subscriber('set_camera_info', CameraInfo, info_cb,
 callback_args=(info_proxy,))
```

rosh basics: parameters

Retrieve a parameter

```
param_value = parameters.foo()
```

Set a parameter

```
parameters.foo = 'bar'
```

Set a bunch of parameters in a namespace

```
parameters.foo = dict(bar='baz', qux='asdf')
$ rosparam list /foo
/foo/bar
/foo/qux
```

Load parameters from yaml file

```
params = rosparam('params.yaml')
```

Set parameters from yaml file

```
parameters.foo = params['foo']
```

```
# params.yaml
foo:
    bar: baz
    qux: asdf
```

Oregon State UNIVERSITY

rosh plugins

Plugins provide additional functionality

Available plugins:

plugin	**provides**
rosh_common	actions, cameras
rosh_geometry	transforms, geometry helpers
rosh_visualization	show(cameras.<camera>)
rosh_robot	meta-plugin (loads rosh_common, rosh_geometry)
rosh_desktop	meta-plugin (loads rosh_visualization, rosh_common, rosh_geometry)

Loading plugins

In code

```
    load('foo_plugin', globals())
```

From the command line

```
    $ rosh --plugins=foo_plugin,bar_plugin
```

At startup

```
    #~/.ros/rosh/roshrc.py
    plugins['foo_plugin', 'bar_plugin']
```

Oregon State UNIVERSITY

rosh basics: geometry

Lookup transform

```
    xform = transforms.<src_frame>('<target_frame>')
```

Also provides

- Point
- Quaternion
- PointStamped
- PoseStamped
- QuaternionStamped
- Vector3Stamped

more features

Topic tools

  mux, relay, throttle

Bags

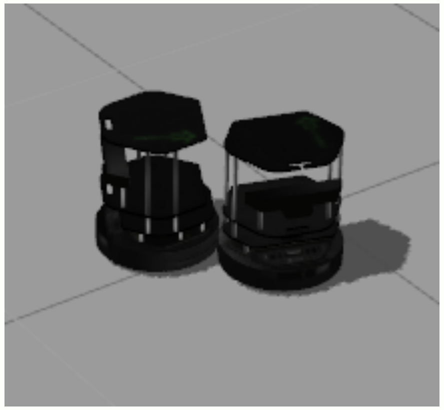
```
with Bag('test.bag') as bag:
  for topic, msg, t in bag.read_messages(topics=['foo']):
      print msg
```

Bagys (like bags, stored as yaml)

Putting it all together

Two robots enter, one leaves…

- **random_move.py**: moves one robot about randomly
- **follow.py**: tries to crash another robot into first robot
- **reset.py**: teleports robots to random locations on crash



Code available at https://github.com/dlaz/rosh_turtlebot_demo http://wiki.ros.org/rosh

https://github.com/OSUrobotics/rosh_robot_core

https://github.com/OSUrobotics/rosh_robot_plugins

https://github.com/OSUrobotics/rosh_desktop_plugins

Oregon State UNIVERSITY