OCI | WE ARE SOFTWARE ENGINEERS.

GalecinoCar:
A Self-Driving Car in Micronaut

August 10, 2018

© 2018, Object Computing, Inc. (OCI). All rights reserved. No part of these notes may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior, written permission of Object Computing, Inc. (OCI)

Ryan Vanderwerf

- Software Engineer on Grails/Micronaut team at OCI
- Father of 2 kiddos 6 and 13
- Talk to me if you need Grails, Groovy, or Micronaut support

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Agenda

- The Project
- The Hardware
- The Software
- The Future

© 2018, Object Computing, Inc. (OCI). All rights reserved.

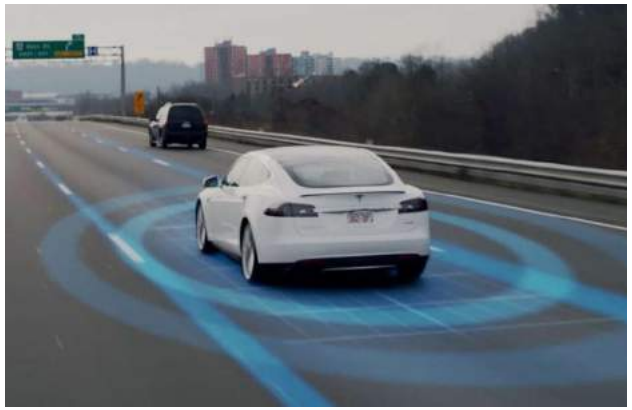
The Project

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Real World Applications

OCI WE ARE SOFTWARE ENGINEERS.

Tesla



Cadillac Super Cruise

Nvidia

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Our Cars

© 2018, Object Computing, Inc. (OCI). All rights reserved.

The Hardware

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Chassis

- Exceed RC Magnet Truck
- 1/16 scale
- Available for about $100 USD
- https://www.nitrorcx.com/51c853-savared-24-ghz.html

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Processing

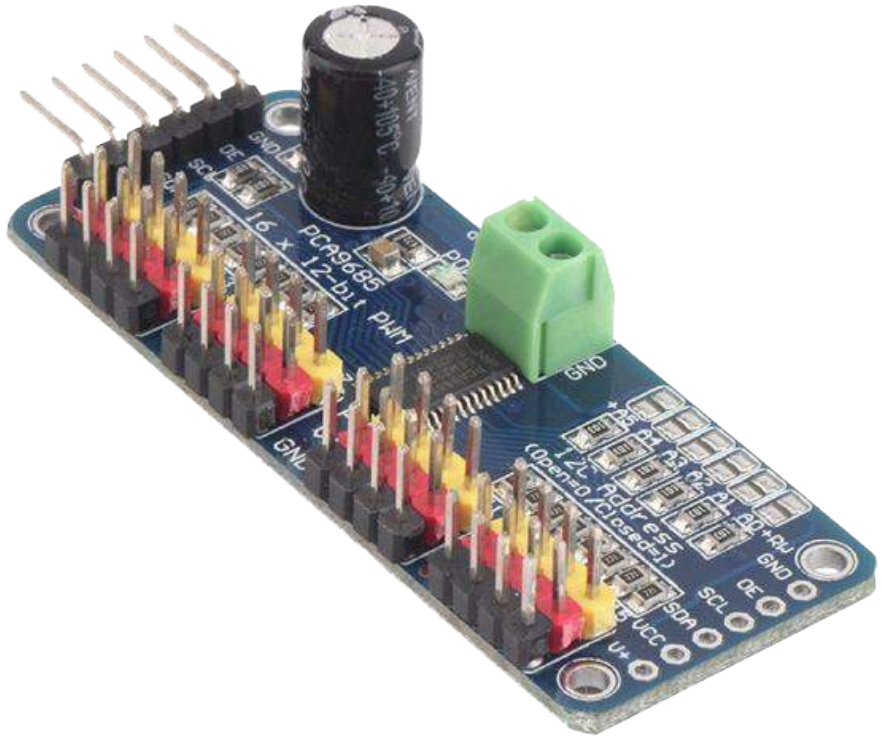
- Raspberry Pi 3B
- 1.2GHz 64 bit quad-core ARM CPU
- 1 GB RAM
- 802.11n wireless adapter
- Camera interface

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Servo/Motor Driver

OCI | WE ARE SOFTWARE ENGINEERS.

- Servo Driver PCA 9685
- 16 Channel
- 12 bit PWM Servo Motor Driver
- Adafruit is most common version
- Costs about $12 USD



© 2018, Object Computing, Inc. (OCI). All rights reserved.

Camera

- Wide Angle Raspberry Pi Camera
- This is used as input for self-driving model
- Costs less than $30 USD
- https://amzn.to/2K5sLqF

© 2018, Object Computing, Inc. (OCI). All rights reserved.

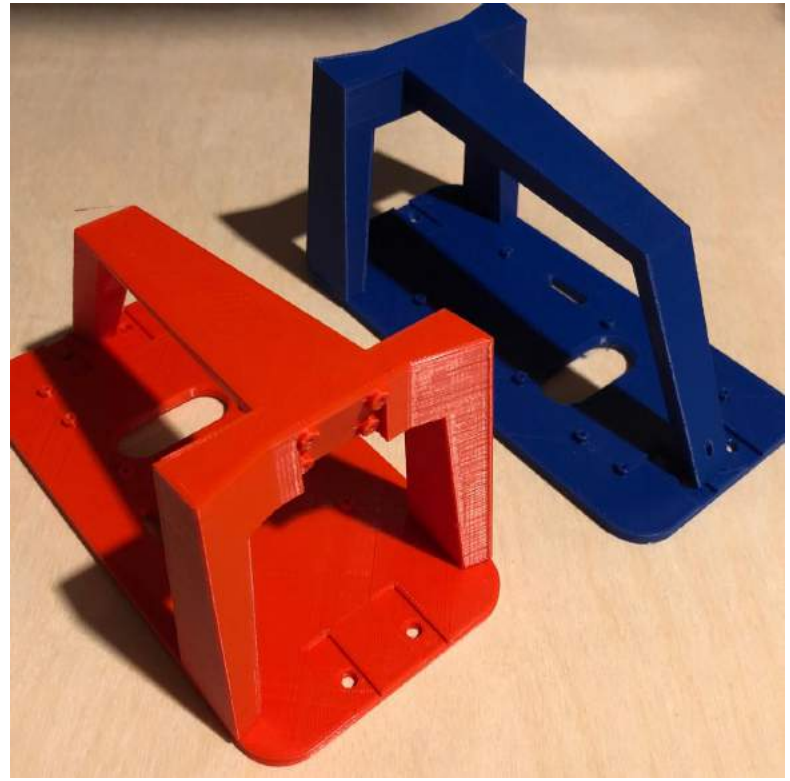
Battery

- Anker Astro E1 Candy-Bar Sized Ultra Compact Portable Charger
- This powers the Raspberry PI Board NOT the motors and servos
- Important because motors draw a lot of power that can make the PI unstable so separate power sources are needed
- Costs $20 USD
- https://amzn.to/2KMNOiB

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Rollcage

- 3D printed
- 2 pieces
- Plate for mounting Pi and 12 bit pwm controller
- Cage has built in mount for camera
- Built to Donkeycar specs
- https://bit.ly/2HZEtGE



© 2018, Object Computing, Inc. (OCI). All rights reserved.

The Training

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Teen Driving

© 2018, Object Computing, Inc. (OCI). All rights reserved.

A Track

© 2018, Object Computing, Inc. (OCI). All rights reserved.

A Track

© 2018, Object Computing, Inc. (OCI). All rights reserved.

A Track

© 2018, Object Computing, Inc. (OCI). All rights reserved.

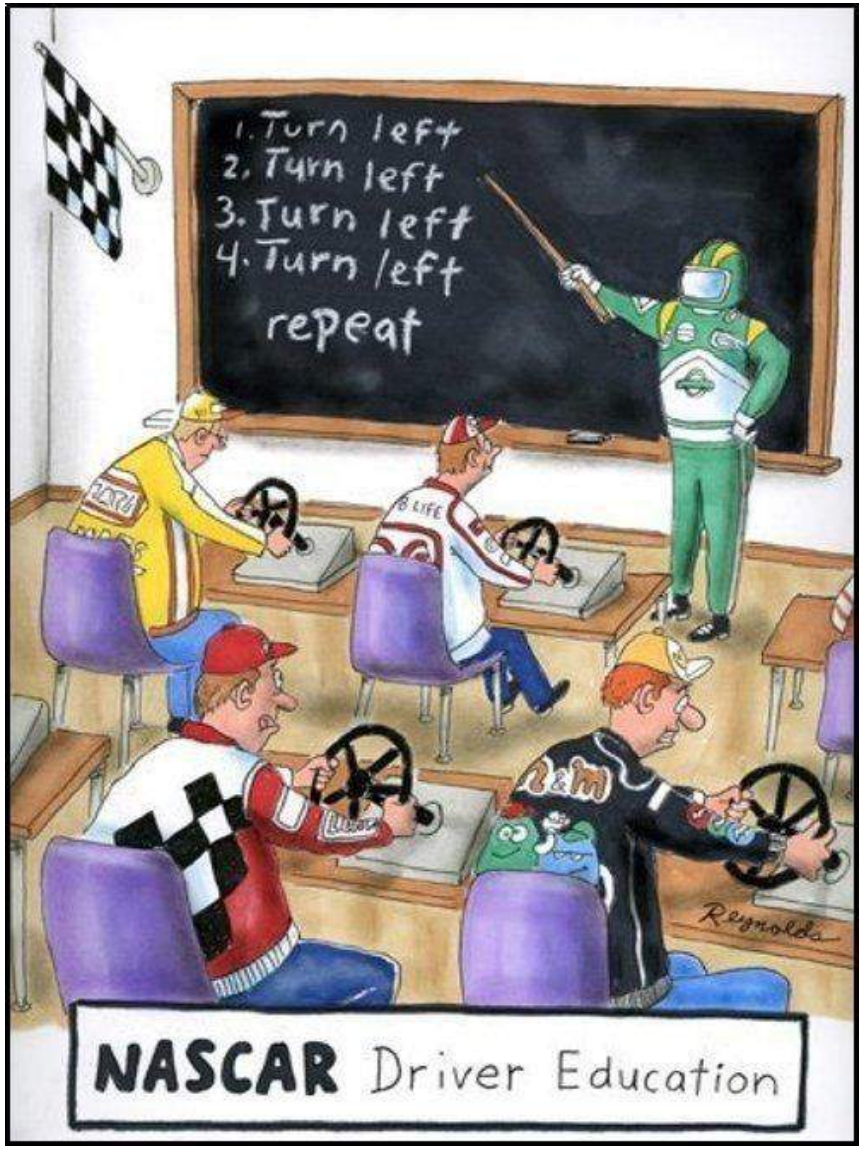
A Track

- Shape doesn’t matter so much
  - Loop
  - Figure 8
  - Curves
  - Straightaways
- Bright lines
- Well lit area
- Variety of materials
  - Tape
  - Ribbon
- Create two if possible

© 2018, Object Computing, Inc. (OCI). All rights reserved.

A Track

- Running reverse track
- May require retraining



© 2018, Object Computing, Inc. (OCI). All rights reserved.

Training the car

- Need manual control of the car
  - Web controller
  - Game controller
- Teach the car by example
- Drive the car 10-20 laps
- Car records data
  - Images
  - Car telemetry
- Train the model

© 2018, Object Computing, Inc. (OCI). All rights reserved.

The Software

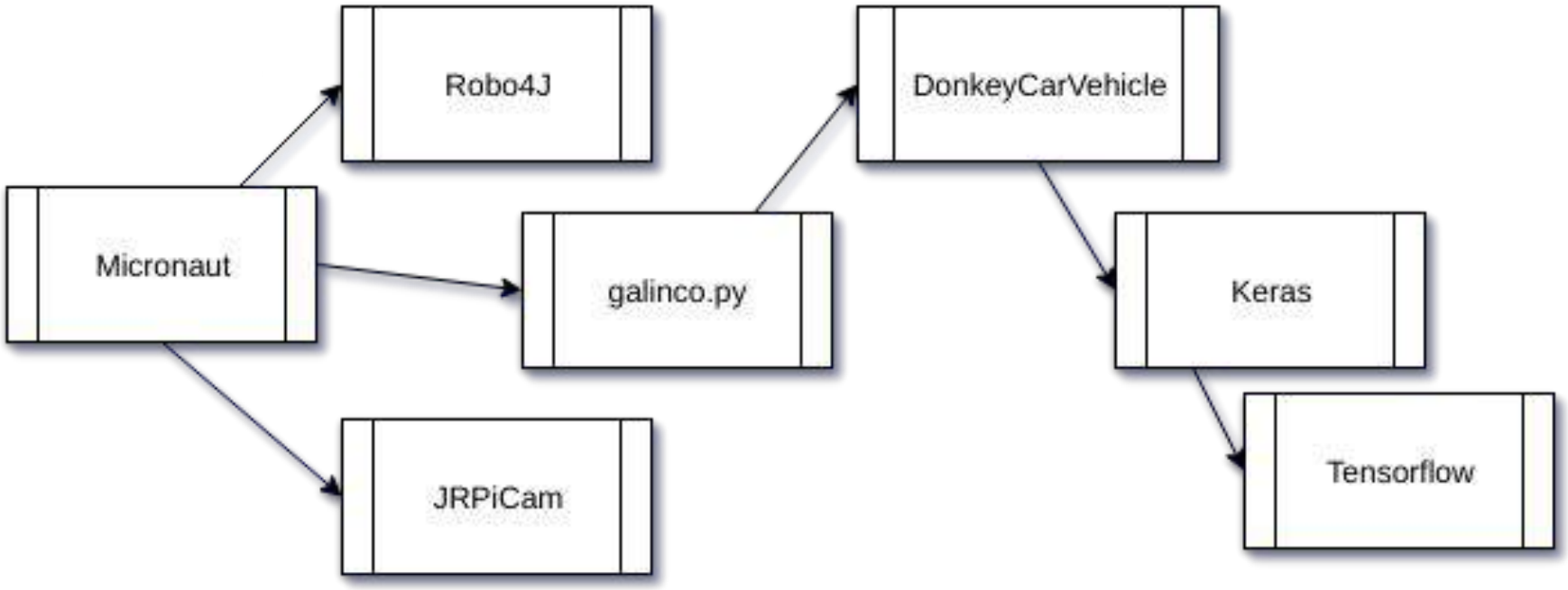
© 2018, Object Computing, Inc. (OCI). All rights reserved.

Software Components

- Micronaut
- Robo4J
- Keras
- Tensorflow
- DonkeyCar - currently for self-driving replaced in future revisions

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Software Components



© 2018, Object Computing, Inc. (OCI). All rights reserved.

What is Micronaut?

- A modern, JVM-based, full-stack framework for building modular, easily testable microservice applications
- Features built in dependency injection, auto configuration, configuration sharing, HTTP routing, fast configuration, and load balancing
- Fast startup time, around 1s on a well equipped development machine
- Supports Java, Groovy and Kotlin
- Has support for distributed tracing with third party tools like zipkin
- Includes a nice CLI for generating code and a console
- Profile support for creating projects of various usage types
- Can act as a server, client, or serverless function

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Micronaut

- What does Micronaut have to do with an RC car?
  - Because of its quick startup and small memory footprint makes it an ideal framework for the Raspberry Pi

© 2018, Object Computing, Inc. (OCI). All rights reserved.

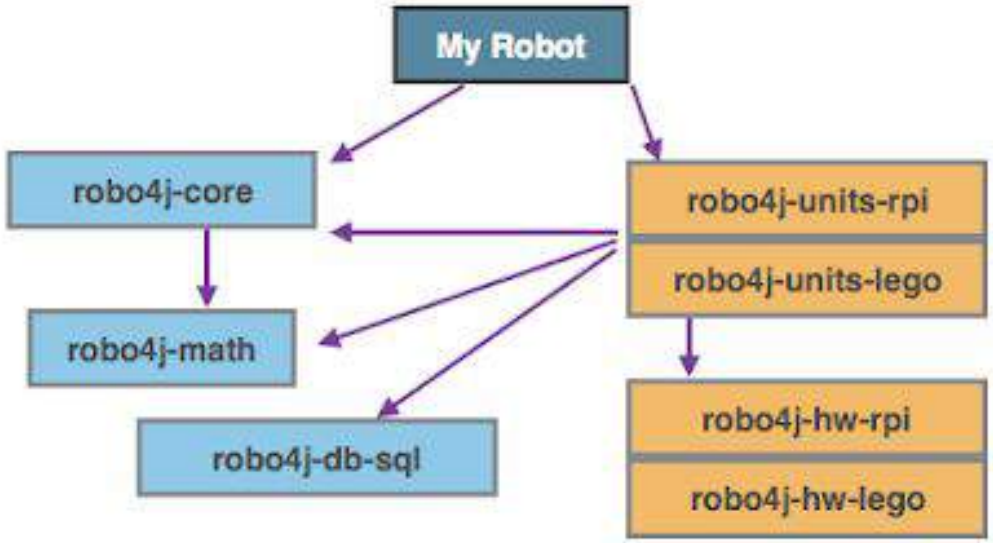
Robo4J

- IoT Robotics Library for Java
  - Robo4J is a framework for quickly getting started building and running robots and IoT devices
  - Winner - Duke's Choice Awards 2017
  - Maintained by Marcus Hirt (@hirt) and Miro Wengner (@miragemiko)
  - http://www.robo4j.io
  - We use these libraries to control the motors and servos
  - Native support for PI and Lego EV3 platforms

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Robo4J

- Contains many components
  - Robo4j-core
  - Robo4j-math
  - Robo4j-units rpi and lego
  - Robo4j-hw rpi and lego
  - Robo4j-db-sql for inline storage



© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar

- Galecino
  - Horse breed developed in Mexico, bred from horses brought from Spain by Hernán Cortés and other conquistadors
  - GalecinoCar is a port of DonkeyCar - a popular python RC project
  - Micronaut runs as the remote control for the RC car motors and servos
  - Serves up remote control UI with camera to see what the car sees
  - Can toggle between remote control and native Keras/Tensorflow self-driving

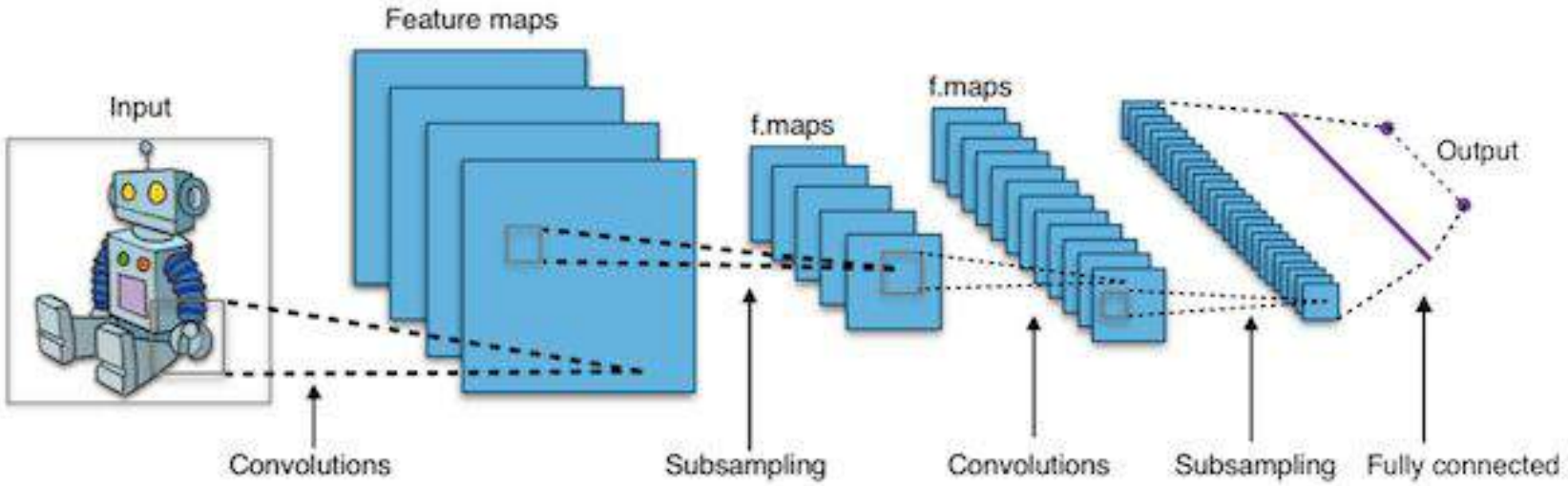
© 2018, Object Computing, Inc. (OCI). All rights reserved.

Keras

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Keras

Convolutional Neural Network



© 2018, Object Computing, Inc. (OCI). All rights reserved.

Keras

- What is Keras?
  - Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano
  - Uses a convolutional neural network (CNN) that you train to drive like a human
  - This is currently controlling the pilot model, which is trained from driving the car around a track about 9 or 10 times
  - The goal is to replace this with DeepLearning4J or proper Tensorflow java support once it becomes stable for 32bit PI systems
  - After training on a track we create the model and import it to Keras
  - Once the model is trained it will control throttle and steering automatically or just steering
  - Instructions for training: http://docs.donkeycar.com/guide/train_autopilot/

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Tensorflow

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Tensorflow

- A Machine learning framework by Google
  - Keras is running on top of Tensorflow to simplify building
  - Pilots from Keras feeds data into Tensorflow for self-driving of car
  - The most popular machine learning framework
  - Currently Java support doesn’t work on 32 bit systems
  - Projects like DeepLearning4J have been built as better solution for JVM users
  - You can create models for almost anything for machine learning
  - Can be applied to all sorts of things like lip reading, chat, object recognition, touch and art
  - Cool list of interesting projects at https://bit.ly/1SVAKpD

© 2018, Object Computing, Inc. (OCI). All rights reserved.

DonkeyCar

© 2018, Object Computing, Inc. (OCI). All rights reserved.

DonkeyCar

- Donkeycar
  - Python project
- GalecinoCar is a port of DonkeyCar
  - Because Deeplearning4J and Tensorflow Java JNI isn’t ready on the Pi we still need a part of it
  - The most promising is Deeplearning4J
  - We have pre-done the training and loaded the model
  - GalecinoCar bridge in Micronaut allows UI switching between remote control and self-driving modes

© 2018, Object Computing, Inc. (OCI). All rights reserved.

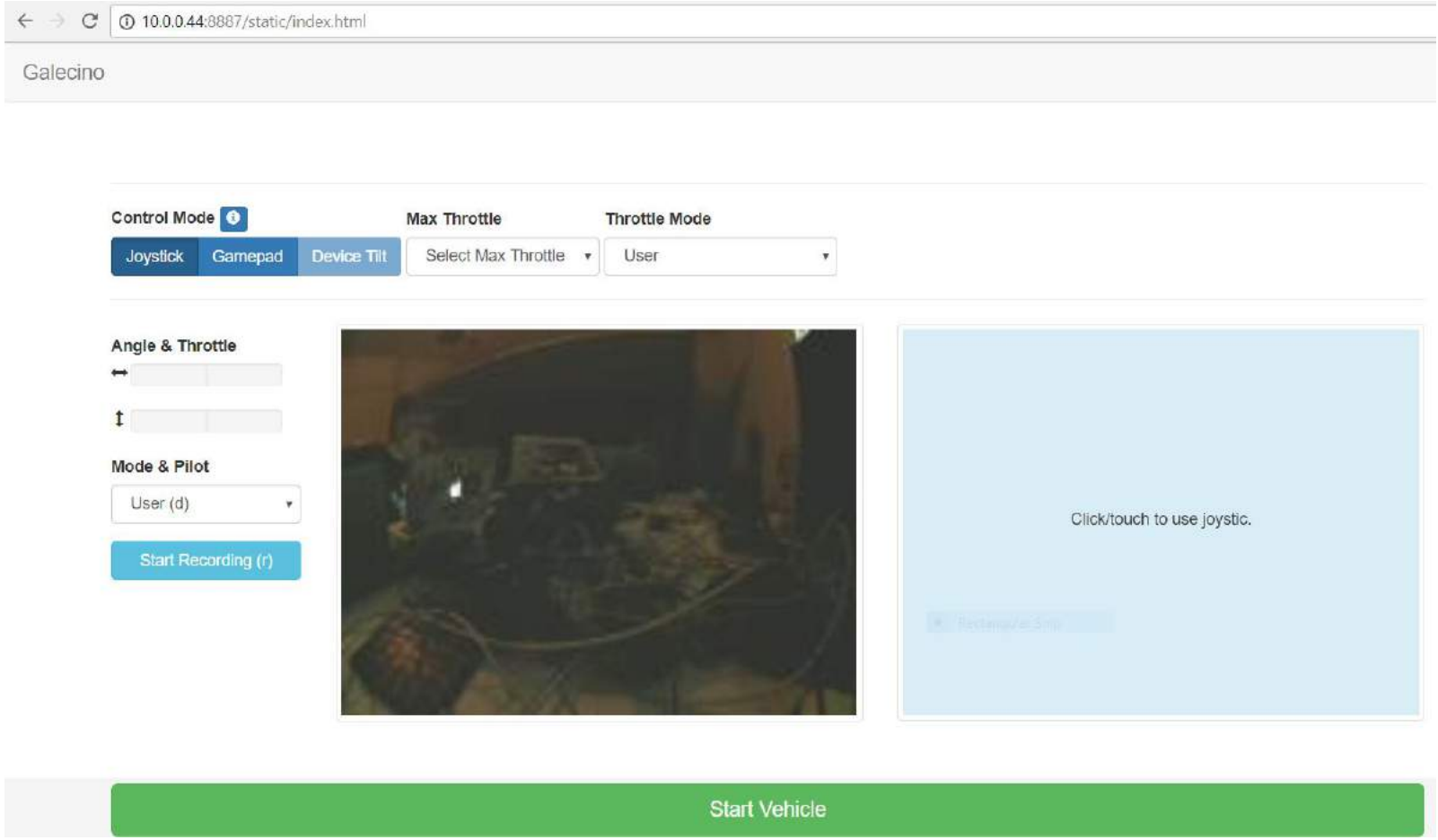
GalecinoCar

OCI | WE ARE SOFTWARE ENGINEERS.

- GalecinoCar Code Walk Through

© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar - Web UI



© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar


```groovy
@Controller("/")
@Singleton
class VehicleController {

    protected static final Logger LOG = LoggerFactory.getLogger(VehicleController.class);

    @Inject
    VehicleService vehicleService

    @Value('${galecino.servo.trim:0.0}')
    protected float configTrim


    List<Vehicle> index() {
        vehicleService.list()
        vehicleService.pwmTest()
    }

    @Transactional
    @PostConstruct
    void setup() {
        //vehicleService.save 'VanderfoxCar'
        LOG.info("called setup from VehicleController")
        vehicleService.init()
        LOG.info("finished setup from VehicleController")
    }
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar


```
@Get(produces = "image/jpeg")
HttpResponse<byte[]> video() {
    byte[] image = takeStill()
    if (!image) {
        image = takeStill() //retry sometimes its null
    }
    return HttpResponse.ok(image)
            .header( name: "Content-type", value: "multipart/x-mixed-replace;boundary=--boundarydonotcross")
}
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar


```
@Get(produces = 'text/html')
HttpResponse<String> drive(float angle, float throttle, String drive_mode = "user", Boolean recording = false) {
    //vehicleService.steer(angle)
    System.out.println("drive called")
    vehicleService.driveScheduled(angle,throttle, drive_mode, recording)
    //vehicleService.drive(angle,throttle)
    return HttpResponse.ok( body: "angle:${angle} throttle:${throttle}")
}
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar


```
byte[] takeStill() {
    return vehicleService.takeStill()
}
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

```groovy
@Service(Vehicle)
abstract class VehicleService {

    //these vales seem double the donkeycar pwm vals
    private static final int SERVO_FREQUENCY = 20
    private static final int MOTOR_BACKWARD = 730
    private static final int MOTOR_FORWARD = 800
    private static final int MOTOR_STOPPED = 760
    private static final int STEERING_LEFT = 880
    private static final float STEERING_STRAIGHT_ANGLE = 44.0
    private static final int STEERING_RIGHT = 570
    private static final int MAX_THROTTLE_FORWORD = 1000
    private static final int MIN_THROTTLE_FORWORD = 800
    private static final int MAX_THROTTLE_BACKWARD = 610
    private static final int MIN_THROTTLE_BACKWARD = 715
    @Value('${galecino.servo.trim:0.0}')
    protected float configTrim
    @Value('${galecino.pwmFrequency:20}')
    protected int pwmFrequency
    Process autopilotThread

    abstract List<Vehicle> list()
    abstract Vehicle save(String name)

    RPiCamera piCamera
    Process process // this is the process python is running in pilot mode
    ArrayBlockingQueue commands
    def running = true
    def delay = 50
    Thread th
    ScheduledThreadPoolExecutor delayThread
    protected static final Logger LOG = LoggerFactory.getLogger(VehicleService.class);
    String currentDriveMode = "user"

    @PostConstruct
    void init() {
        LOG.info("Init thread started")
        delayThread = Executors.newScheduledThreadPool( corePoolSize: 1)
        commands = new ArrayBlockingQueue( capacity: 100)
        initThrottle( frequency: 20, on: 0,MOTOR_FORWARD) // make sure motor is ready
        Thread.sleep( millis: 100)
        initThrottle( frequency: 20, on: 0,MOTOR_STOPPED)
        Thread.sleep( millis: 100)
        startDriveThread()
        LOG.info("Init thread finished")
    }
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

```
byte[] takeStill() {
    if (!autopilotThread || !autopilotThread.alive) {
        if (!piCamera) {
            synchronized (this) {
                long startTime = System.currentTimeMillis()
                piCamera = new RPiCamera()
                piCamera.setAWB(AWB.AUTO)        // Change Automatic White Balance setting to automatic
                        .setTimeout(30)            // Wait 1 second to take the image
                        .setBrightness(60)
                        .turnOffPreview()            // Turn on image preview
                        .setEncoding(Encoding.JPG) //
                long endTime = System.currentTimeMillis()
                System.out.println("init camera took ${endTime - startTime}ms")
            }

        long startTime = System.currentTimeMillis()

        BufferedImage image = piCamera.takeBufferedStill( width: 160,  height: 120)
        long endTime = System.currentTimeMillis()
        System.out.println("camera pic took ${endTime - startTime}ms")
        startTime = System.currentTimeMillis()
        ByteArrayOutputStream baos = new ByteArrayOutputStream()
        if (!image) {
            image = piCamera.takeBufferedStill( width: 160,  height: 120)
        }
        if (image) {
            ImageIO.write(image,  formatName: "jpg", baos)
            byte[] imageOut = baos.toByteArray()
            endTime = System.currentTimeMillis()
            System.out.println("pic jpg convert took ${endTime - startTime}ms")
            imageOut
        } else {
            null
        }
        } else {
            piCamera.stop()
            piCamera = null
        }
    } else {
        if (autopilotThread) {
            LOG.info("autopilot thread alive=${autopilotThread.alive}")
        }
    }

}
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

```
private void startDriveThread() {

    th = Thread.start {
        try {
            LOG.info("inside thread")
            while (running) {
                def recent = []

                commands.drainTo(recent)
                LOG.info("recent="+recent)

                if (recent.size()) {
                    LOG.info recent.size().toString()
                    def command = recent[-1]
                    float throttle = command.throttle
                    LOG.info command.direction
                    switch (command.direction) {
                        case 'forward':
                            if (process && process?.alive) {
                                process.destroyForcibly()
                            }
                            steer(command.angle)
                            int pulse = 0
                            if (throttle > 0) {
                                pulse = map_range(throttle,
                                        X_min: 0, X_max: 1,
                                        MIN_THROTTLE_FORWORD, MAX_THROTTLE_FORWORD)
                                LOG.info("fwd Pulse=${pulse} throttle:"+throttle)
                            } else {
                                if (throttle < 0) {
                                    pulse = map_range(throttle,
                                            X_min: -1, X_max: 0,
                                            MAX_THROTTLE_BACKWARD, MIN_THROTTLE_BACKWARD)
                                    LOG.info("backwd  Pulse=${pulse} throttle:"+throttle)
                                }
                                if (throttle == 0) {
                                    LOG.info("stop")
                                    stop(pwmFrequency)
                                    return
                                }
                            }
                            // set throttle
                            forward(pwmFrequency, on: 0, pulse)
                            break
                        case 'stop':
                            stop(pwmFrequency)
                            break
                    }
                    if (commands.size() == 0) {
                        stop(pwmFrequency)
                    }
                }
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

```
void driveScheduled(float angle, float throttle, String driveMode = "user", Boolean recording = false) {
    String direction = "forward"
    currentDriveMode = driveMode
    int duration = 0
    // set steering
    LOG.info("drivemode="+driveMode)
    LOG.info("drivethread="+th+" status:"+th?.state+" isalive:"+th?.isAlive())
    if (driveMode == "user") {
        if (autopilotThread) {
            autopilotThread.destroyForcibly()
        }
        LOG.info("delayThread="+delayThread)
        if ((th && th.state == Thread.State.TERMINATED) || !th) {
            startDriveThread()
        }
        if (delayThread && delayThread.terminated) {
            init() // reset state
        }
        delayThread.schedule({
            LOG.info("command queued")
            commands.put([direction:direction, duration:duration, angle:angle, throttle:throttle])
        } as Runnable, delay, TimeUnit.MILLISECONDS)
    } else if (driveMode == "local") {
        //stop all remote control and reset motors?
            //stop all remote control and reset motors in case car is moving
            if (piCamera) {

                piCamera.turnOffPreview()
                piCamera.stop()

                Thread.sleep( millis: 500)
                //piCamera = null
            }
            stop()
            if (th) {
                th.stop()
            }
            if (delayThread) {
                delayThread.shutdownNow()
            }
            def out = new StringBuffer()
            def err = new StringBuffer()
            autopilotThread = "python /home/pi/d2/galencino.py --model /home/pi/d2/models/smartpilot".execute()
            autopilotThread.consumeProcessOutput( out, err )
            autopilotThread.waitFor()
            if( out.size() > 0 ) LOG.info out.toString()
            if( err.size() > 0 ) LOG.info err.toString()
            LOG.info("Autopilot started:"+autopilotThread.toString())
    }
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar


```
    /**
     * controls steering of the car
     * @param angle
     * @param trim
     */
    void steer(float angle, float trim = 0.0) {
        // seems like 360 right 520 left
        PWMPCA9685Device device = new PWMPCA9685Device()
        device.setPWMFrequency(50) //internetz says 50 for servos is the shiz
        Servo servo0 = new PCA9685Servo(device.getChannel( channel: 1))
        LOG.info("steer angle non corrected:${angle} trim:${trim}")
        if (trim != 0) {
            trim = configTrim
            servo0.setTrim(trim)
        }
        servo0.setInput((angle).toFloat())
        System.out.println("configTrim in service=${configTrim}")
        Thread.sleep( millis: 1000) // impor

    }
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar


```
    /**
     * this makes the motors 'wake up' with special pwm values.
     * @param frequency
     * @param on
     * @param off
     */
    void initThrottle(int frequency = pwmFrequency, int on = 0, int off = MOTOR_STOPPED) {
        PWMPCA9685Device device = new PWMPCA9685Device()
        device.setPWMFrequency(frequency)
        PWMPCA9685Device.PWMChannel motor0 = device.getChannel( channel: 0)
        LOG.info("init motor frequency:"+frequency+" on:"+on+" off:"+off)
        motor0.setPWM(on, off)
    }
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

GalecinoCar


```
167     /**
168      * controls the throttle on the car for forward and backward (based on pwm values
169      * @param frequency
170      * @param on
171      * @param off
172      */
173     void forward(int frequency = pwmFrequency, int on = 0, int off = MOTOR_FORWARD) {
174         PWMPCA9685Device device = new PWMPCA9685Device()
175         device.setPWMFrequency(frequency)
176         PWMPCA9685Device.PWMChannel motor0 = device.getChannel( channel: 0)
177         LOG.info("fwd frequency:"+frequency+" on:"+on+" off:"+off)
178         motor0.setPWM(on,off)
179         Thread.sleep( millis: 1000) // this is important or the motor doesn't have time to respond
180     }
181
```

© 2018, Object Computing, Inc. (OCI). All rights reserved.

The Future

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Phase I (current)

- Build DonkeyCar
- Convert from Python code to Groovy, Micronaut and Robo4J

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Phase II

- Switch from Keras and Tensorflow to Deeplearning 4J
- Deeplearning 4J can import Keras models and replace Tensorflow
- https://github.com/deeplearning4j/
- Currently build issues on PI are blocking this https://bit.ly/2wpSQ1H

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Phase III

- Add more hardware:
  - Accelerometer for more telemetry
- Improve autopilot for control

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Phase IV

- LIDAR
- Add object detection and avoidance
- Road sharing

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Not Planned

- GPS
- Scaling up

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Demo Video

Also viewable on YouTube at https://youtu.be/672G7CUpovM

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Special Thanks

- Lee Fox (@FoxInATX)
- Todd Higgins (@AnActOfTodd)
- Charles Grossman
- Justin Howe (@syntaxinvalid)
- Robo4J Team:
  - Marcus Hirt (@hirt)
  - Miro Wengner (@miragemiko)

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Resources

- Micronaut
  - http://micronaut.io/
- GalecinoCar Source
  - https://github.com/vanderfox/GalecinoCar
- Donkeycar
  - http://donkeycar.com
  - https://github.com/wroscoe/donkey
- Robo4J
  - http://www.robo4j.io
- DeepLearning4J
  - https://github.com/deeplearning4j
- Keras autopilot
  - https://wroscoe.github.io/keras-lane-following-autopilot.html
- Opensource.com GalecinoCar Article
  - https://opensource.com/article/18/6/galecino-car

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Image Credits

- Nvidia Car - Dean Takahashi
- CNN Diagram - Elite Data Science
- Cadillac Super Cruise - ExtremeTech
- Tesla - Tesla
- Galecino - Nature Mobile
- Nascar cartoon - Takealookatthis
- Robo4J - robo4j.io
- Hardware - Amazon and Donkeycar
- Tracks - Donkeycar

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Any questions?

- Ryan Vanderwerf
  - @RyanVanderwerf
  - rvanderwerf@gmail.com
- Lee Fox
  - @FoxInATX
  - lee.h.fox@gmail.com

Deck: https://bit.ly/2LVlgHS

Code: http://bit.ly/galecino-code

© 2018, Object Computing, Inc. (OCI). All rights reserved.