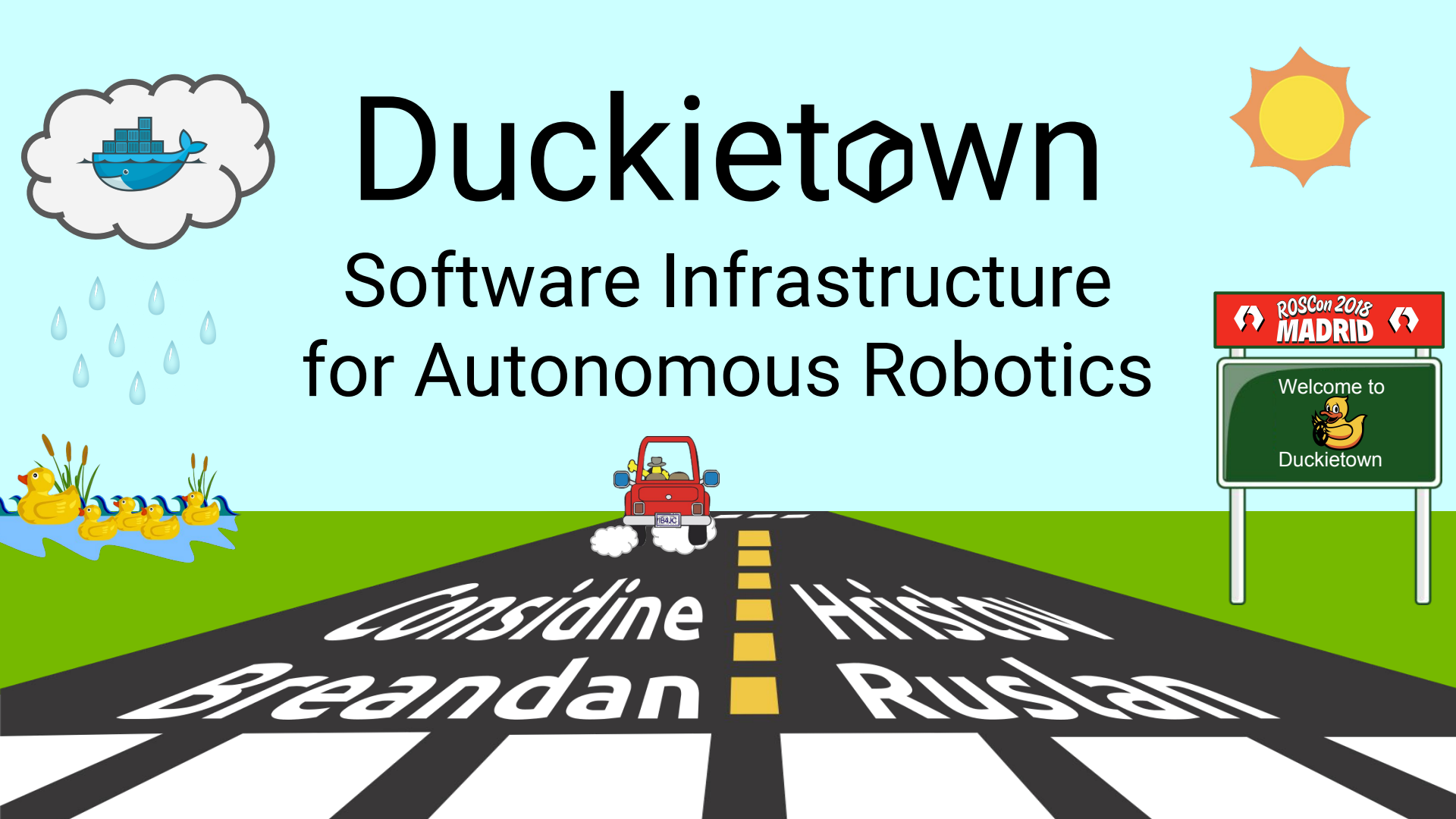
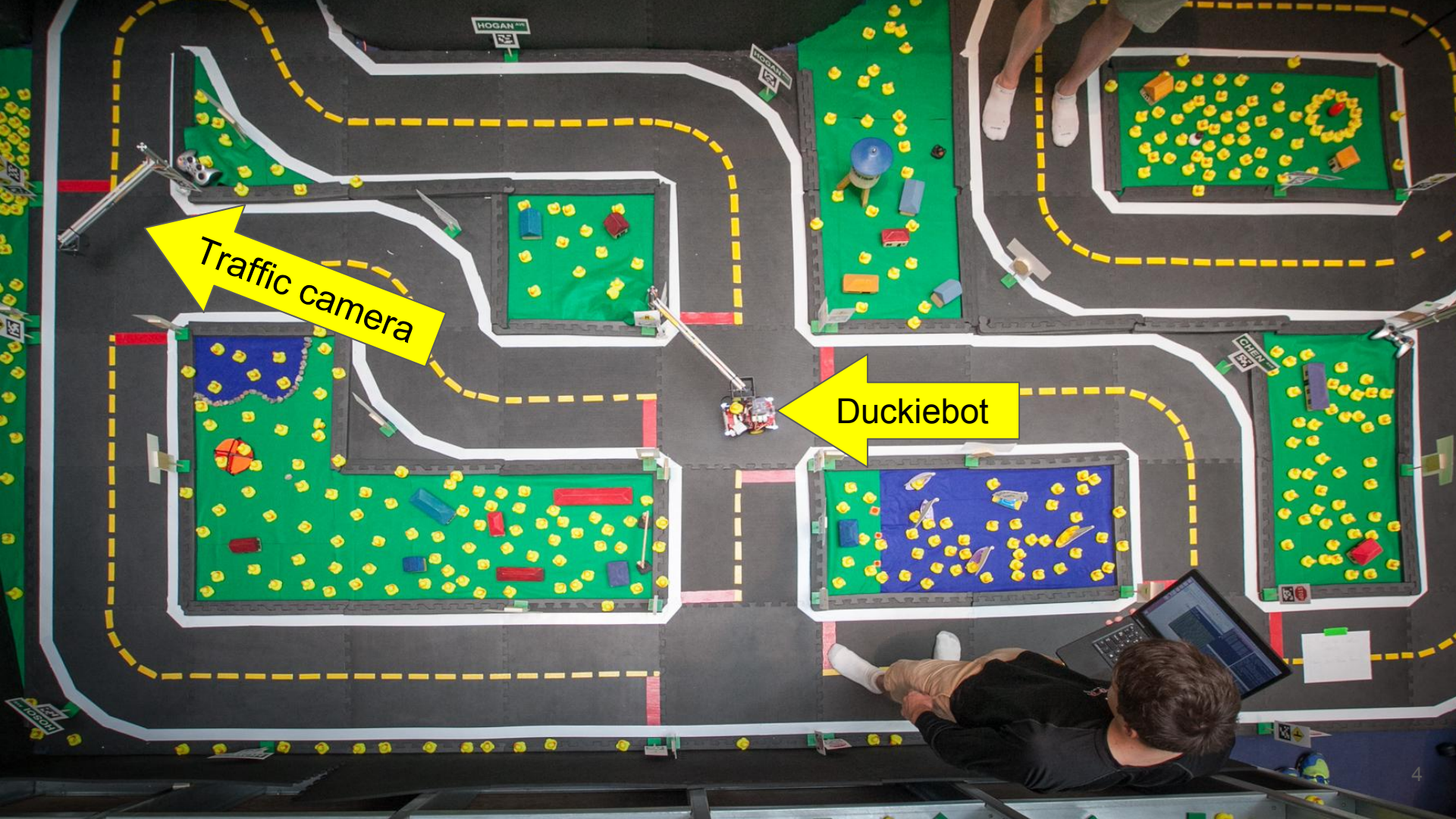
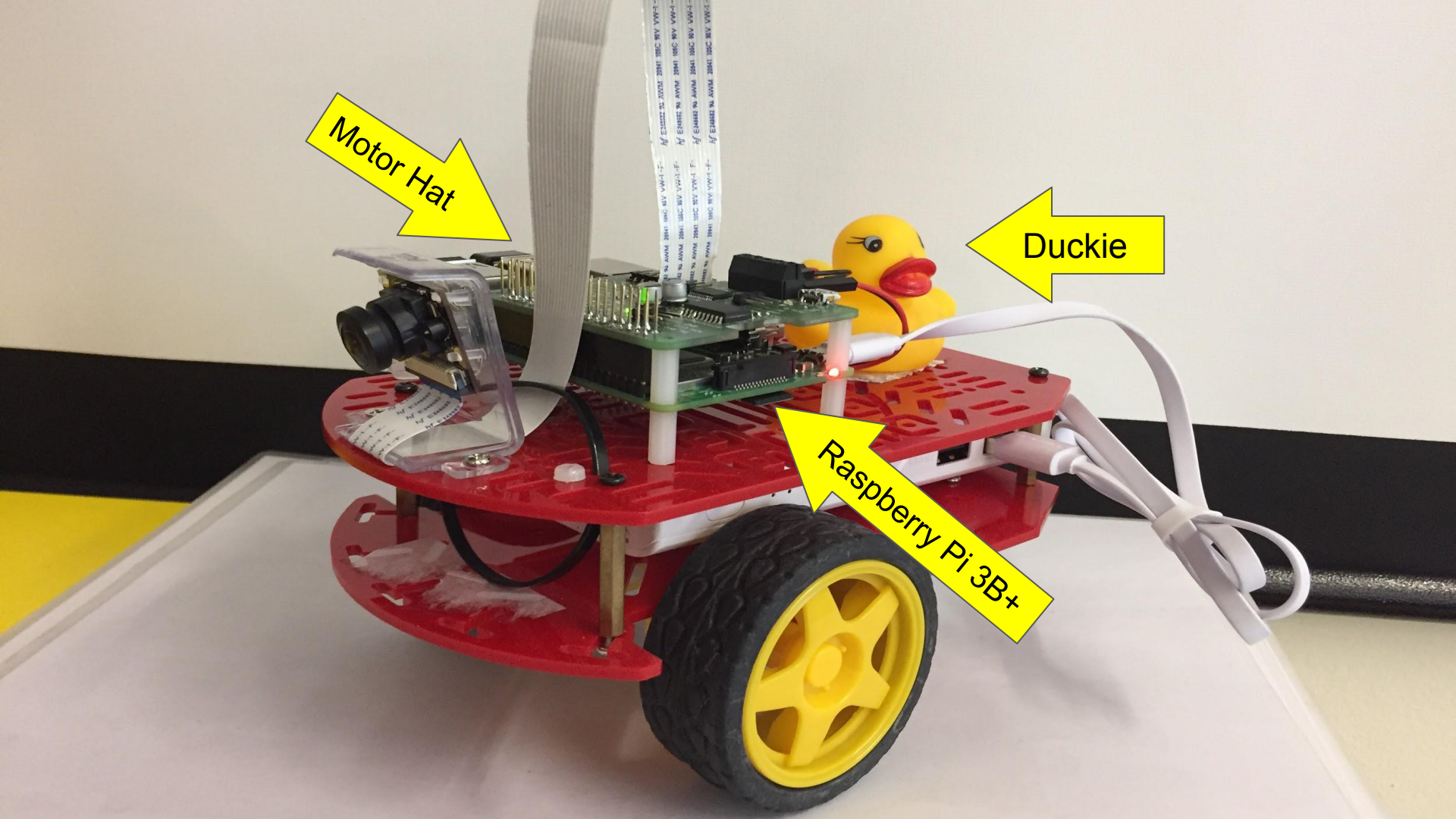

Duckietown
Software Infrastructure
for Autonomous Robotics
ROSCon 2018
MADRID
Welcome to
Duckietown
Considine
Hinsen
Breandan
Ruslan

Hello Duckietown!

- Low-cost platform for autonomy education and research
- Robotics and machine learning for students of all ages
- Teaches concepts in perception, planning, and control with miniature autonomous vehicles in the classroom

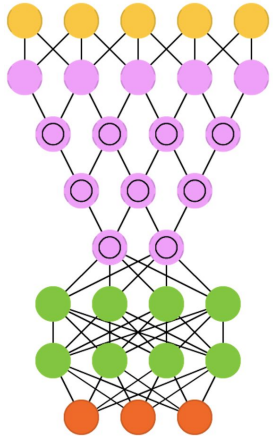

Watchtower
April Tag

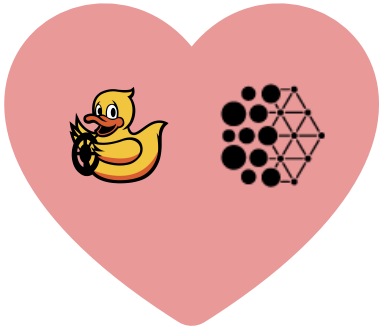

Traffic camera
Duckiebot


Motor Hat
Duckie
Raspberry Pi 3B+

Duckiebots and Deep Learning

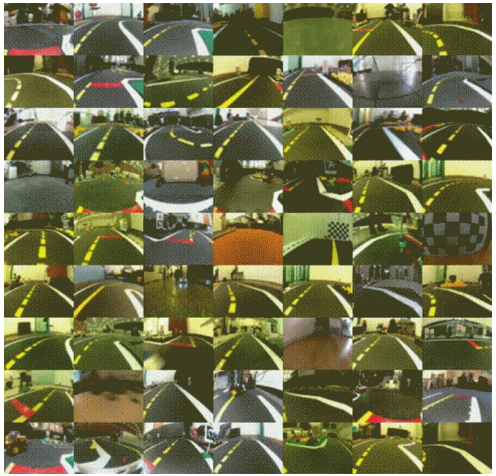
- Duckiebots are battery-powered vehicles
  - Camera is the only sensor
  - Classic computer vision pipeline
    - Multiple stages of hand-designed image processing
    - Proportional integral derivative (PID) controller
    - Each robot needs to be individually calibrated
- Mila is a deep learning research lab
  - Students interested in applying DL & RL
  - Teach students to teach robots how to drive

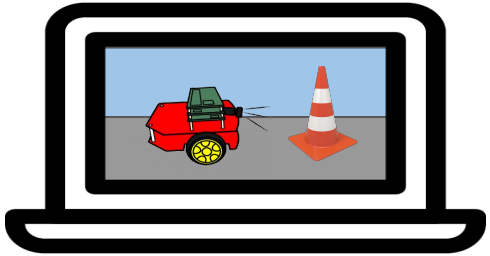




A Tale of Two Duckietowns

- Gathering real data is real slow
  - Time consuming and tedious to collect
    - Real robots break down, drain batteries
  - Even worse if you need a diverse dataset
    - Need to vary lighting conditions, etc.
- Simulation is an appealing alternative...
  - Can produce arbitrary amounts of data
  - Easy to augment with generated data
  - Simplifies reproducibility and verifiability

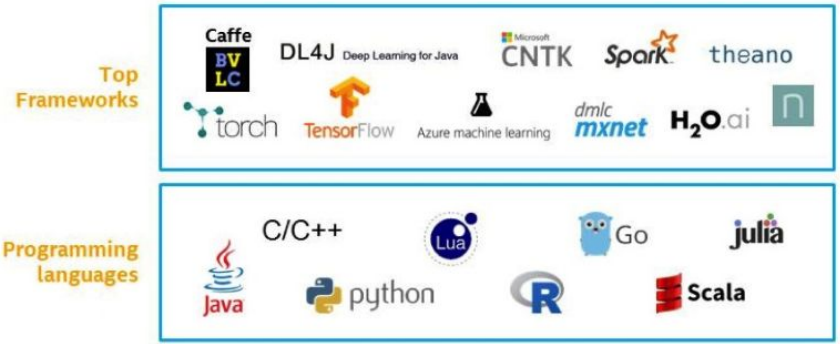


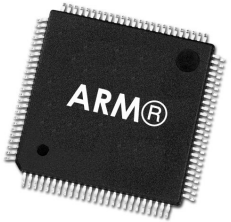


Machine learning is a changin’

- Evolving tools, frameworks, and languages
- Evolving domain models and architectures
- Evolving hardware technology and platforms

Cloud TPU

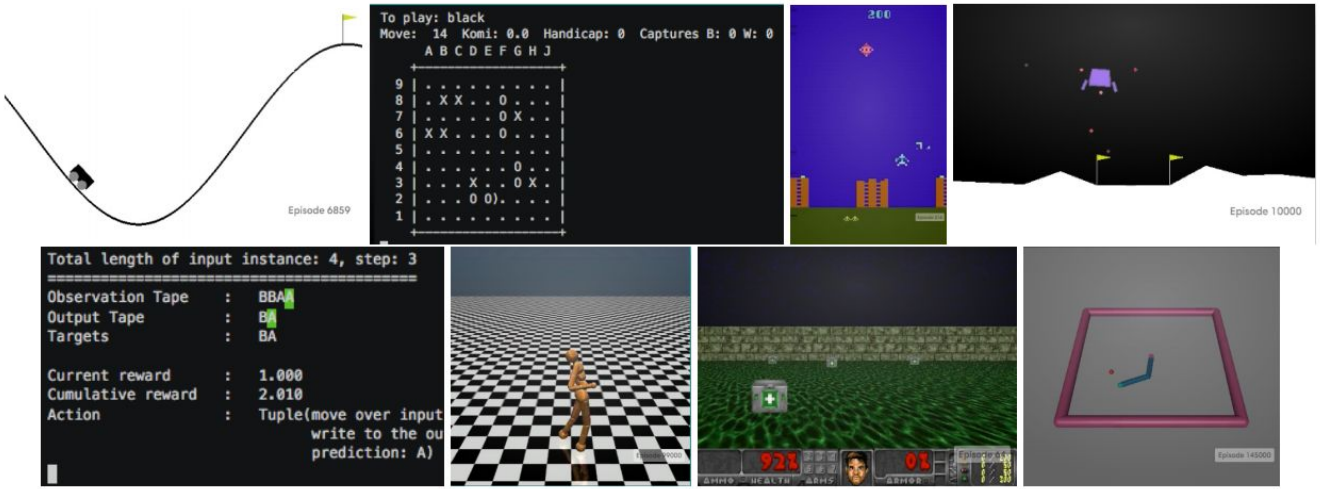




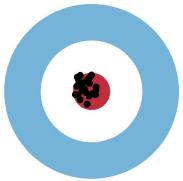
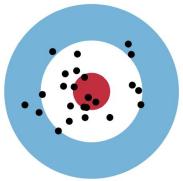
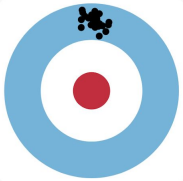
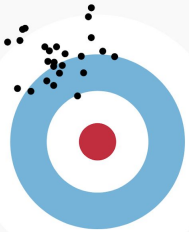
Simulators and environments

Brockman, Greg, et al. **"Openai gym."** arXiv:1606.01540 (2016).

https://arxiv.org/pdf/1606.01540.pdf



Problem: Bias-Variance and Overfitting

	Low Variance (High Precision)	**High Variance** (Low Precision)
Low Bias	 *Most Accurate*	
High Bias		 *Least Accurate*

Software is a changin’

- Karpathy, Andrej. **"Software 2.0."** (2017).

https://medium.com/@karpathy/software-2-0-a64152b37c35

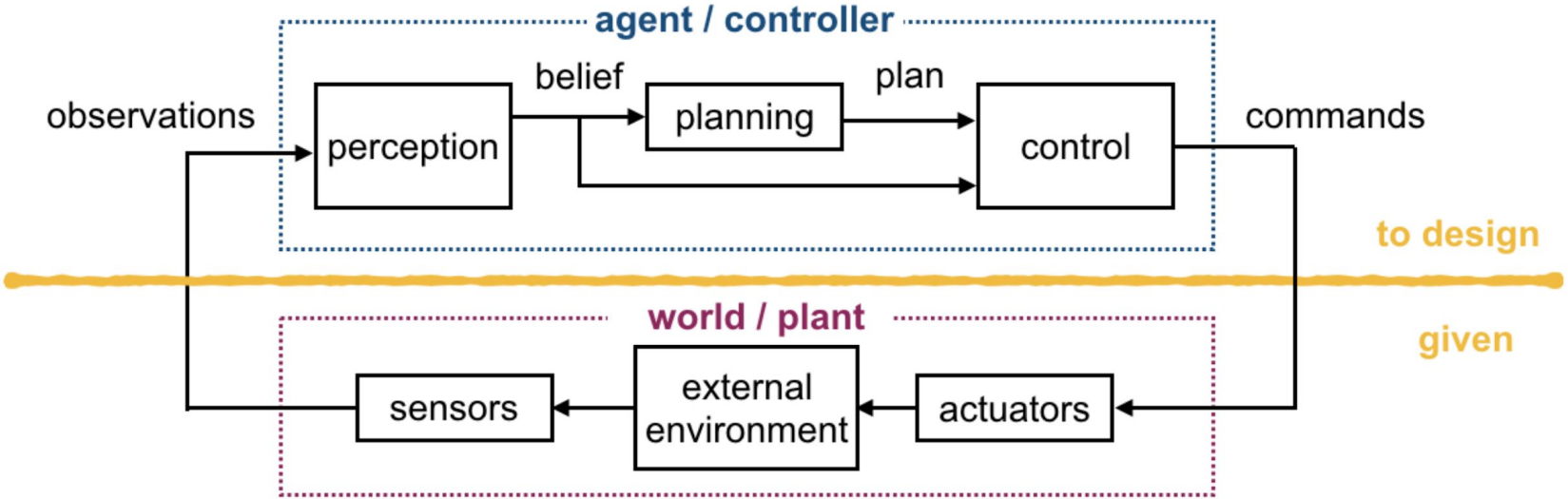
Benefits

- Computationally homogeneous
- Portability & runtime characteristics
- Predictable latency/accuracy tradeoffs
- Modularity, portability, agility

(Current) Limitations

- Low interpretability
- Unintuitive failure modes
- Software stack is immature
- Difficult to train and test

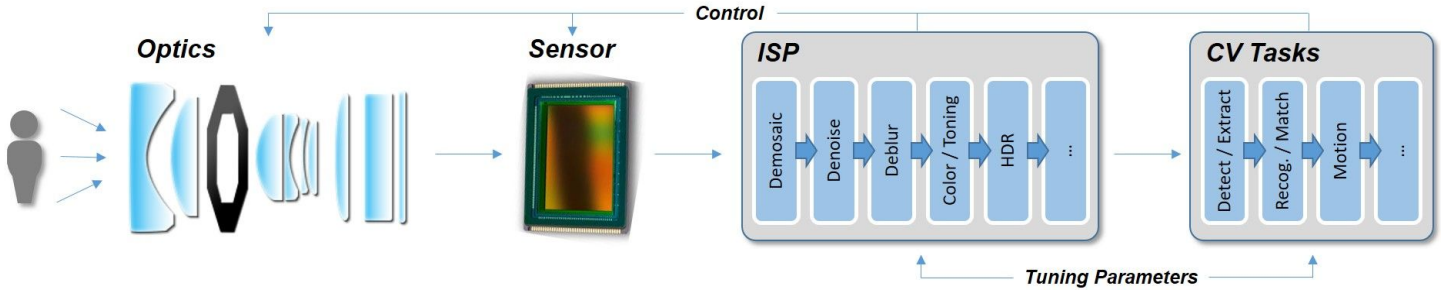
Robotics is a changin’



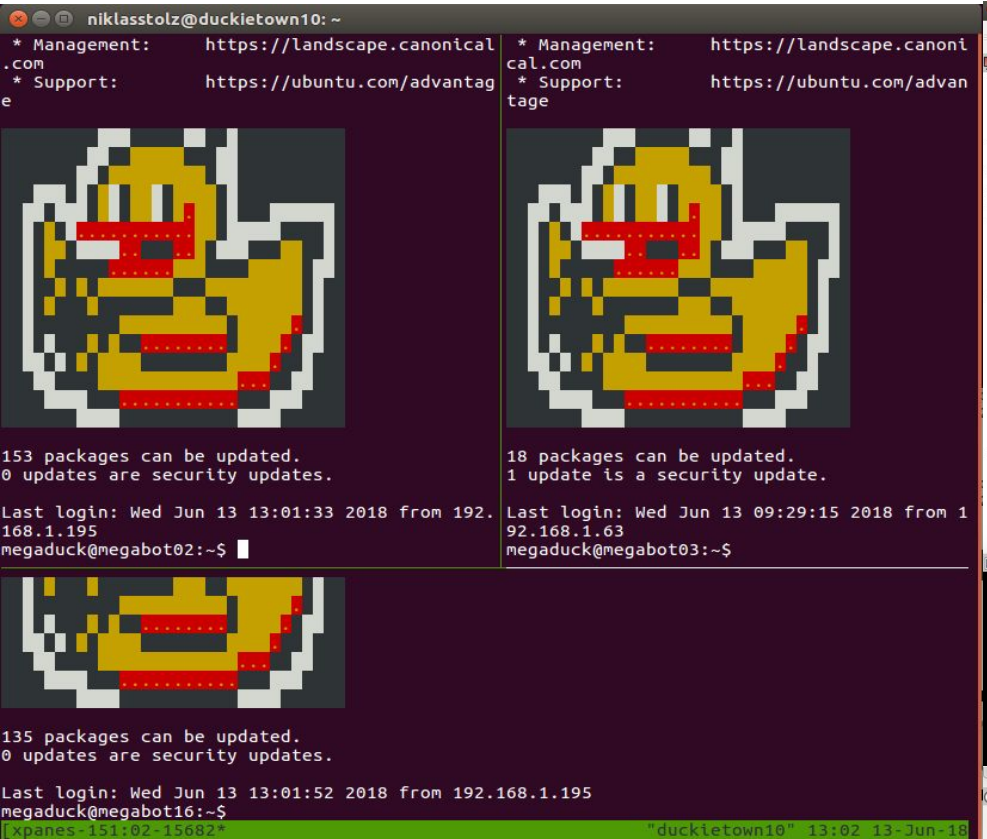
Idealized Agent

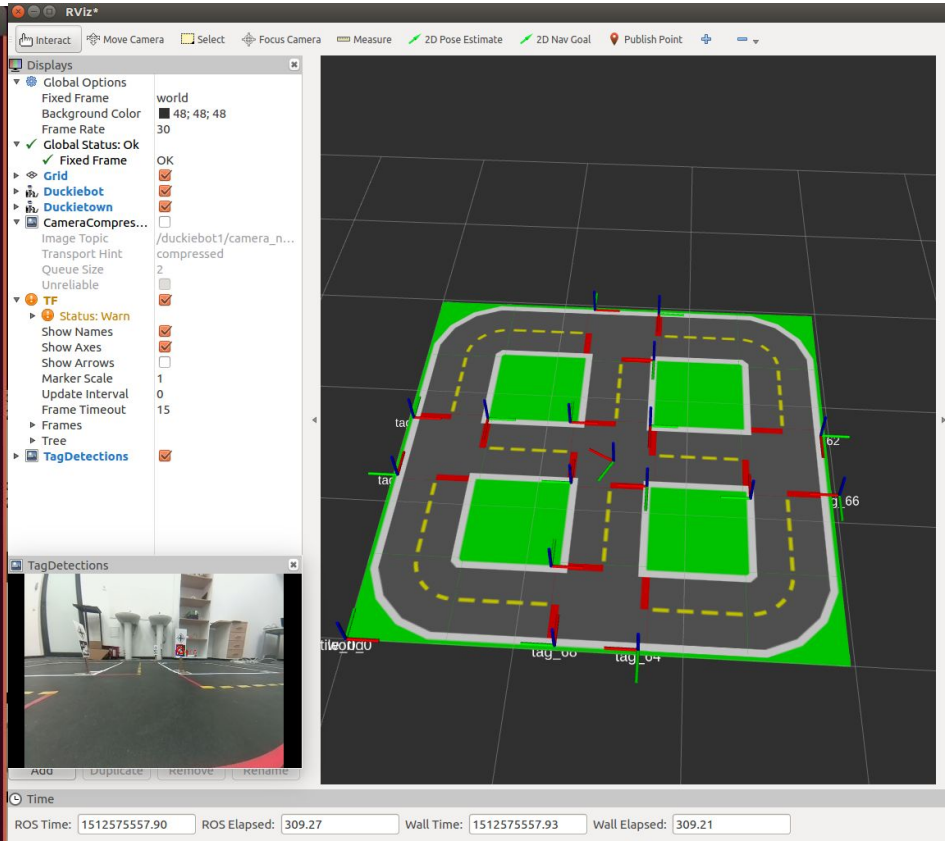
Environment

Reality



ROS / Duckietown Tools (rviz, rqt_*, sh)





Let’s try to make installs more repeatable

Step 1. Partition hard drive

Step 2. Install Ubuntu

Step 3. Install ROS

Step 4. Install Python stuff

Step 5. Configure network

…

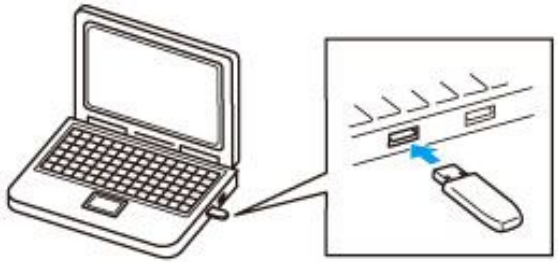
Step 98. `source environment.sh`

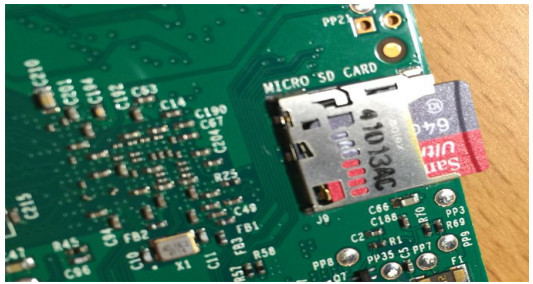
Step 99. `catkin_make -C ...`

```
$ dts init_sd_card
```

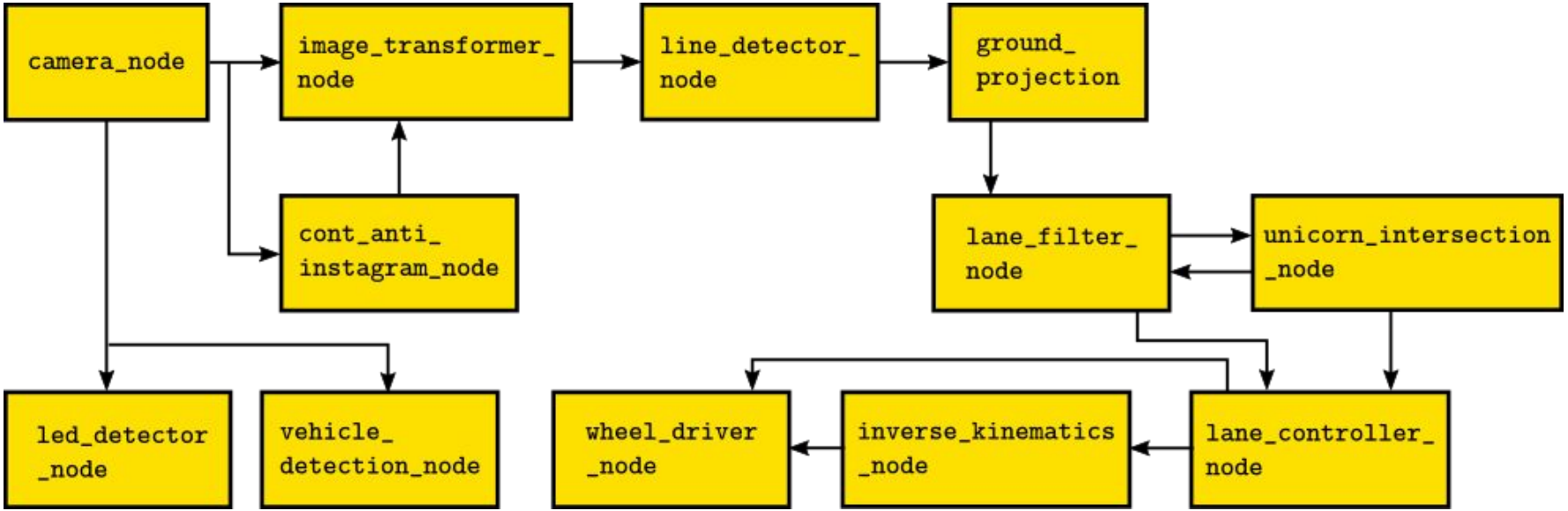
Containerization: A User Story

1. Type a short command, e.g. **$ dts init_sd_card**
2. Follow the installation wizard to flash an image.
3. Transfer flashed SD card to Duckiebot and boot.
4. Open a URL, e.g. http://duckiebot.local:9000/
5. Start or download a container, e.g. **duckietown/rpi-duckiebot-base**, **duckietown/gym-duckietown-agent**, **duckietown/rpi-joystick-demo**
6. Open a browser console and run, e.g. **roslaunch joystick ...**



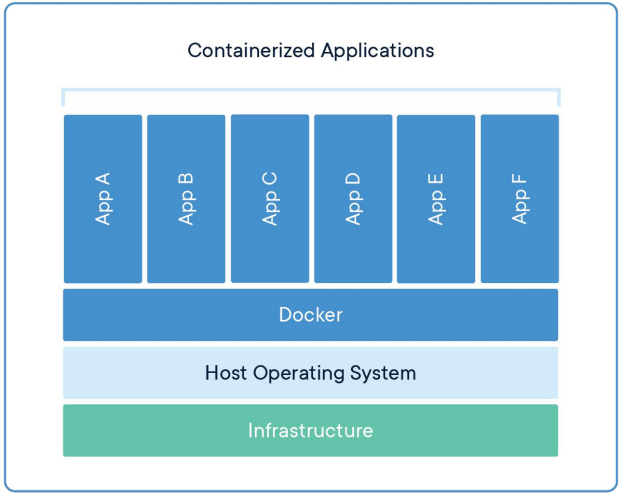


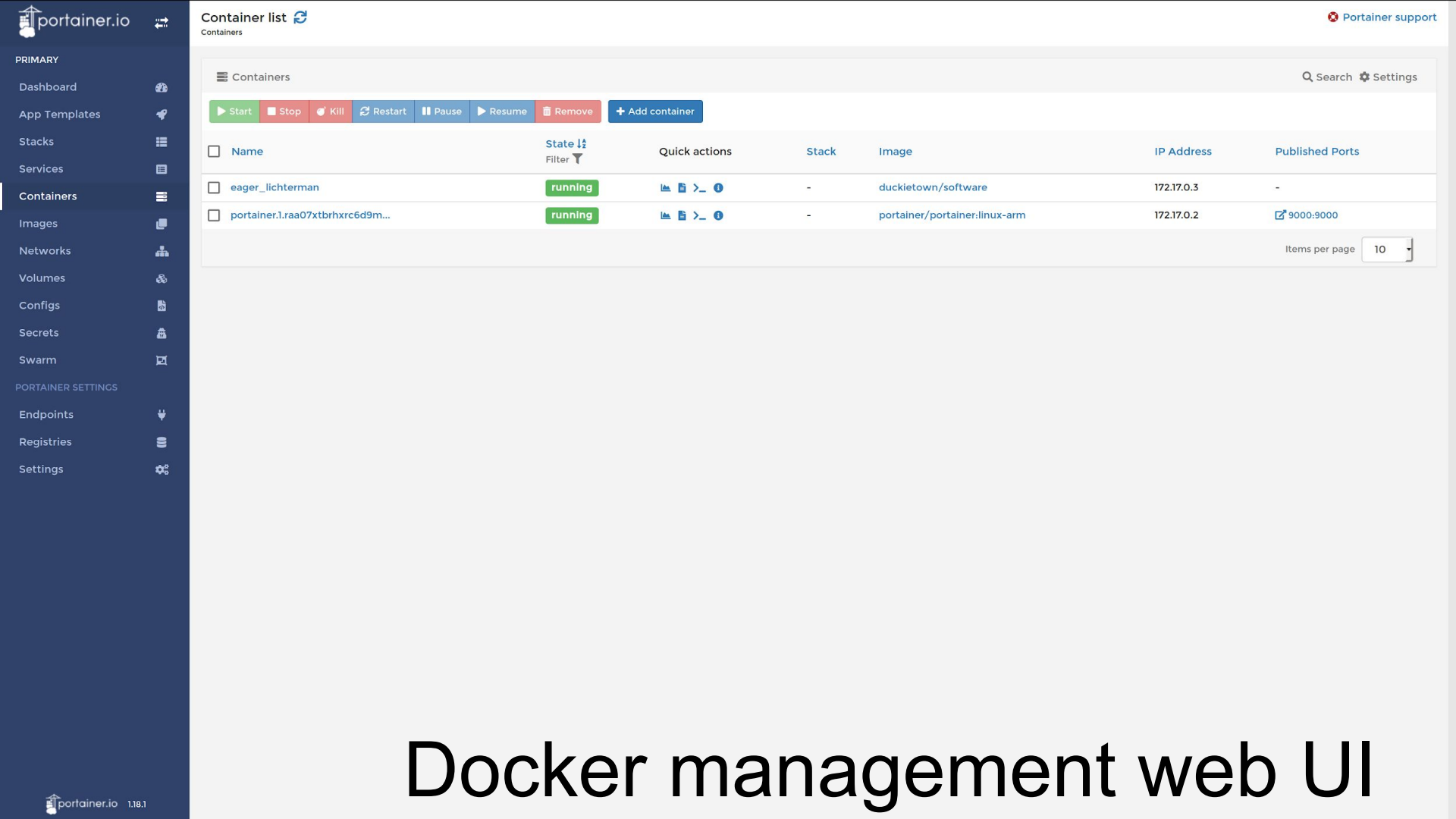
Duckietown ROS Nodes



Reproducible Builds & Containerization

- Benefits of containerization:
  - Reproducible build and deployment artifacts
  - Specified, documented software environments
  - Reusable, multi-platform applications
- Disadvantages:
  - Learning curve for Docker containers
  - GUI applications and X11
  - Migration complexity




portainer.io
Container list
Containers
Portainer support
PRIMARY
Dashboard
App Templates
Stacks
Services
Containers
Images
Networks
Volumes
Configs
Secrets
Swarm
PORTAINER SETTINGS
Endpoints
Registries
Settings
Containers
Search
Settings
Start
Stop
Kill
Restart
Pause
Resume
Remove
Add container
Name
State
Filter
Quick actions
Stack
Image
IP Address
Published Ports
eager_lichterman
running
-
duckietown/software
172.17.0.3
-
portainer.1.raa07xtbrhxrc6d9m...
running
-
portainer/portainer:linux-arm
172.17.0.2
9000:9000
Items per page
10
portainer.io 1.18.1
Docker management web UI

portainer.io

PRIMARY

- Dashboard
- App Templates
- Stacks
- Services
- Containers
- Images
- Networks
- Volumes
- Configs
- Secrets
- Swarm

PORTAINER SETTINGS

- Endpoints
- Registries
- Settings

portainer.io 1.18.1

Container console

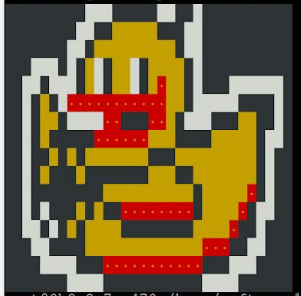
Containers > eager_lichterman > Console

Portainer support

Console

Exec into container as `default user` **using command** `bash` Disconnect

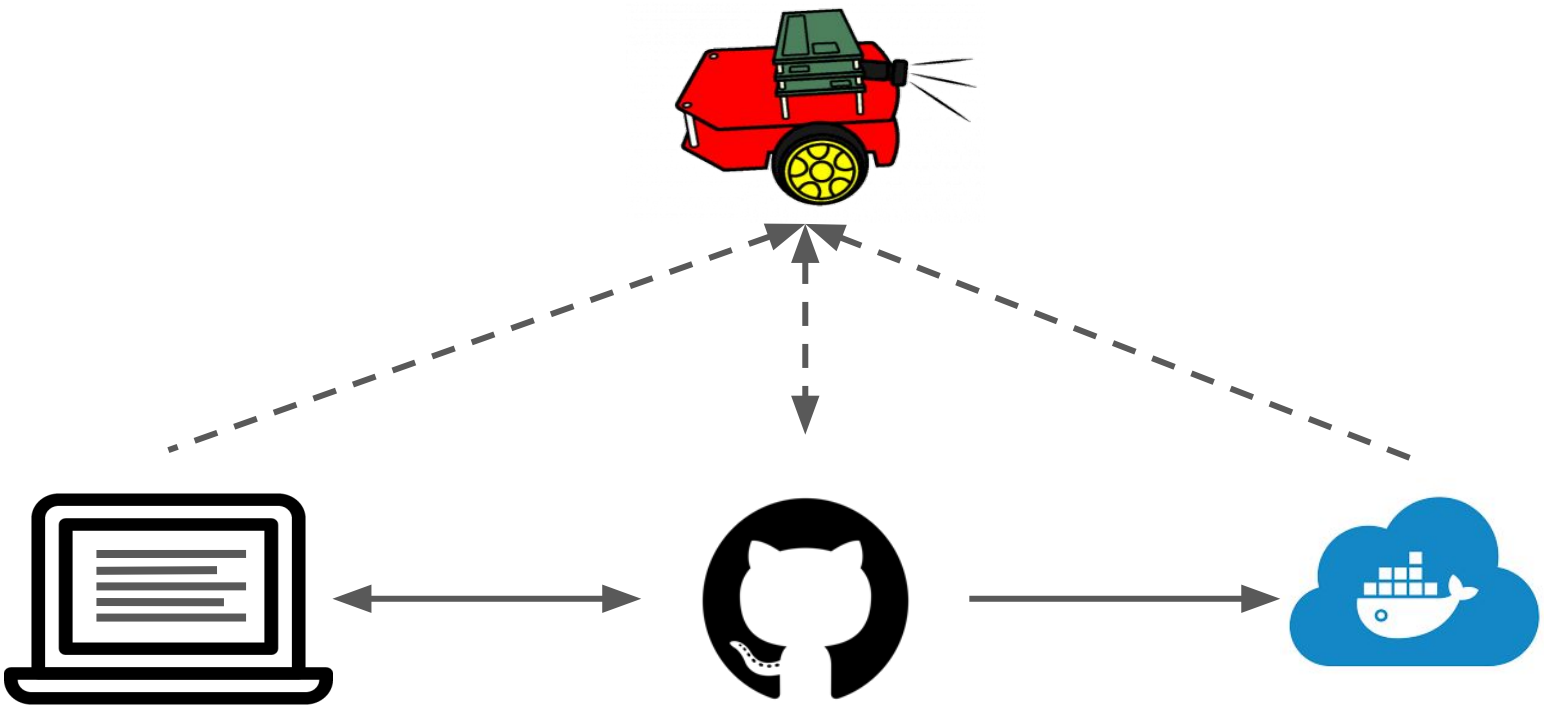
```
Activating ROS with shell: /bin/bash
Set ROS_HOSTNAME to: localhost
Set DUCKIETOWN_ROOT to: /home/software
Set PYTHONPATH to: /home/software/catkin_ws/src:/opt/ros/kinetic/lib/python2.7/dist-packages
Activating development environment...
```



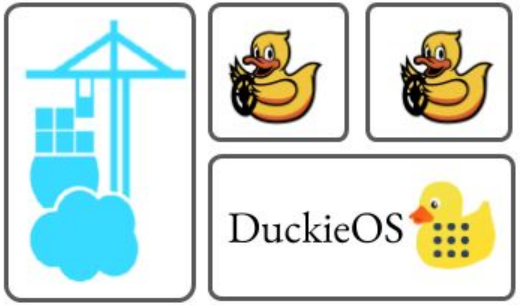
```
root@8b8c8a7ec470:/home/software# ls
LICENSE.md  Makefiles  catkin_ws              dependencies_for_duckiebot.sh  docker       hardware  out.jpg           set_ros_master.sh    shortcuts
Makefile    README.md  dependencies_common.sh  dependencies_for_laptop.sh     environment.sh  misc      requirements.txt  set_vehicle_name.sh  what-the-duck
root@8b8c8a7ec470:/home/software#
```

Docker management web UI

Containerization: Deployment Models



Containerization: Classic (ROS) Stack



Demos and ROS nodes

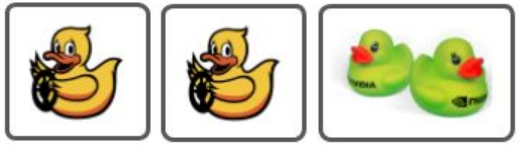
Based on ARM32v7, ROS Kinetic Kame,
Ubuntu Xenial Xerus, Python 2.7

Docker Client (via get.docker.com)

Lightweight base operating system

ARM-based single board computer (SBC)

Containerization: Laptop / Cloud Stack



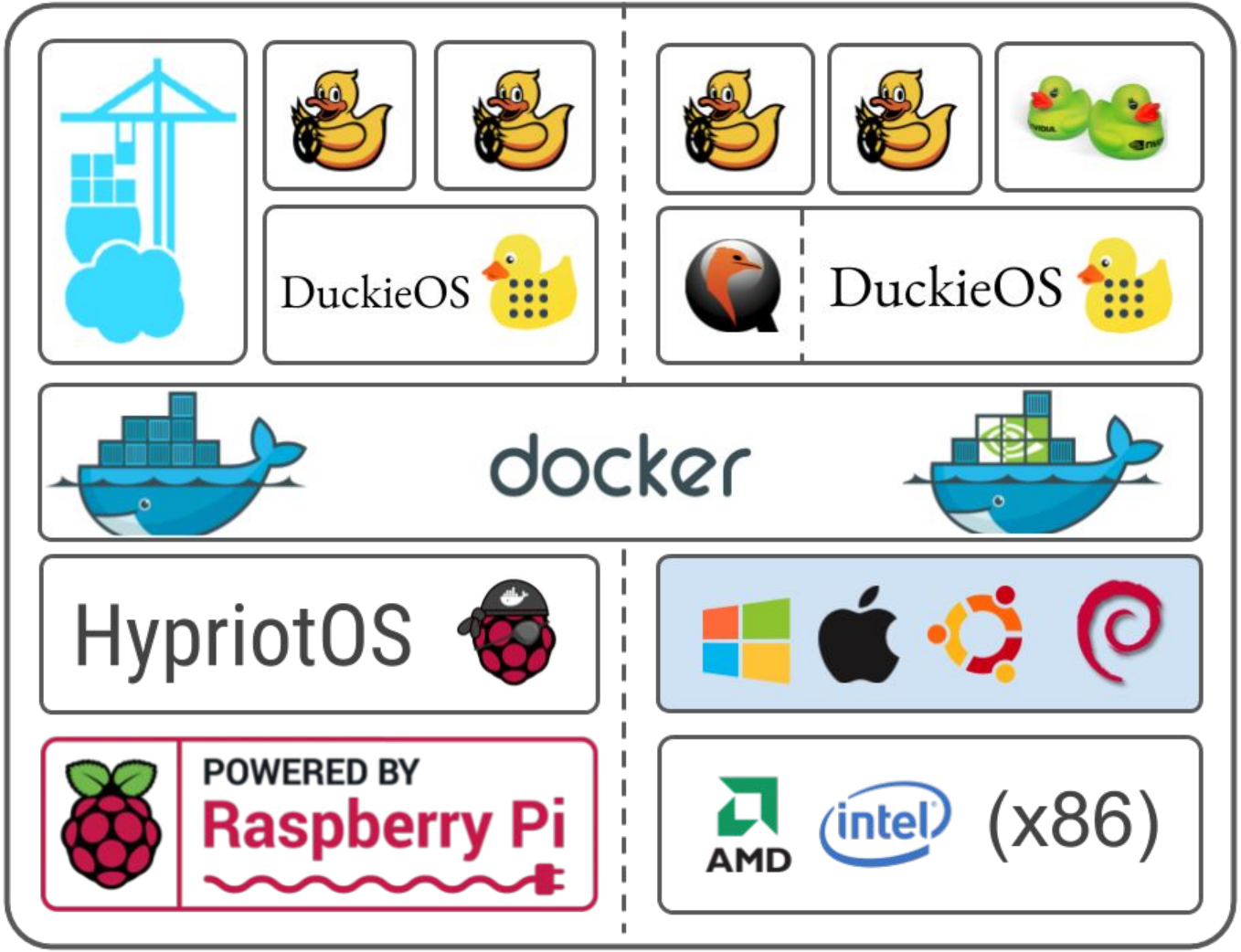
Demos and ROS nodes

Based on ARM32v7, ROS Kinetic Kame,
Ubuntu Xenial Xerus, Python 2.7

Docker Client (via get.docker.com)

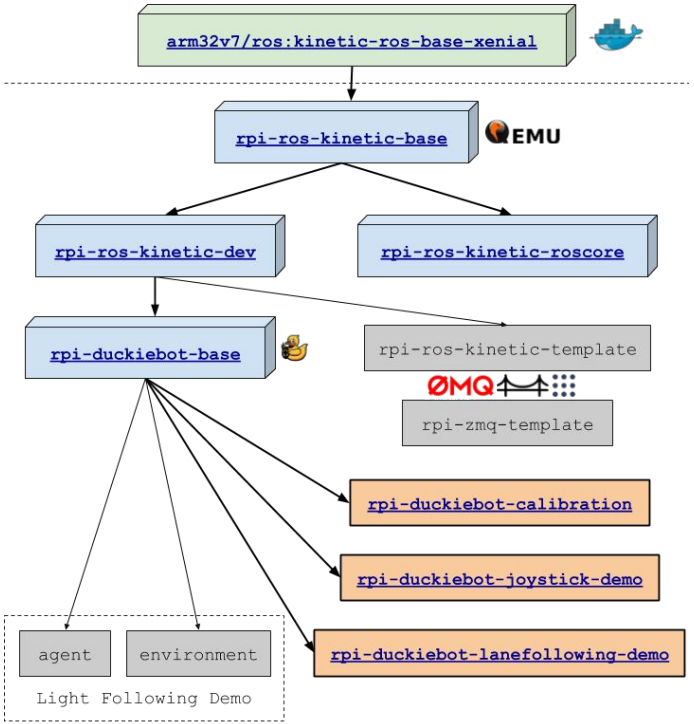
Any major OS (Windows/MacOS/Deb)

Any x86 compatible architecture will do


DuckieOS
DuckieOS
docker
HypriotOS
POWERED BY
Raspberry Pi
AMD
intel
(x86)

Lessons Learned

- Be careful with Docker inheritance
  - Rebuilds can play havoc on a large tree
- Use a versioning scheme from the outset
- Don’t try to over automate
  - There is still value in teaching manual commands
- Compile nodes and run tests in the build
  - Prevents changes from propagating downstream
- Utilize emulator tools for cross-building
- Don’t compile libraries unnecessarily
  - PiWheels et al. have precompile binaries for ARM
  - Long builds will slow down your development
- Utilize caching whenever possible



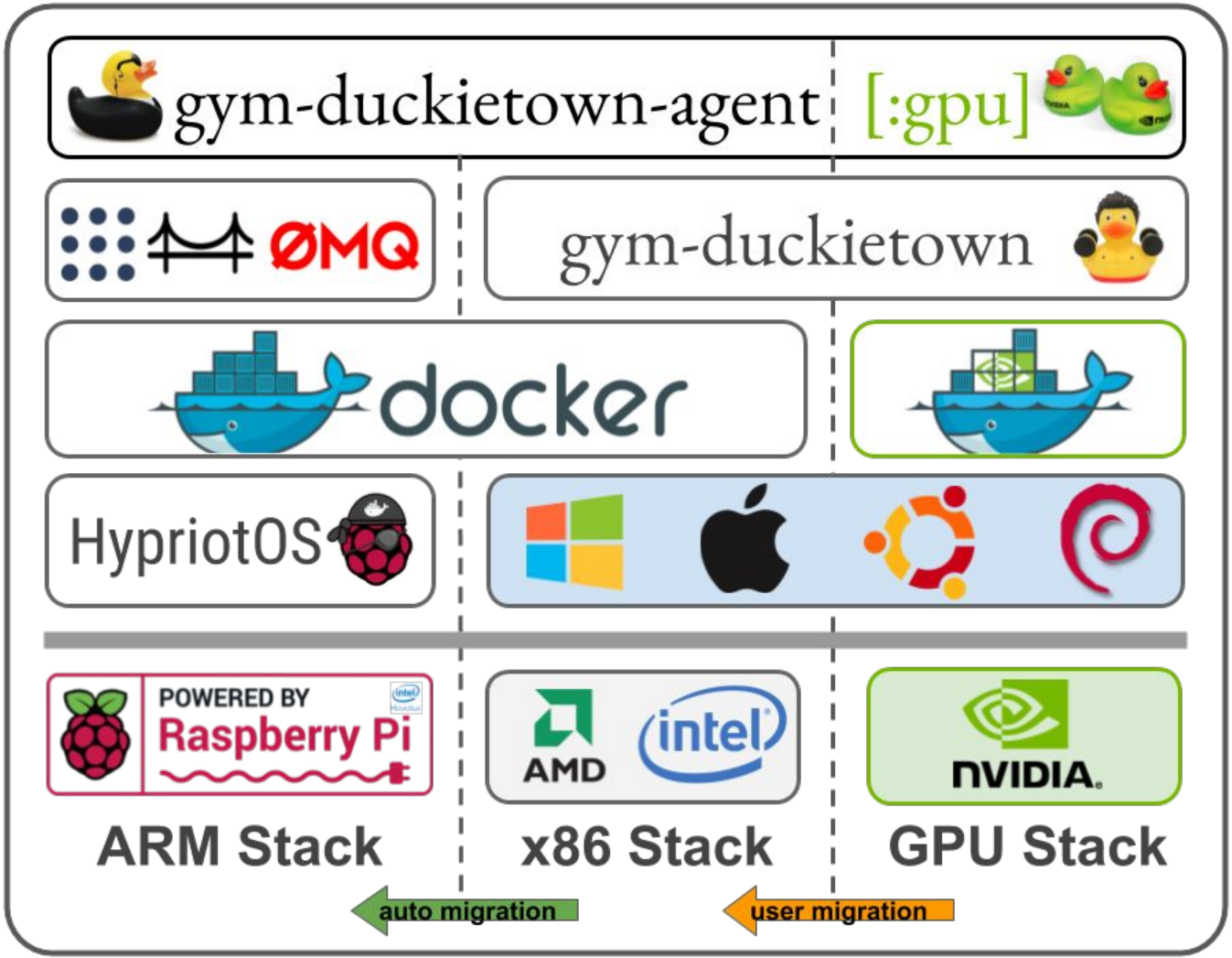
AI Driving Olympics

Three principal challenges:

- Lane following
- Lane following with obstacles
- Fleet management / mobility on demand

We evaluate your submission in a simulator and run it on a real Duckiebot!

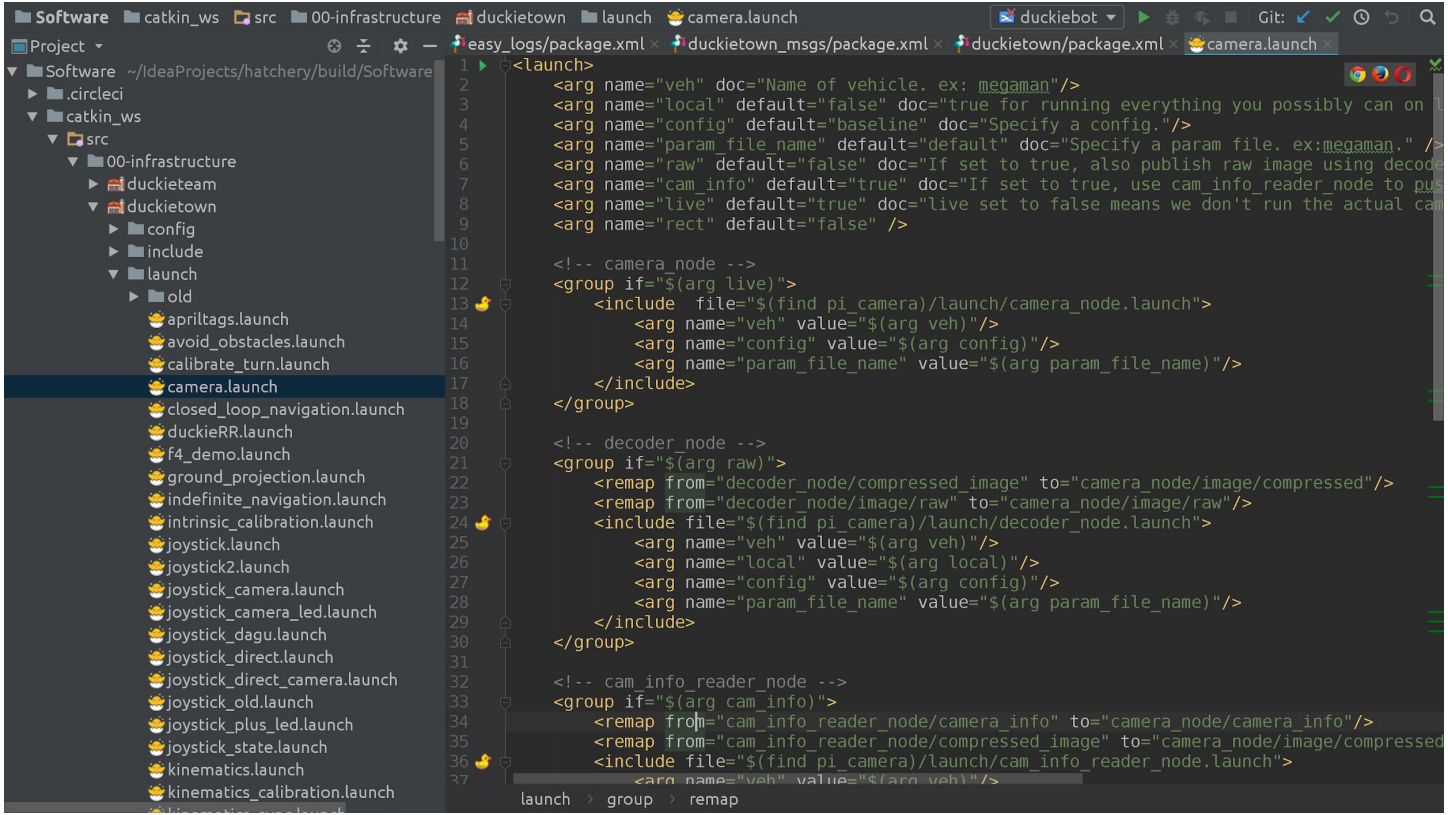
Coming to NIPS 2018 and ICRA 2019. Register today at duckietown.org


gym-duckietown-agent
[:gpu]
ØMQ
gym-duckietown
docker
HypriotOS
POWERED BY
Raspberry Pi
AMD
intel
NVIDIA
ARM Stack
x86 Stack
GPU Stack
auto migration
user migration

Recap and future work

- Two groups of users: researchers and students
  - One wants reproducibility, both want user friendly tooling
- Need to facilitate comparability with baseline implementations
- Simplify deployment model to fleet with easy rollbacks
- Gracefully degrade services and exert precise control over QOS
- Docker helps us achieve this, but it is not a silver bullet for reproducibility
- By utilizing emulation, we can gradually deploy and fail early
- Hardware-in-the-loop testing can give us further predictability

Help Wanted: Hatchery, a ROS IDE



Learn more at github.com/duckietown/hatchery

Special Thanks

Rusi Hristov

Liam Paull

Andrea Censi