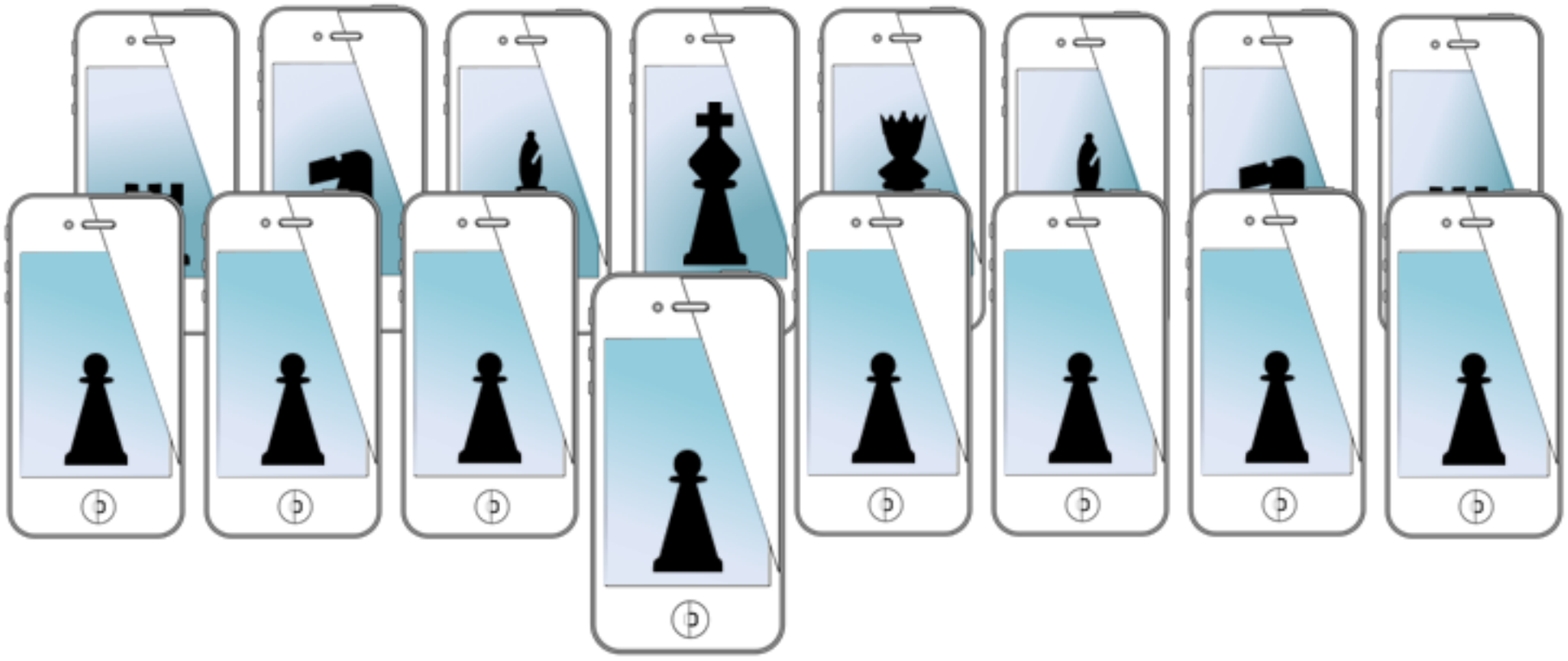
MOBILE SENSING LEARNING



CSE5323 & 7323

Mobile Sensing and Learning

CoreML

Eric C. Larson, Lyle School of Engineering,
Computer Science and Engineering, Southern Methodist University

Wanted:

Participants for a Research Study in Biometrics and Privacy

- SMU Researchers are seeking participants for an upcoming research study.
- We will be gathering data about your biometrics as you do a series of everyday tasks on a mobile phone and laptop.
- We ask for about **1.5-2 hours** of your time! For completing the session, you will receive a $10 Amazon gift card.
- Please contact **eesharp@smu.edu**

IRB Approved: 02/23/2018 SL
Expires: 02/23/2019
Study ID: H18-020-LARE

Contact
eesharp@smu.edu

Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu

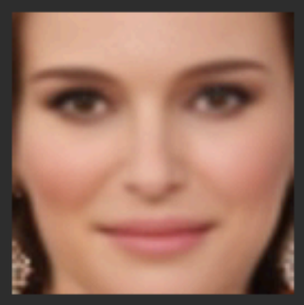
eesharp@smu.edu

http://faceface.lyle.smu.edu/

SMU FaceFace

VOTE RANKING ABOUT

Which of the following two faces looks MORE FAKE to you?

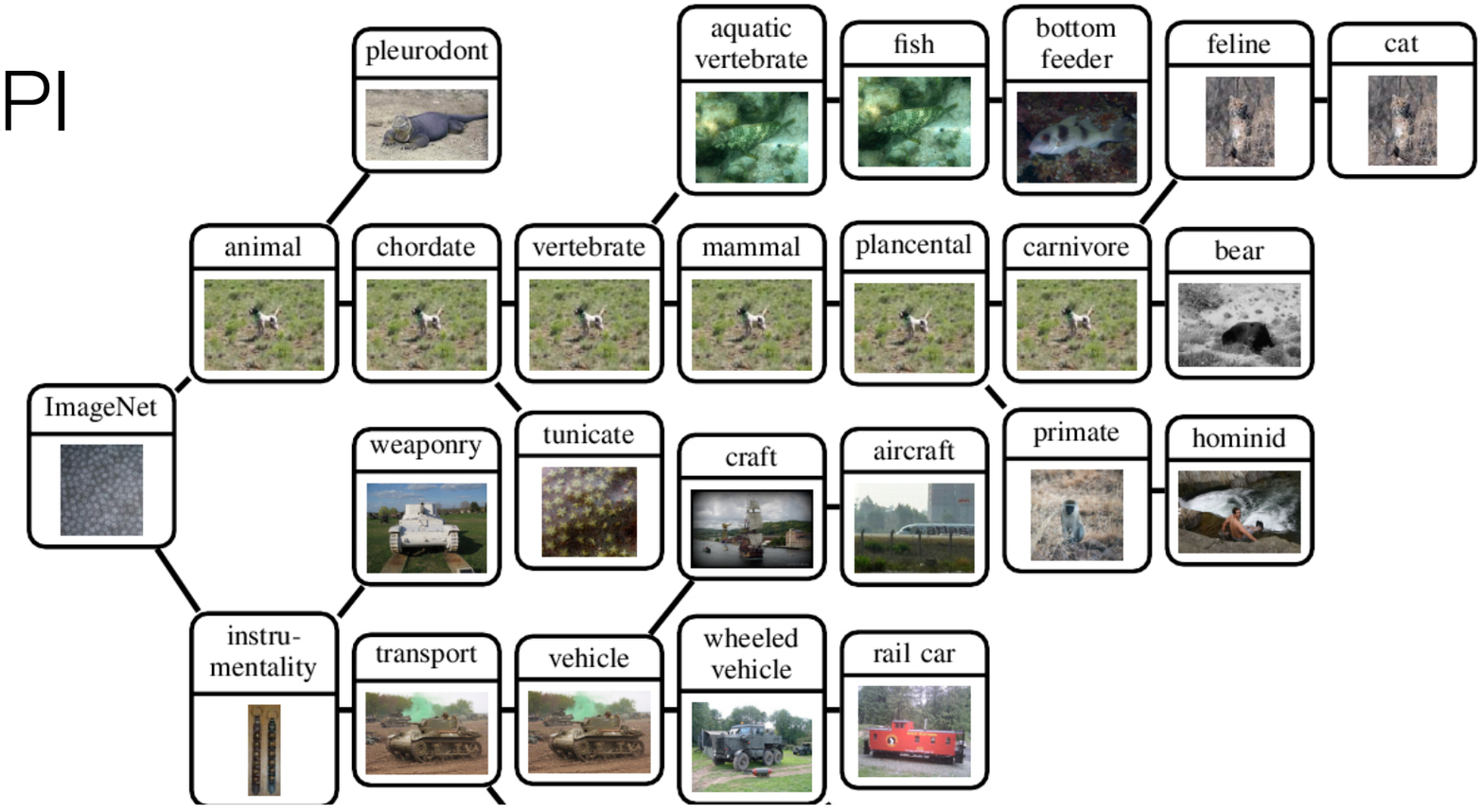


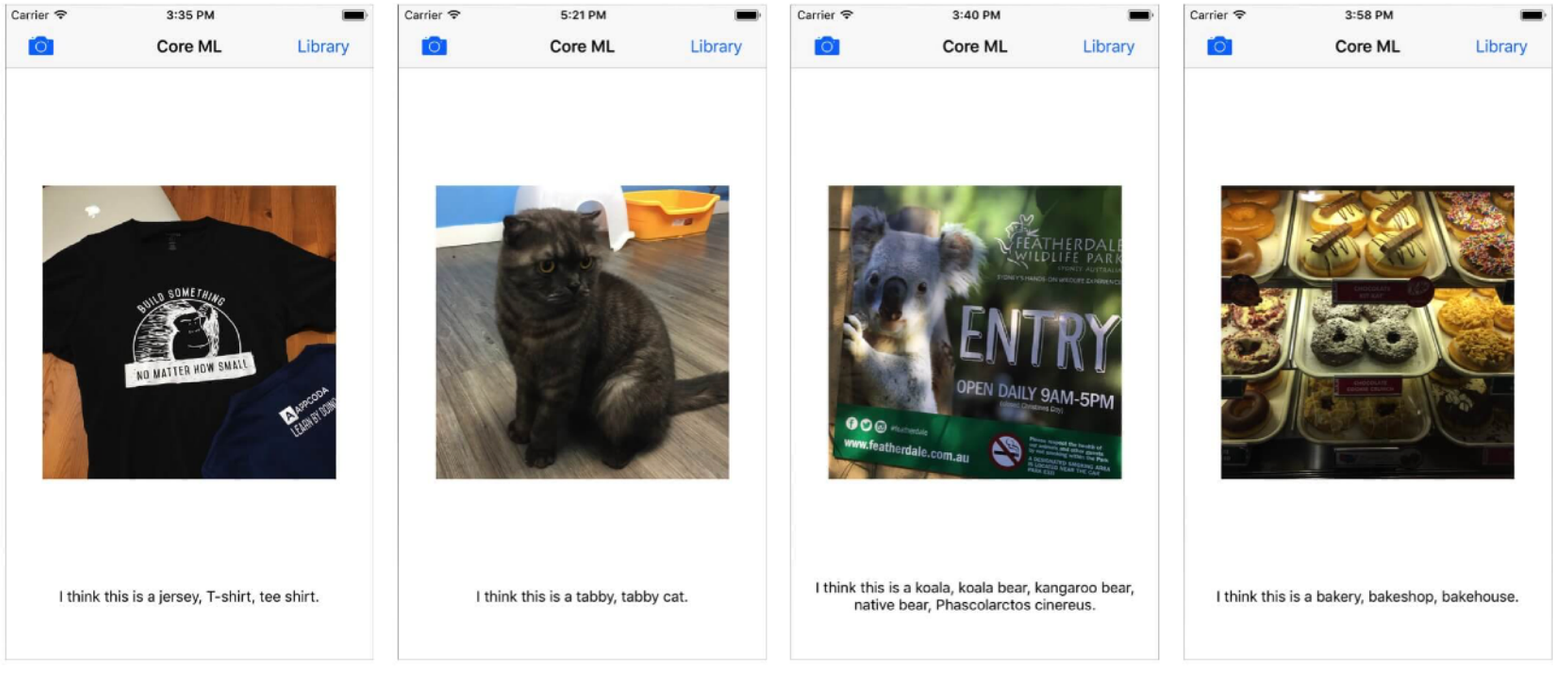
course logistics

- **A6** is due over the weekend
  - you can run server from your laptop (or cloud)
  - three lectures remaining!
- **final project proposal** is also due at same time
  - submit draft with description and deliverables

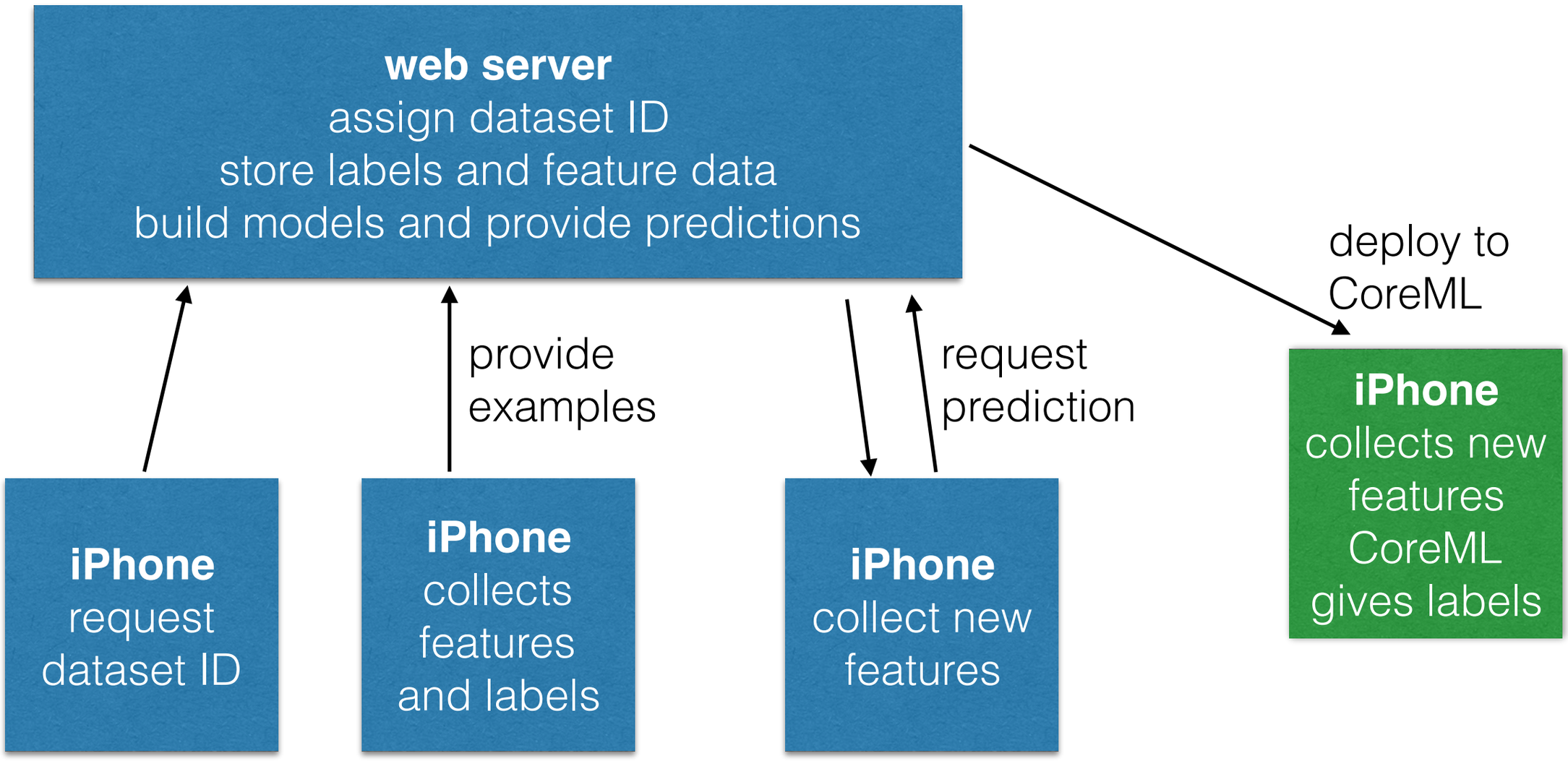
Last Time: Core Vision

- demo using vision API





assignment 6 demo

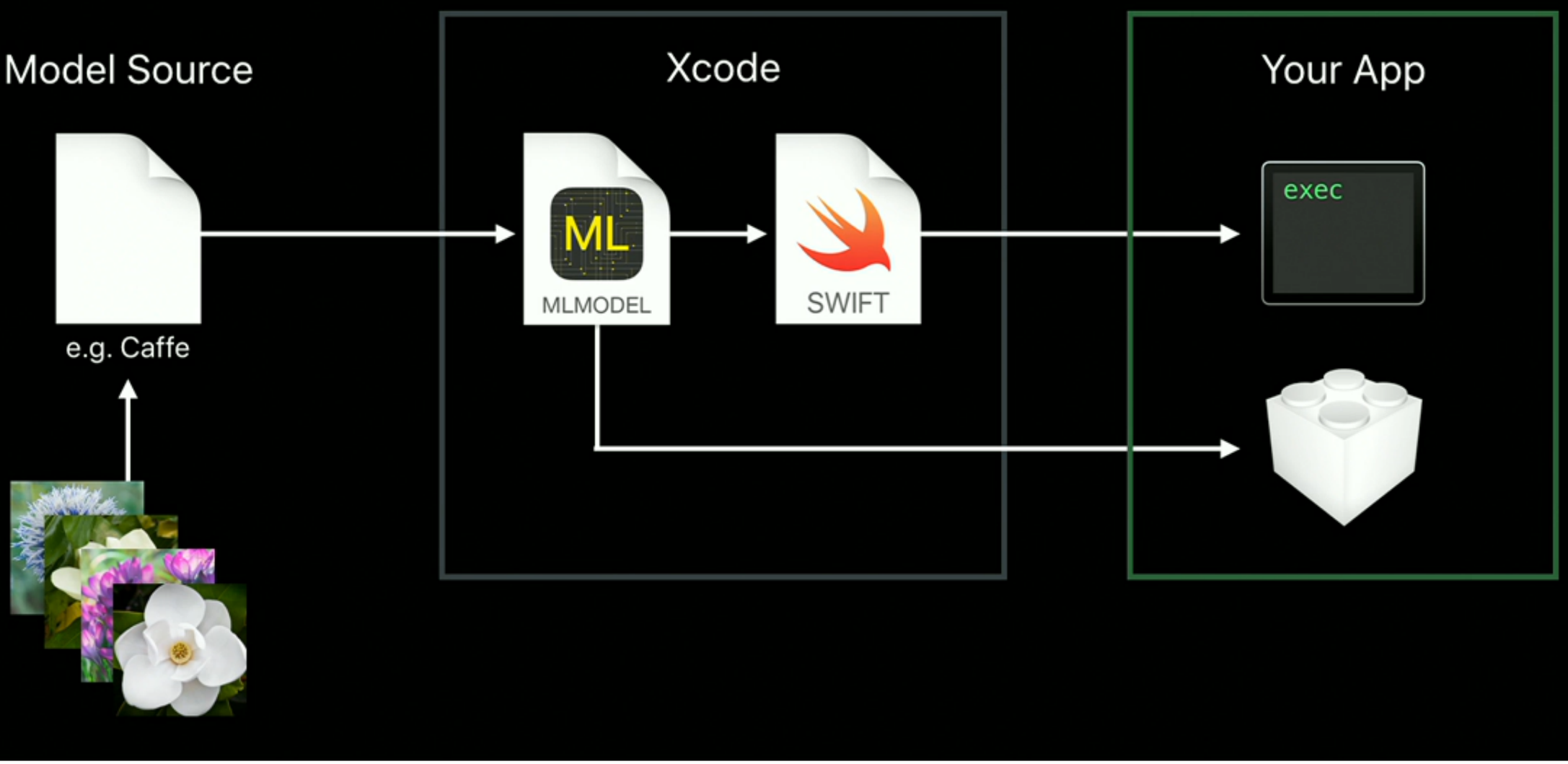


all uploads handled via JSON formatted HTTP requests

most of code is already posted

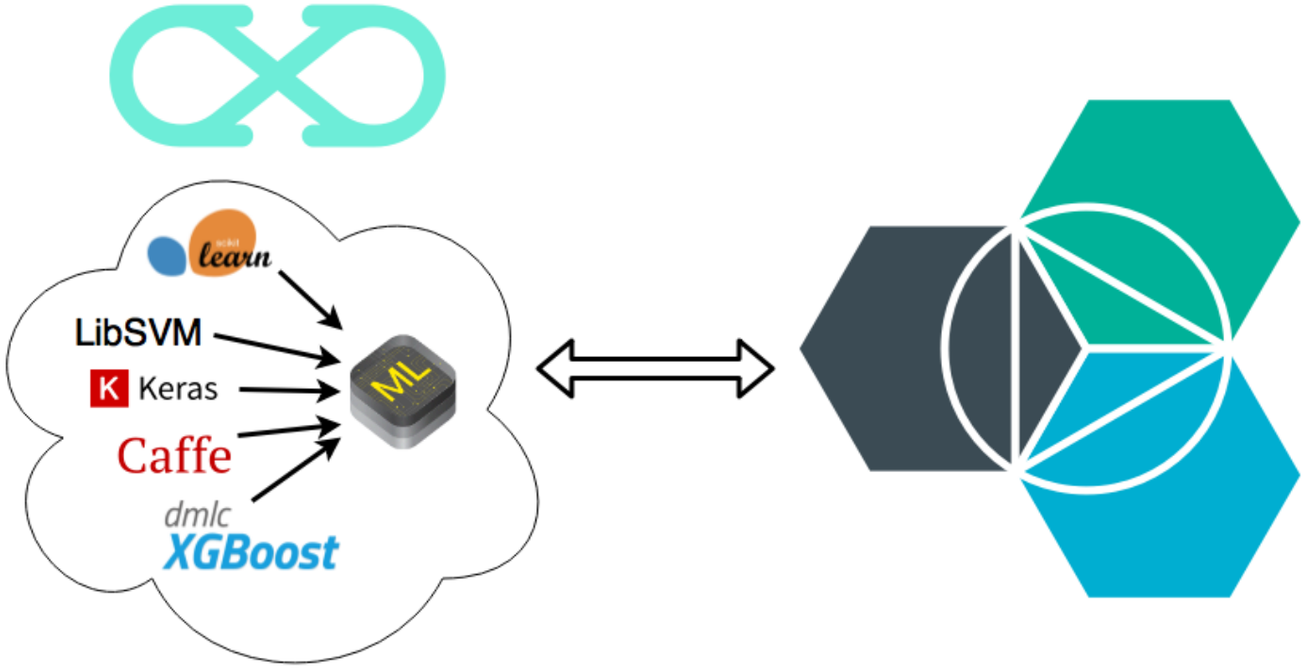
CoreML

Conversion Workflow



CoreML

getting trained models:



https://developer.apple.com/machine-learning/

Working with Models

Build your apps with the ready-to-use Core ML models below, or use Core ML Tools to easily convert models into the Core ML format.

Models

MobileNet

MobileNets are based on a streamlined architecture that have depth-wise separable convolutions to build lightweight, deep neural networks.

Detects the dominant objects present in an image from a set of 1000 categories such as trees, animals, food, vehicles, people, and more.

View original model details ›

Download Core ML Model

File size: 17.1 MB

but… we want more than **pre-trained** models

Installing CoreMLTools

- Currently supports python 2 and 3
- so create a conda environment and install coremltools
  - `conda install sklearn numpy (+others) …`
  - `pip install coremltools`

```
clf = RandomForestClassifier(n_estimators=50)
print("Training Model", clf)

clf.fit(X,y)

print("Exporting to CoreML")

coreml_model = coremltools.converters.sklearn.convert(
    clf,
    ["accelX"]*50+["accelY"]*50+["accelZ"]*50, # feature names (optional)
    "Direction") # label name (optional)

# save out as a file
coreml_model.save('rf.mlmodel')
```

Using CoreML

- drag into project

Machine Learning Model

Name	RandomForestAccel
Type	Tree Ensemble Classifier
Size	28 KB
Author	unknown
Description	description not included
License	unknown

Model Class

RandomForestAccel

Automatically generated Swift model class

Model Evaluation Parameters

Name	Type	Desc
inputs		
input	MultiArray (Double 150)	
outputs		
classLabel	String	
classProbability	Dictionary (String → Double)	

```
/// Class for model loading and prediction
@available(macOS 10.13, iOS 11.0, tvOS 11.0, watchOS 4.0, *)
class RandomForestAccel {
    var model: MLModel

    /**
        Construct a model with explicit path to mlmodel file
        - parameters:
           - url: the file url of the model
           - throws: an NSError object that describes the problem
    */
    init(contentsOf url: URL) throws {
        self.model = try MLModel(contentsOf: url)
    }

    /// Construct a model that automatically loads the model from th
    convenience init() {
        let bundle = Bundle(for: RandomForestAccel.self)
        let assetPath = bundle.url(forResource: "RandomForestAccel",
        try! self.init(contentsOf: assetPath!)
    }

    /**
        Make a prediction using the structured interface
        - parameters:
           - input: the input to the prediction as RandomForestAccel
        - throws: an NSError object that describes the problem
        - returns: the result of the prediction as RandomForestAccel
    */
    func prediction(input: RandomForestAccelInput) throws -> RandomF
```

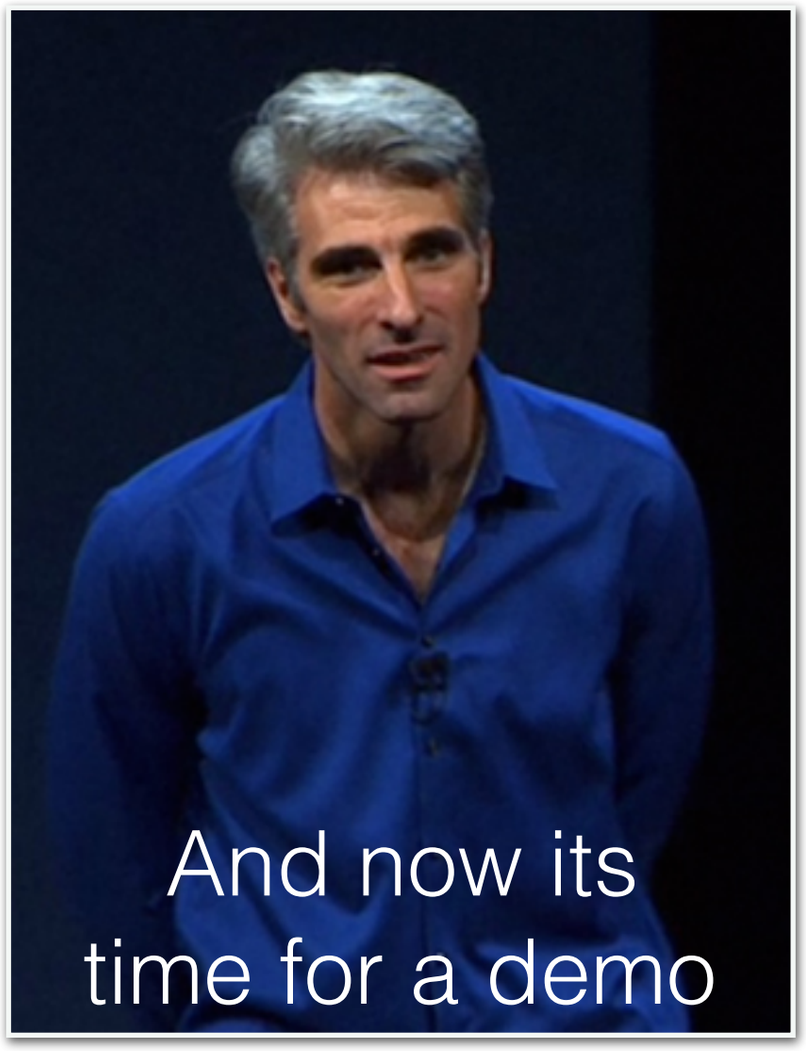
Using CoreML

- drag into project
- use like previously in HTTP model

```
var model = RandomForestAccel()

//predict a label
let seq = toMLMultiArray(self.ringBuffer.getDataAsVector())
guard let output = try? model.prediction(input: seq) else {
    fatalError("Unexpected runtime error.")
}

displayLabelResponse(output.classLabel)
```

CoreML

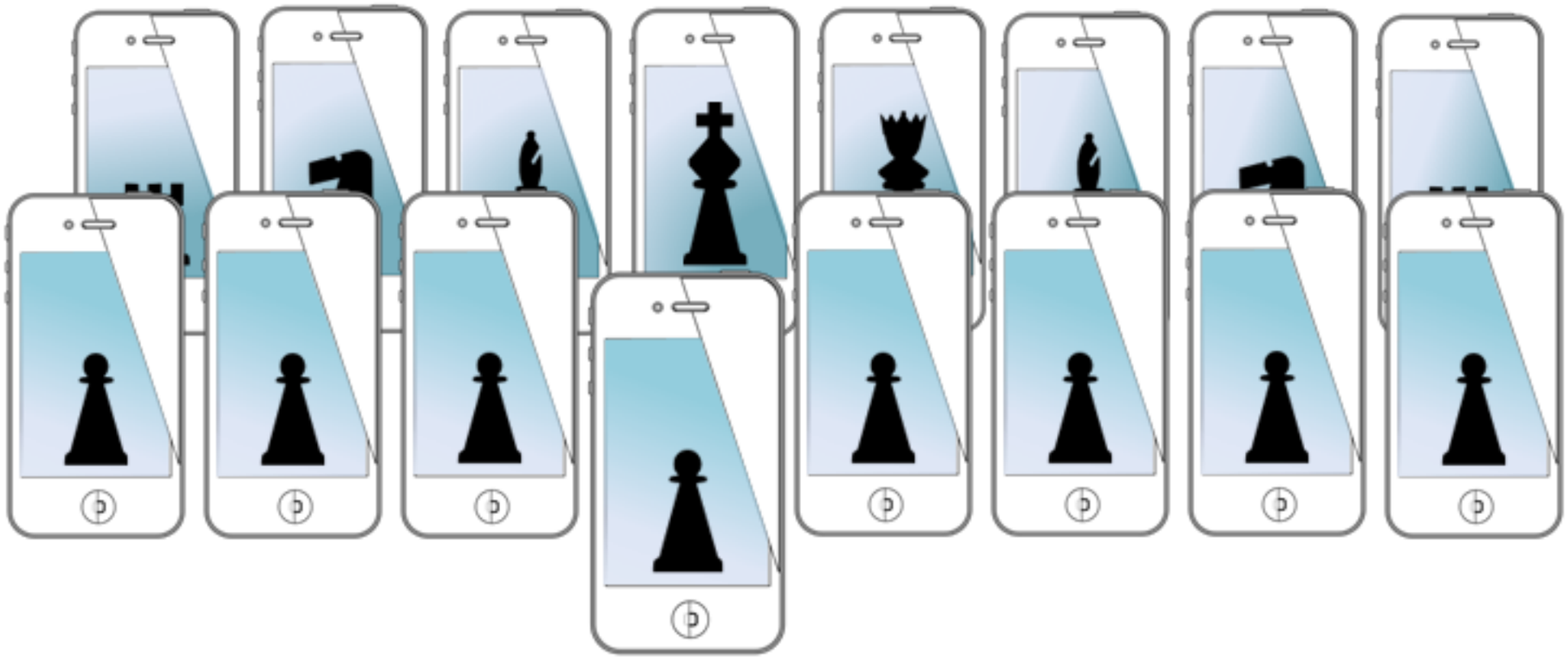
- extended demo from beginning to end
- let's add support for another (user selectable) sklearn classifer…



for next time…

- ARKit from 1000 feet
- Pitching
- ~Fin~

MOBILE SENSING LEARNING



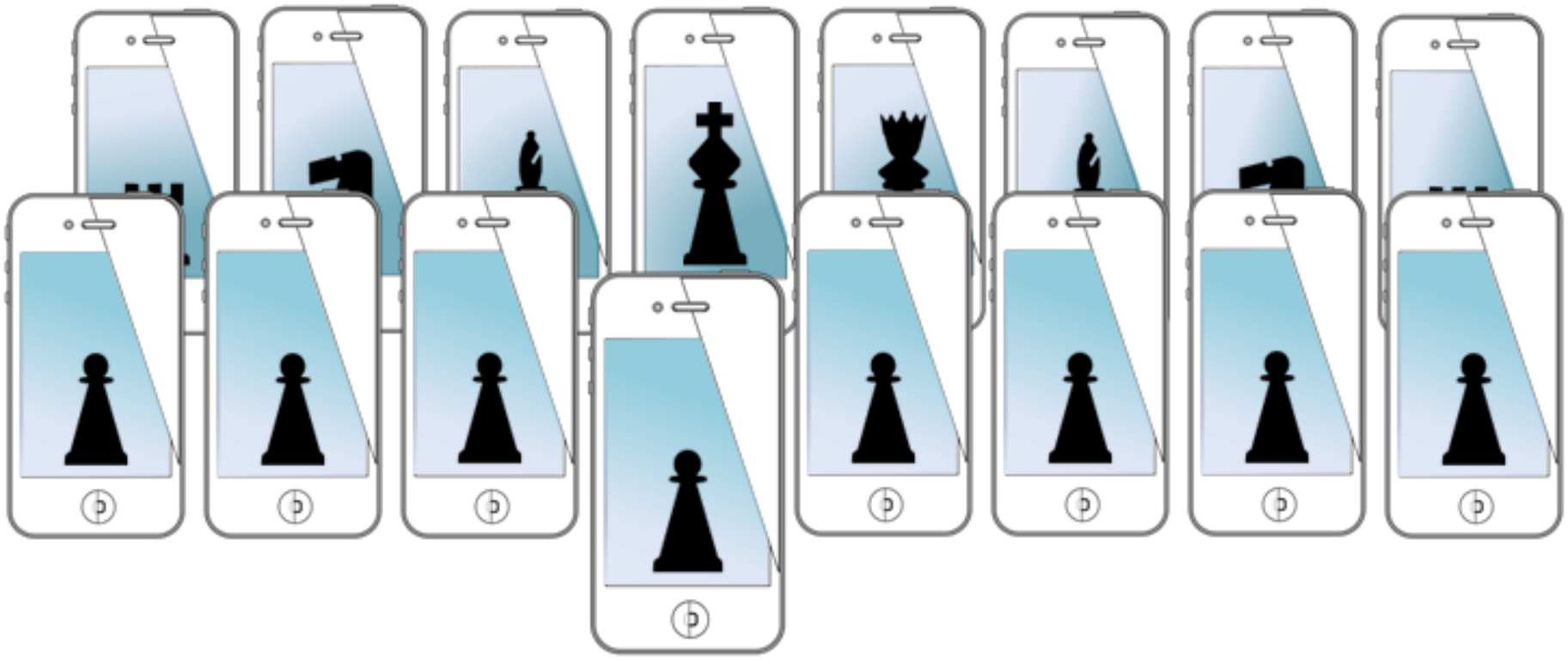
CSE5323 & 7323

Mobile Sensing and Learning

CoreML

Eric C. Larson, Lyle School of Engineering,
Computer Science and Engineering, Southern Methodist University

MOBILE SENSING LEARNING



CSE5323 & 7323

Mobile Sensing and Learning

ARKit and SceneKit

Eric C. Larson, Lyle School of Engineering,
Computer Science and Engineering, Southern Methodist University

Wanted:

Participants for a Research Study in Biometrics and Privacy

- SMU Researchers are seeking participants for an upcoming research study.
- We will be gathering data about your biometrics as you do a series of everyday tasks on a mobile phone and laptop.
- We ask for about **1.5-2 hours** of your time! For completing the session, you will receive a $10 Amazon gift card.
- Please contact **eesharp@smu.edu**

IRB Approved: 02/23/2018 SL
Expires: 02/23/2019
Study ID: H18-020-LARE

Contact
eesharp@smu.edu

Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu
Contact
eesharp@smu.edu

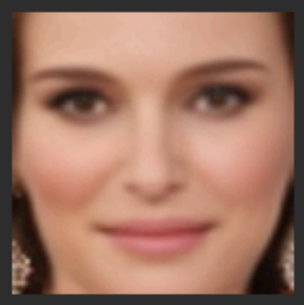
eesharp@smu.edu

http://faceface.lyle.smu.edu/

SMU FaceFace

VOTE RANKING ABOUT

Which of the following two faces looks MORE FAKE to you?



course logistics

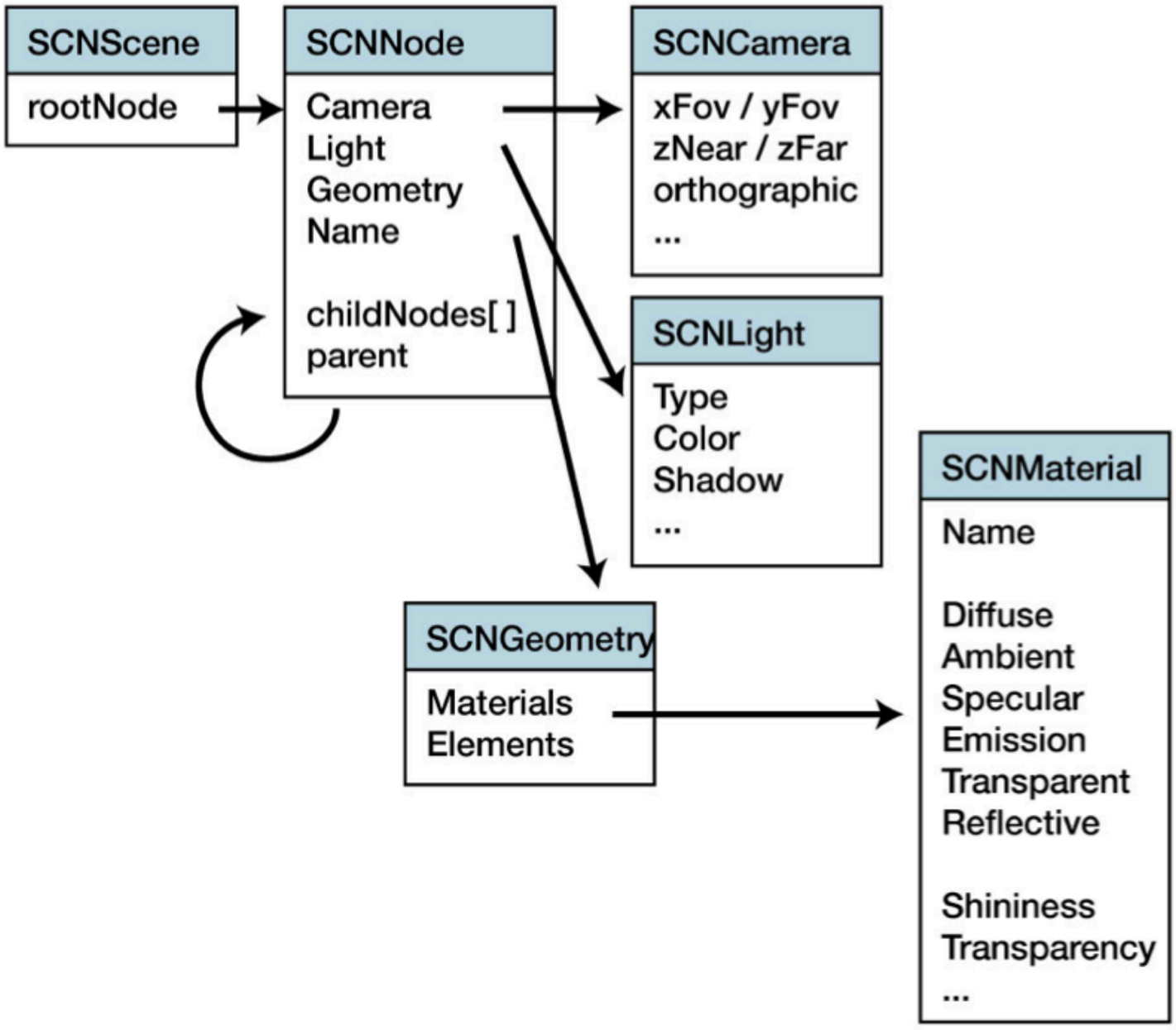
- (editied) project proposals posted
- final project should be the main concern right now
  - app / presentation / video
- No class next time, will meet after Thanksgiving
- agenda:
  - SceneKit
  - ARKit
  - demo

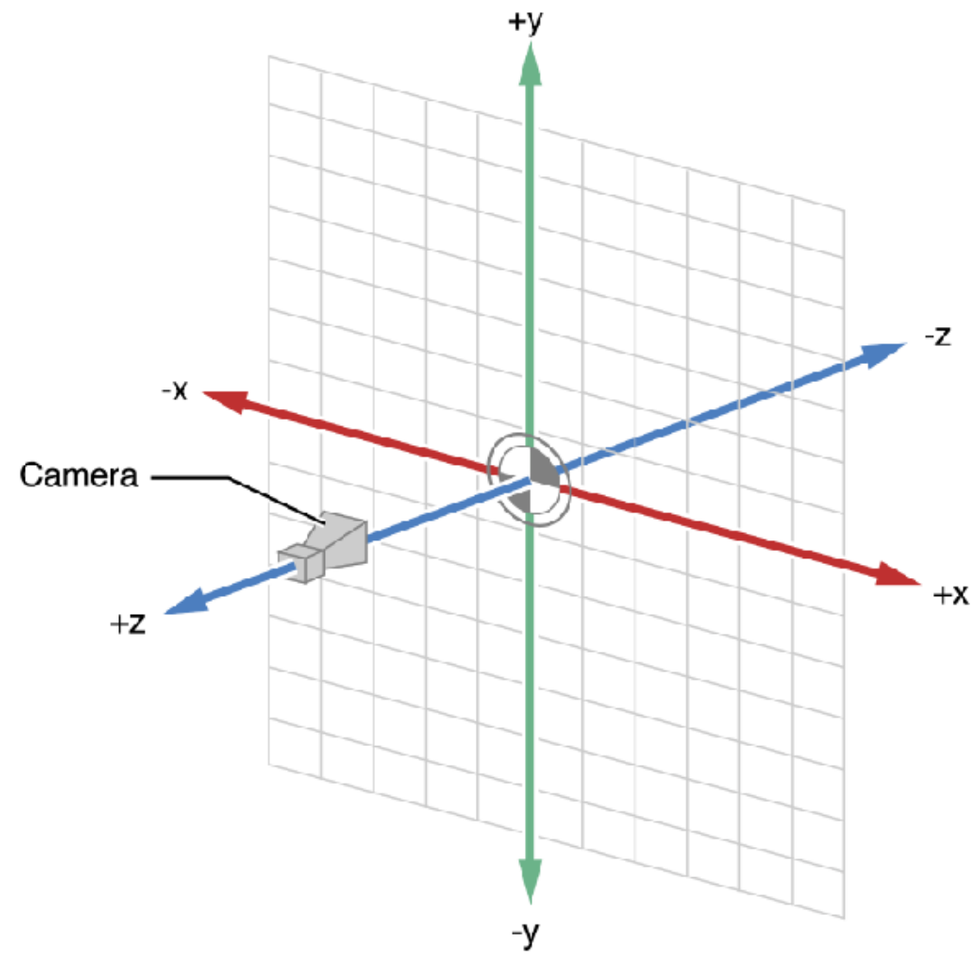
https://developer.apple.com/videos/play/wwdc2017/602/

SceneKit: 3D scenes

- need some basic knowledge to understand augmented reality and digital scene
- SceneKit allows you to create a 3D world and add physics, nodes, lighting, etc.
  - very powerful
- basic workflow:
  - setup world
  - add nodes

work flow in 3D scenes

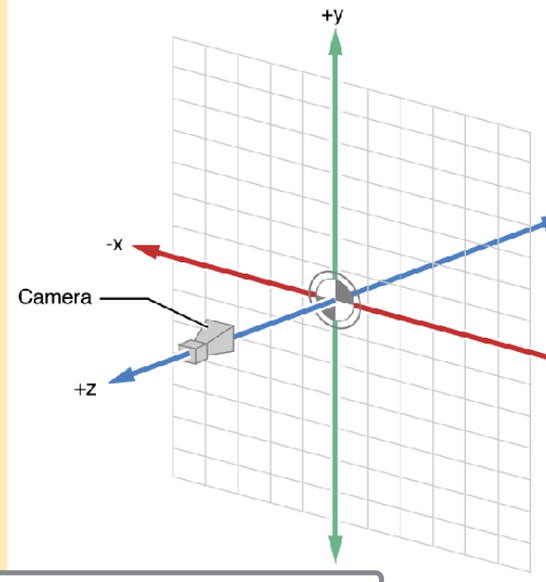




setting up a world

```
// Setup scene
scene = SCNScene()
scene.physicsWorld.speed = 1

// Setup camera position
cameraNode = SCNNode()
cameraNode.camera = SCNCamera()
cameraNode.position = SCNVector3(x: 0, y: 0, z: 30)
scene.rootNode.addChildNode(cameraNode)

// add a plane to the view that users must bounce the ball on
//setup the geometry of node (as a plane)
let wall = SCNPlane(width: 10.0, height: 10.0)
wall.firstMaterial?.doubleSided = true
wall.firstMaterial?.diffuse.contents = UIColor.redColor() // make it red!!

// add the plane to the world as a static body (no dynamic physics)
wallNode = SCNNode()
wallNode.geometry = wall
wallNode.physicsBody = SCNPhysicsBody.staticBody()
wallNode.position = SCNVector3(x: 0.0, y: 0.0, z: -5)

scene.rootNode.addChildNode(wallNode)

// Setup view
let view = self.view as SCNView
view.scene = scene
```

add camera



add plane

make this scene the world

add to world

```
func addBall() {

        // add a sphere to the world
        let ballGeometry = SCNSphere(radius: 1.0)

        // make it have texture
        let ballMaterial = SCNMaterial()
        ballMaterial.diffuse.contents = UIImage(named: "texture")

        // adjust physics to make it slightly highly boun
        let ball = SCNNode(geometry: ballGeometry)
        ball.geometry?.firstMaterial = ballMaterial;
        ball.physicsBody = SCNPhysicsBody.dynamicBody()
        ball.physicsBody?.restitution = 2.5
        ball.position = SCNVector3(x: 0, y: 0, z: 0)

        scene.rootNode.addChildNode(ball)

}
```

make ball

add physics

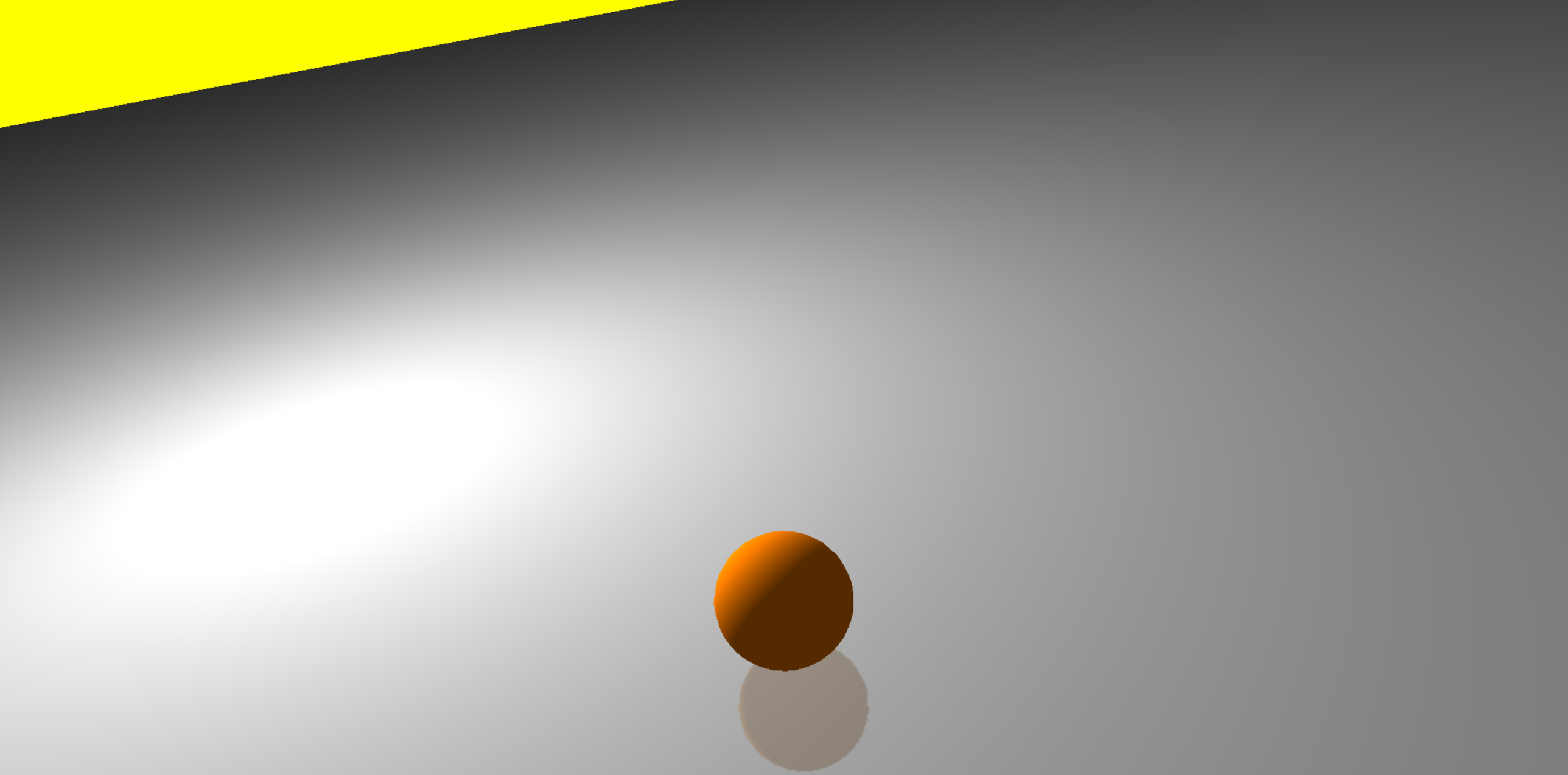
make bouncy

add to world

physics in world

```
motionManager.startDeviceMotionUpdatesToQueue(
   NSOperationQueue.currentQueue())
        {
            (deviceMotion, error) -> Void in

            let accel = deviceMotion.gravity
            let userAccel = deviceMotion.userAcceleration

            let accelX = Float(9.8 * accel.x + userAccel.x*9.8)
            let accelY = Float(9.8 * accel.y + userAccel.y*9.8)
            let accelZ = Float(9.8 * accel.z + userAccel.z*9.8)

            self.scene.physicsWorld.gravity =
                SCNVector3(x: accelX, y: accelY, z: accelZ)

        }
```

similar to SpriteKit
but in three dimensions!!



but… we want the SCNScene

to be the real world

```
// Setup view
let view = self.view as SCNView
view.scene = scene
```

```
// Setup view
let view = self.view as ARSCNView
view.scene = scene
```

ARKit basics

Tracking

World tracking

Visual inertial odometry

No external setup

Rendering

Easy integration

AR views

Custom rendering

Scene Understanding

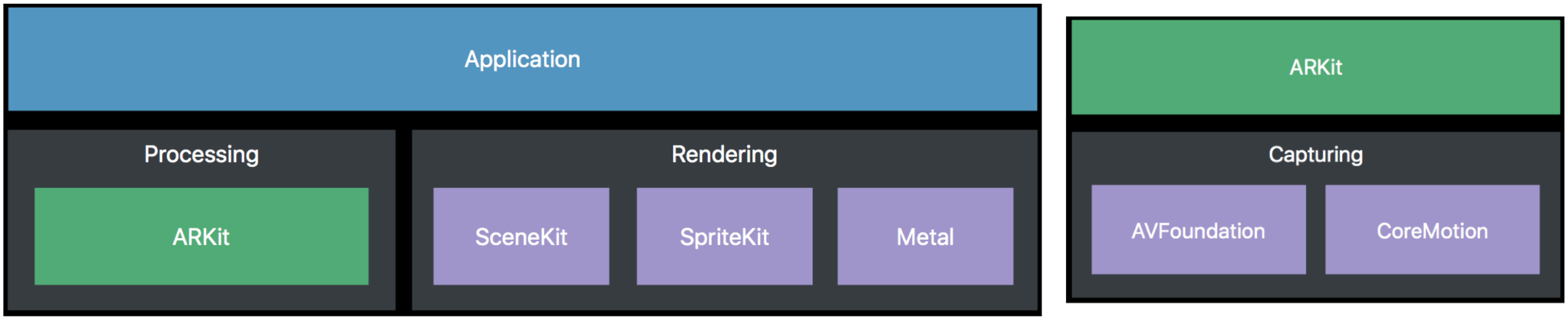
Plane detection

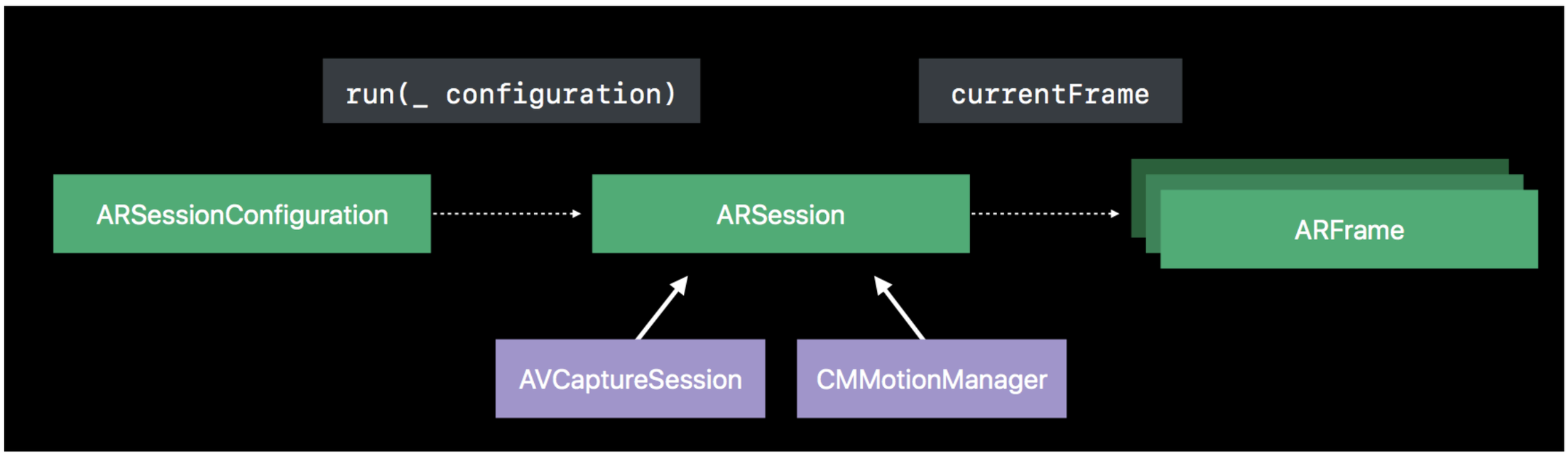
Hit-testing

Light estimation

https://developer.apple.com/videos/play/wwdc2017/602/

ARKit system integration





https://developer.apple.com/videos/play/wwdc2017/602/

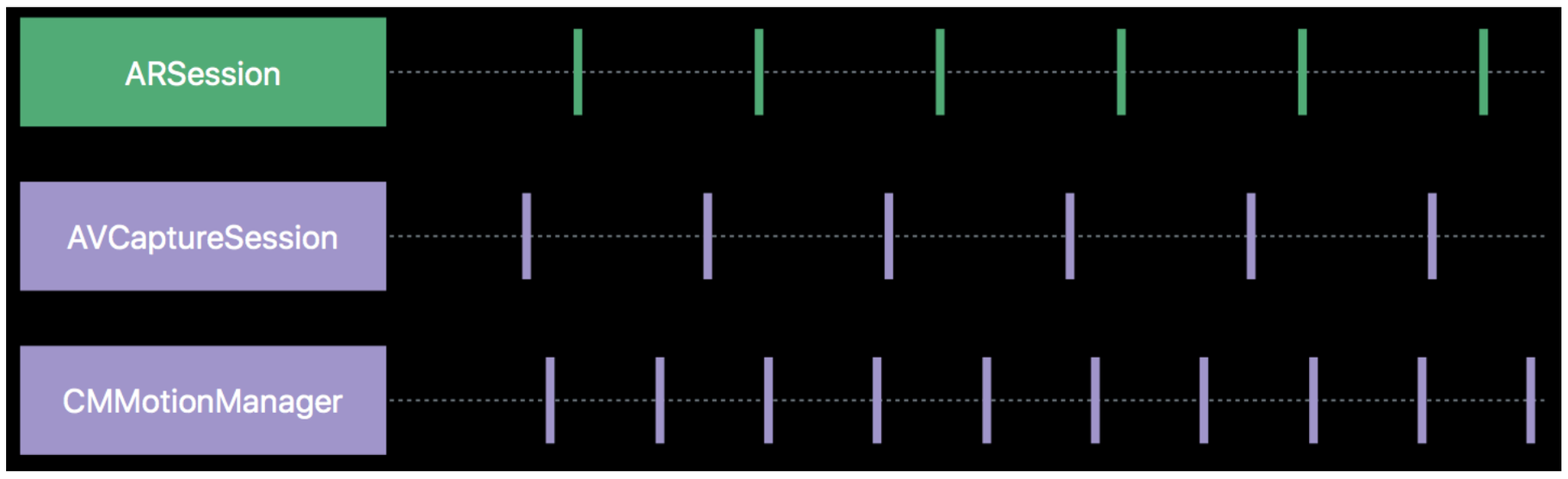
session basics

```swift
let session = ARSession()
let conf = ARWorldTrackingSessionConfiguration()
session.run(conf)

// Run your session
session.run(configuration)

// Pause your session
session.pause()

// Resume your session
session.run(session.configuration)

// Change your configuration
session.run(otherConfiguration)
```

session basics

```
// Access the latest frame
func session(_: ARSession, didUpdate: ARFrame)

// Handle session errors
func session(_: ARSession, didFailWithError: Error)
```

delegation



unified sampling

https://developer.apple.com/videos/play/wwdc2017/602/

adding to the AR world

poll ARSession

```
// grab the current AR session frame from the scene, if possible
guard let currentFrame = sceneView.session.currentFrame else {
    return
}

// setup some geometry for a simple plane
let imagePlane = SCNPlane(width:sceneView.bounds.width/6000,
                          height:sceneView.bounds.height/6000)
```

create a plane

```
// add the node to the scene
let planeNode = SCNNode(geometry:imagePlane)
sceneView.scene.rootNode.addChildNode(planeNode)

// update the node to be a bit in front of the camera inside the AR session


// step one create a translation transform
var translation = matrix_identity_float4x4
translation.columns.3.z = -0.1

// step two, apply translation relative to camera for the node
planeNode.simdTransform = matrix_multiply(currentFrame.camera.transform, translation )
```

add to world

translate

operations

Figure 1-8 Matrix configurations for common transformations

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
Identity

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix}$$
Translate

$$\begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
Scale

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
Rotate around X axis

$$\begin{bmatrix} \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
Rotate around Y axis

$$\begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
Rotate around Z axis

```
// step one create a translation transform
var translation = matrix_identity_float4x4
translation.columns.3.z = -0.1

// step two, apply translation relative to camera for th
planeNode.simdTransform = matrix_multiply(currentFrame.c
```

SIMD: single instruction, multiple data

Creating Transform Matrices

`func SCNMatrix4MakeTranslation(Float, Float, Float)`

Returns a matrix describing a translation transformation.

`func SCNMatrix4MakeRotation(Float, Float, Float, Float)`

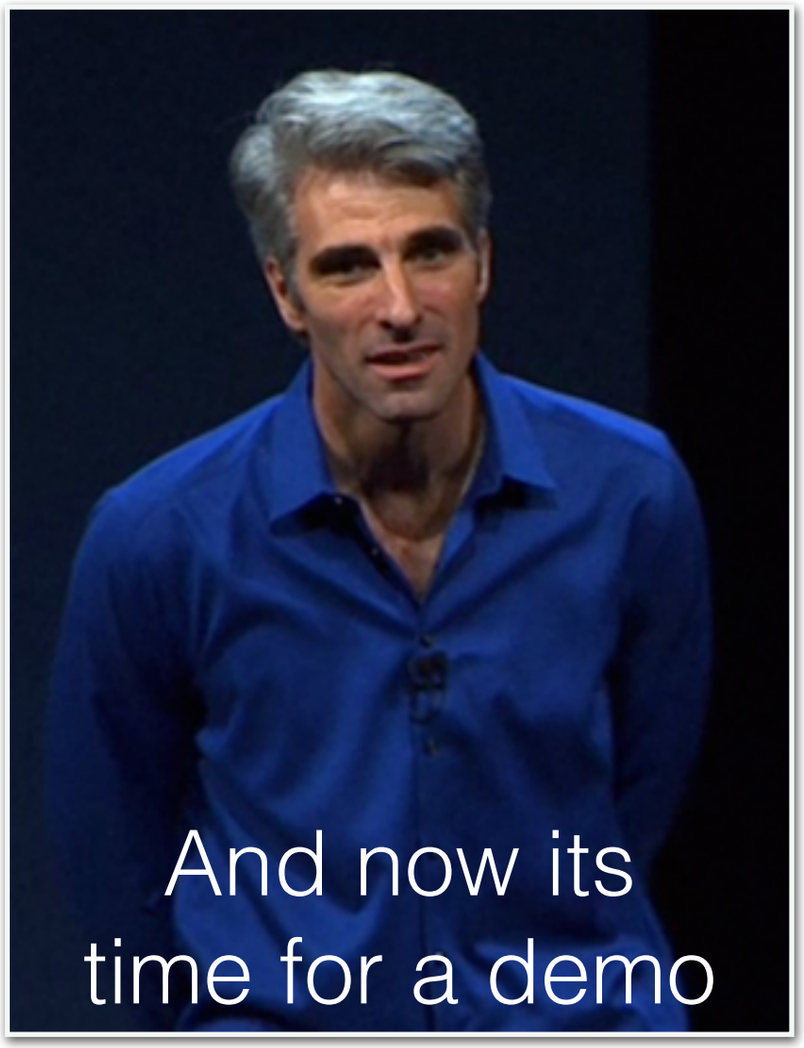
Returns a matrix describing a rotation transformation.

`func SCNMatrix4MakeScale(Float, Float, Float)`

Returns a matrix describing a scale transformation.

ARKit and SceneKit

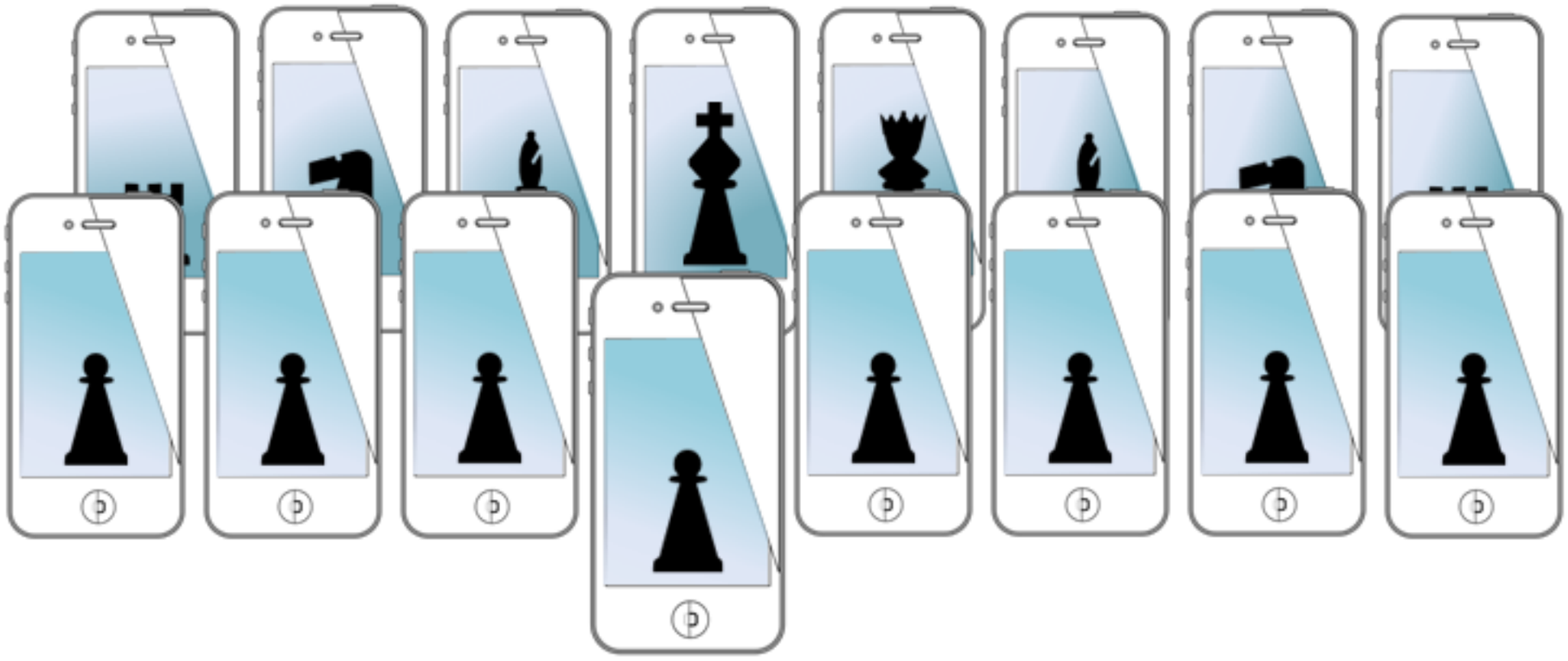
- extended demo
- close with final projects



for next time…

- no lecture next time
- Pitching
- ~Fin~

MOBILE SENSING LEARNING



CSE5323 & 7323

Mobile Sensing and Learning

ARKit and SceneKit

Eric C. Larson, Lyle School of Engineering,
Computer Science and Engineering, Southern Methodist University