MOBILE APPLICATIONS AND CLOUD COMPUTING

INTRODUCTION

OUTLINE

- Logistic info
- course presentation (syllabus)
- About the exam

LOGISTIC INFO

- Friday lessons may start 15 min delays
- Material is published on the web site, just before the lectures
- Office hours: Friday @ 3 pm
- Smartphone not allowed during lessons (for chatting etc.)

SYLLABUS

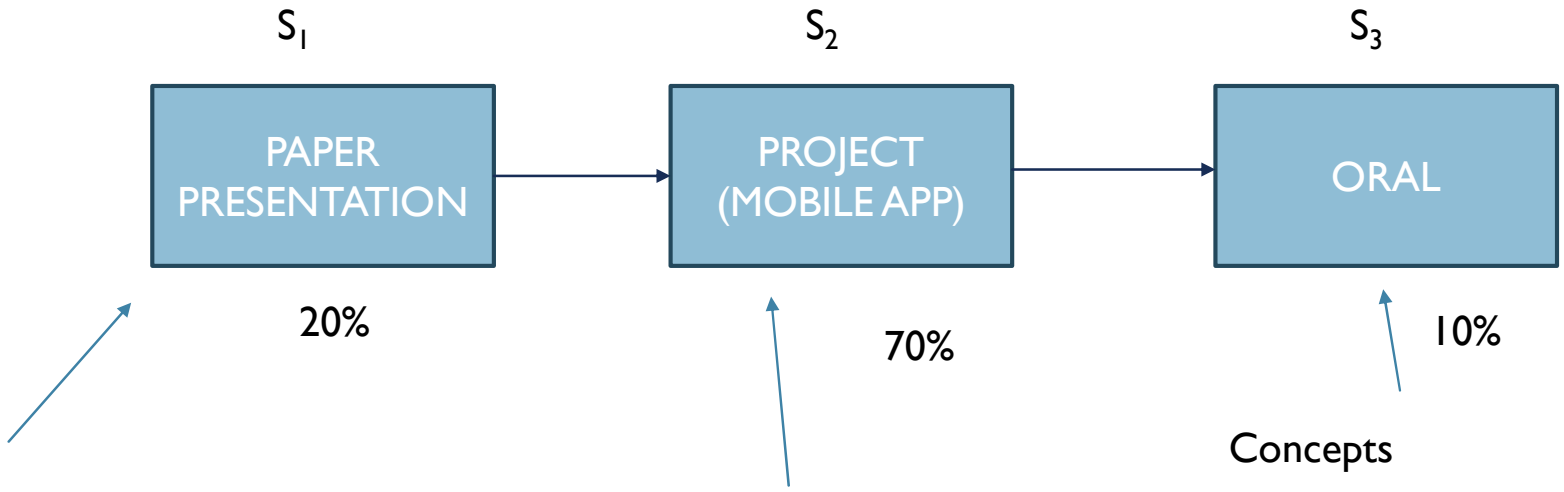
Cloud computing:

- **Evolution of computing concept**
- **Cloud computing models**
- **Examples of cloud services and providers**

Mobile applications (Android)

- **Core components: Activity, Broadcast Receiver, Service, Content provider**
- **Modern Android Development (MAD)**
- **Sensors and orientation**
- **2D graphics basics**

ABOUT THE EXAM



Choice a scientific paper and prepare a 10 min presentation of it

Solo or in group (2, 3)
The code of the project should be available on GitHub (or another equivalent site) and registered before the exam

$S = 0.2 \times S_1 + 0.7 \times S_2 + 0.1\ S_3$ (Provided that $S_i >= 18$)

CODING IN THE AI ERA

- **Coding ≠ Programming**

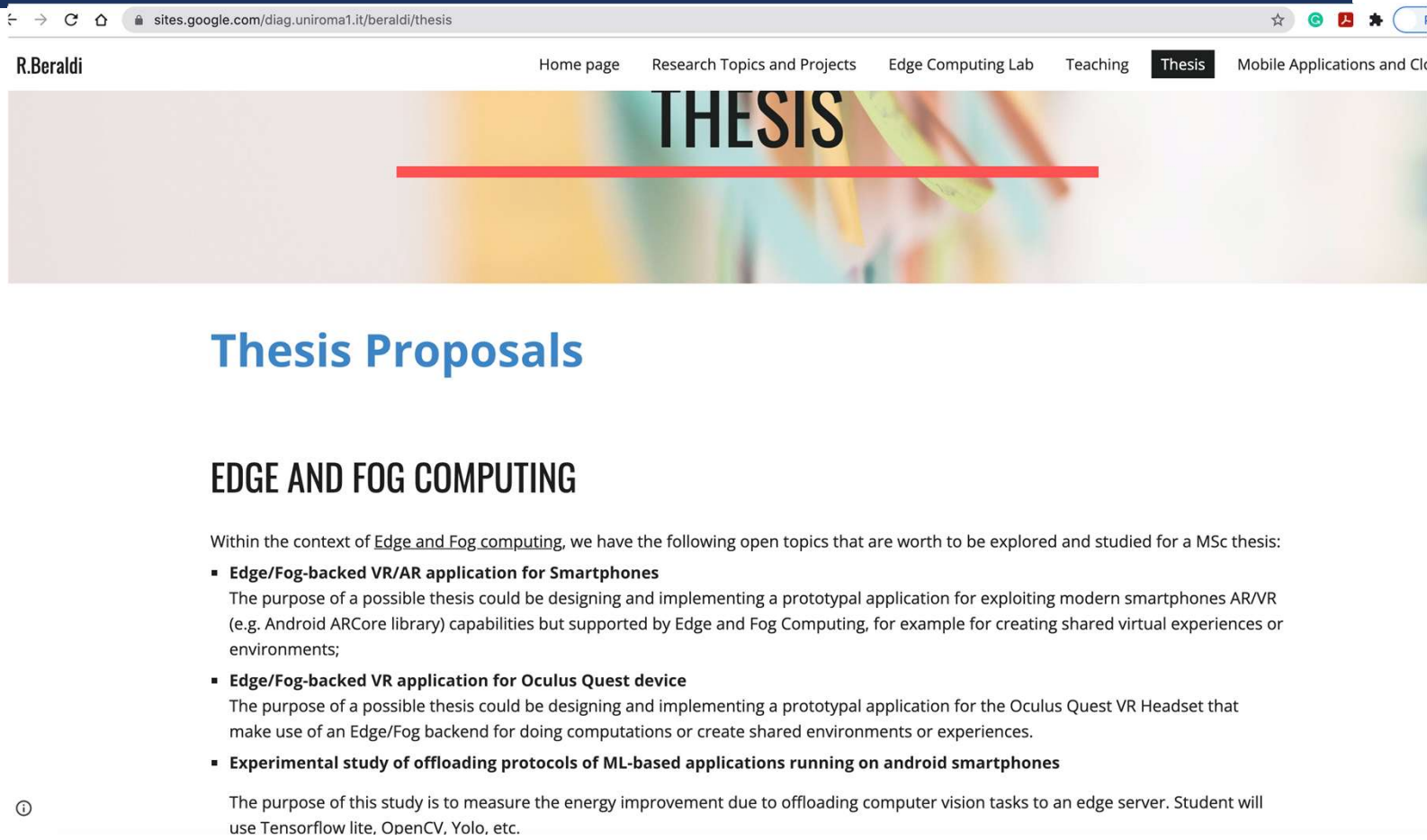
- AI helps for Faster Code Generation, Improved Debugging & Troubleshooting
- AI tools can automate code writing and suggest optimized solutions, reducing development time and minimizing repetitive tasks, analyze errors, recommend fixes, etc.
  - Github copilot, Gemini in android studio
  - Cloud assistant
- But…

CODING AND PROGRAMMING

- Programmer has the responsibility of the final outcome
- Focus shifted on higher (more abstracted) tasks like
  - designing system architecture (more focused on what the system should do rather than coding),
  - security concerns
  - optimizing performance
  - Integration (testing, etc.)
  - …



TURN YOUR PROJECT INTO THE FINAL THESIS…



MOBILE WEB SITE

- Web site for mobile devices are not considered mobile apps (they are not installed and do not have access to all sensors and data) although the web technology can be used to develop the UI part of a mobile app

MORE ABOUT YOUR PROJECT IN 10 POINTS

1. Use Cloud services
2. Multi-users
3. Graphics
4. Sensors
5. Camera/Image
6. Concurrency

Mobile application

Cloud provider

7. Authentication
8. Others

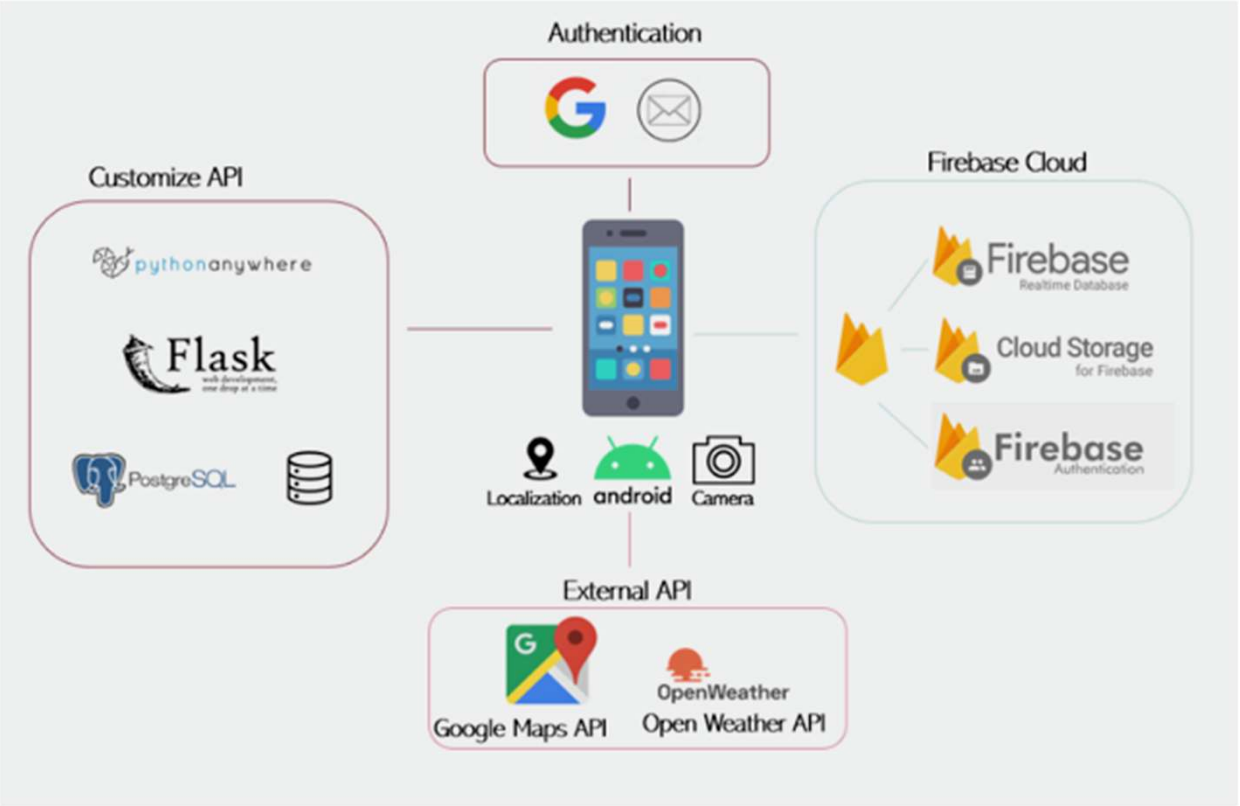
Cloud service from some provider (e.g. Firebase,..)

Create your API service

9. CRUD interface
10. Data storage

Your service running
on a PaaS, or IaaS (e.g., a Docker container)

EXAMPLE


Authentication
Customize API
pythonanywhere
Flask
PostgreSQL
Firebase Cloud
Firebase
Realtime Database
Cloud Storage
for Firebase
Firebase
Authentication
Localization
android
Camera
External API
Google Maps API
OpenWeather
Open Weather API

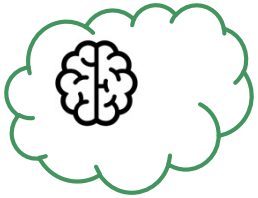
SOME IDEA

Take a photo of a chess board

Detection on board

Send the photo

Detection on cloud



Recognize board and create a digital description of the board (Portable Game Notation)

```
[Event "Interpolis International Tournament"]
[Site "Tilburg NED"]
[Date "1994.09.10"]
[Round "1.1"]
[White "Seirawan, Yasser"]
[Black "Smyslov, Vassily"]
[Result "0-1"]

1. d4 Nf6 2. c4 e6 3. Nf3 Bb4+ 4. Nbd2 c5 5. a3 Bxd2+ 6. Bxd2 d6 7. dxc5 dxc5
8. Qc2 Nbd7 9. O-O-O Qc7 10. g3 Ng4 11. Bf4 e5 12. Bh3 h5 13. Bxg4 hxg4 14.
Nxe5 Nxe5 15. Rd5 Rh5 16. Rxe5+ Rxe5 17. Qh7 f6 18. Rd1 Kf7 19. Qh8 Qc6 20. Rd8
Re8 21. Qh5+ Ke7 22. Rd6 Qe4 23. Rd3 Be6 24. Bd6+ Kd8 25. Be5+ Bd7 26. Qf7 Qf5
27. Bf4 g5 28. Be3 Qe6 29. Qg7 b6 30. Rd5 Qe7 31. Rxg5 Qxg7 32. Rxg7 Kc7 33. b3
Rxe3 34. fxe3 Rh8 35. e4 Rxh2 36. Kd2 Rg2 37. Rf7 Rxg3 38. e3 Rf3 39. Ke2
{On time} 0-1
```

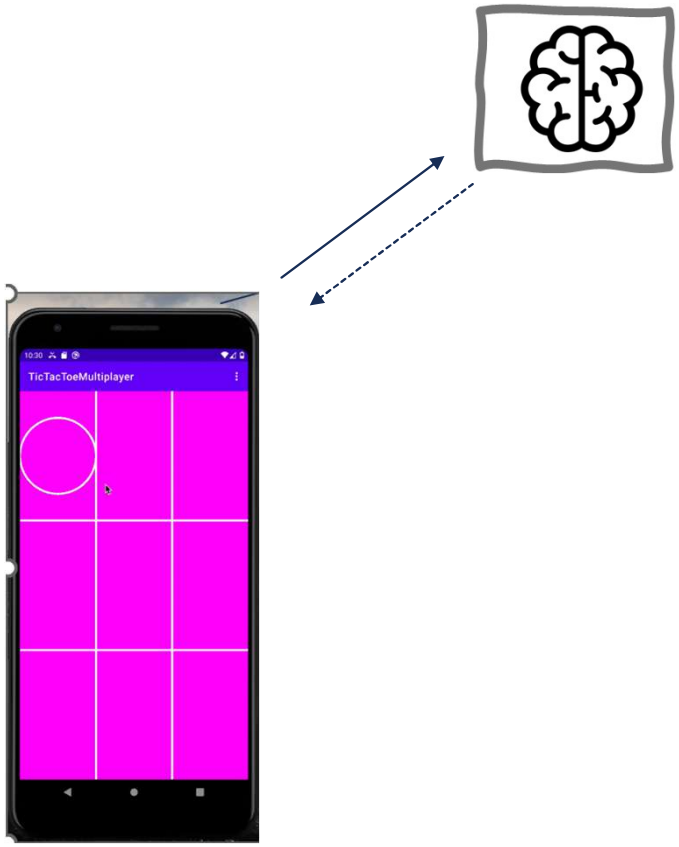
Play the game on the smartphone

Use external chess engine as opponent

SOME IDEA

In this example, two players share a board for the TTT

SOME VARIATION

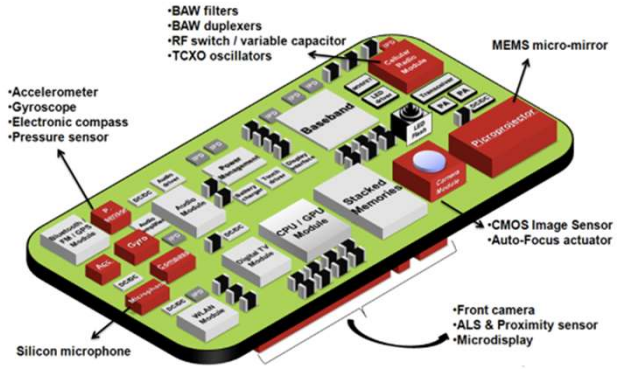


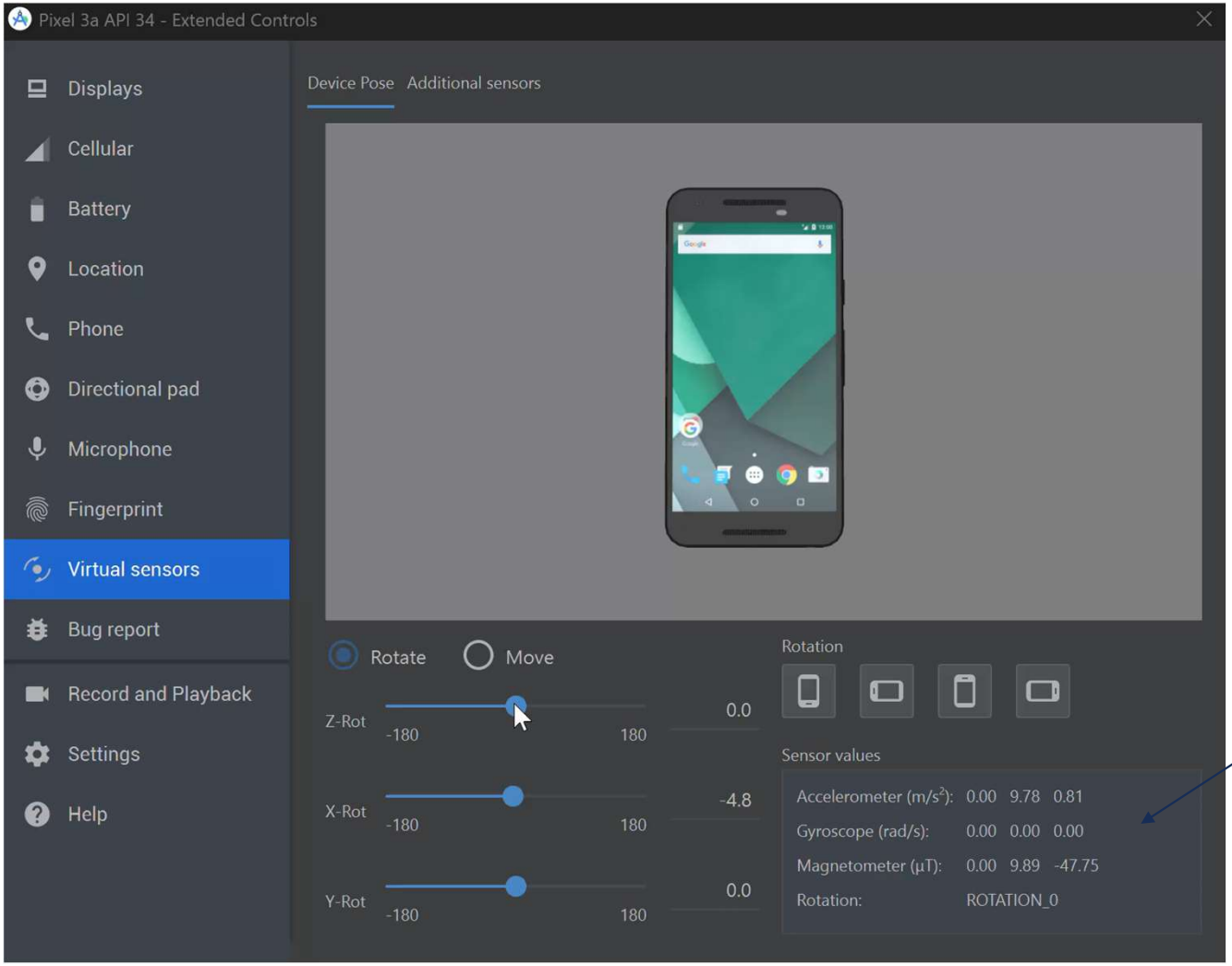
Agent (running on a cloud service) replies

WHY/HOW MOBILE APPS DIFFER FROM DESKTOP ONE?

Three main reasons

1. **Smaller Screen size**
   - Organize the information accordingly, e.g., list
2. **Reacher Input/output system**:
   - Sensors: sensors (GPS, orientation), output: vibration, ..
   - Camera
3. **Energy consumption and efficiency**
   - Apps have a lifecycle (can be suspended)
   - Memory management





Data from sensors sampled at constant rate and **notified** to applications

EXAMPLE OF RAW DATA FROM SENSORS

ΔT

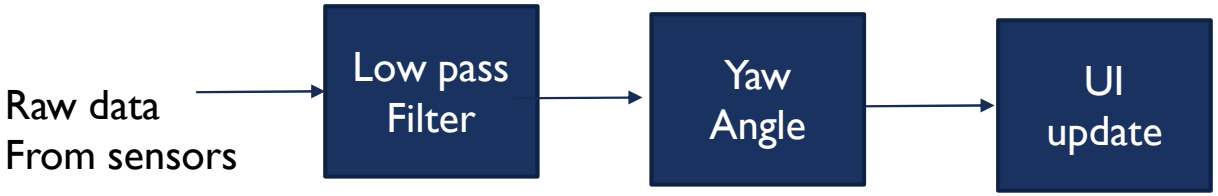
Time (s)	Magnetic Field x (μT)	Magnetic Field y (μT)	Magnetic Field z (μT)	Absolute field (μT)
0,00125025	-17,98095703	25,31533813	-58,80501938	66,49971026
0,01129625	-18,03736877	25,32446671	-58,63410568	66,36741396
0,02134225	-18,09669495	25,49047852	-58,88801193	66,67122927
0,03138825	-18,01911926	25,44477463	-58,59611893	66,37492274
0,04143425	-18,12969971	25,49904251	-58,48772049	66,33023917
0,05148025	-18,09500122	25,47503662	-58,6035347	66,41370972
0,06152625	-18,04927063	25,52371216	-58,80498123	66,59776175
0,07157225	-18,15600586	25,56219482	-58,71731949	66,56417926
0,08161825	-18,08384705	25,53522491	-58,81367874	66,61923178
0,09166425	-18,03250122	25,37212753	-58,72034073	66,46047225
0,10171025	-18,03495789	25,5076561	-58,72506332	66,51716536
0,11175625	-18,20140076	25,54646683	-58,64561844	66,50730425
0,12180225	-18,1255188	25,56977081	-58,36952591	66,25216348
0,13184825	-18,18673706	25,45669937	-58,57455826	66,40617308
0,14189425	-18,14057922	25,61658478	-58,45231247	66,3472898
0,15194025	-18,16746521	25,58127975	-58,60407639	66,47478044
0,16198625	-18,06530762	25,48977661	-58,69140244	66,48883193

Data Resolution

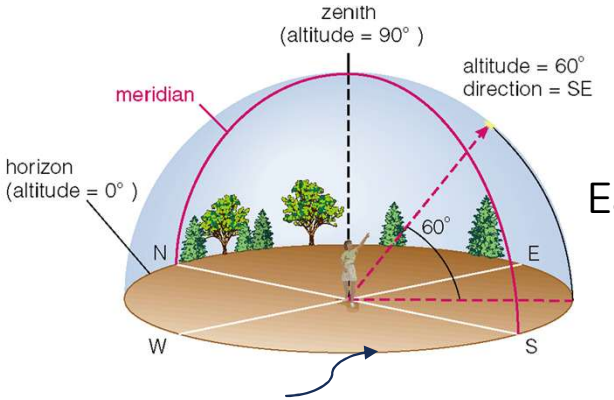
Noise

0.01129625-0.00129625=0.0100460 → f ≅ 100 Hz

DATA PROCESSING

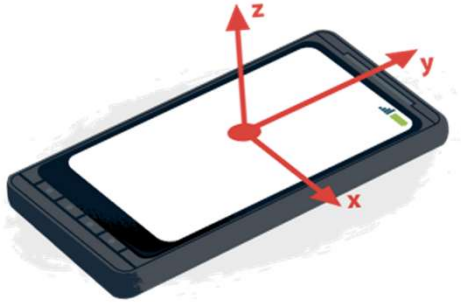


DEVICE ORIENTATION



East North Up frame (**ENU**)

LTP (Local Tangent Plane)



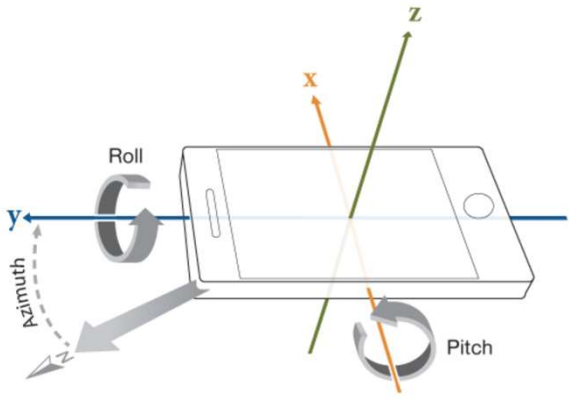
Device Reference Frame (**DRF**)
Reading from sensors given respect to the DRF

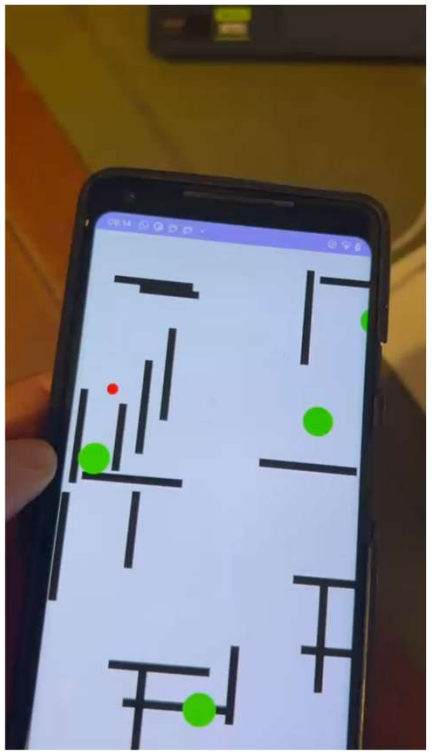
$v_{world} = \mathbf{R} \cdot v_{device}$

The orientation is defined by an Orientation matrix

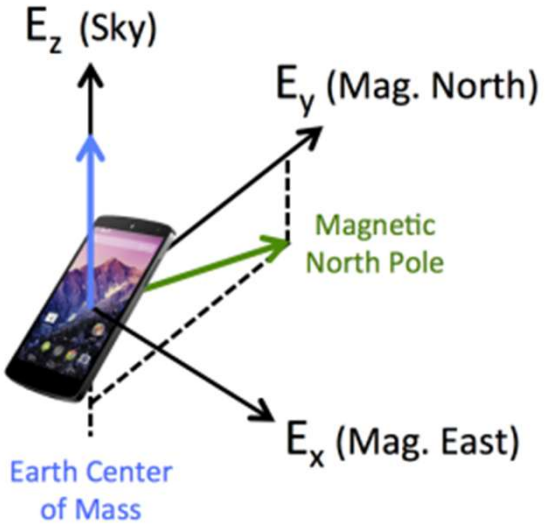
HOW ORIENTATION BECOMES AN INPUT?

- From orientation it is easy to derive two tilt angles (respect to the Local Tangent Plane), that can become inputs of an app
- In this example, they drive the ball





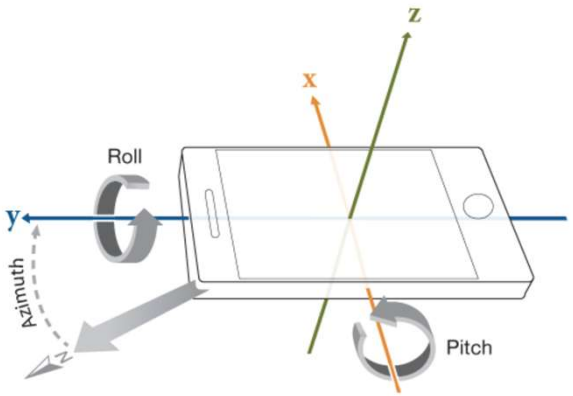
POSE ESTIMATION

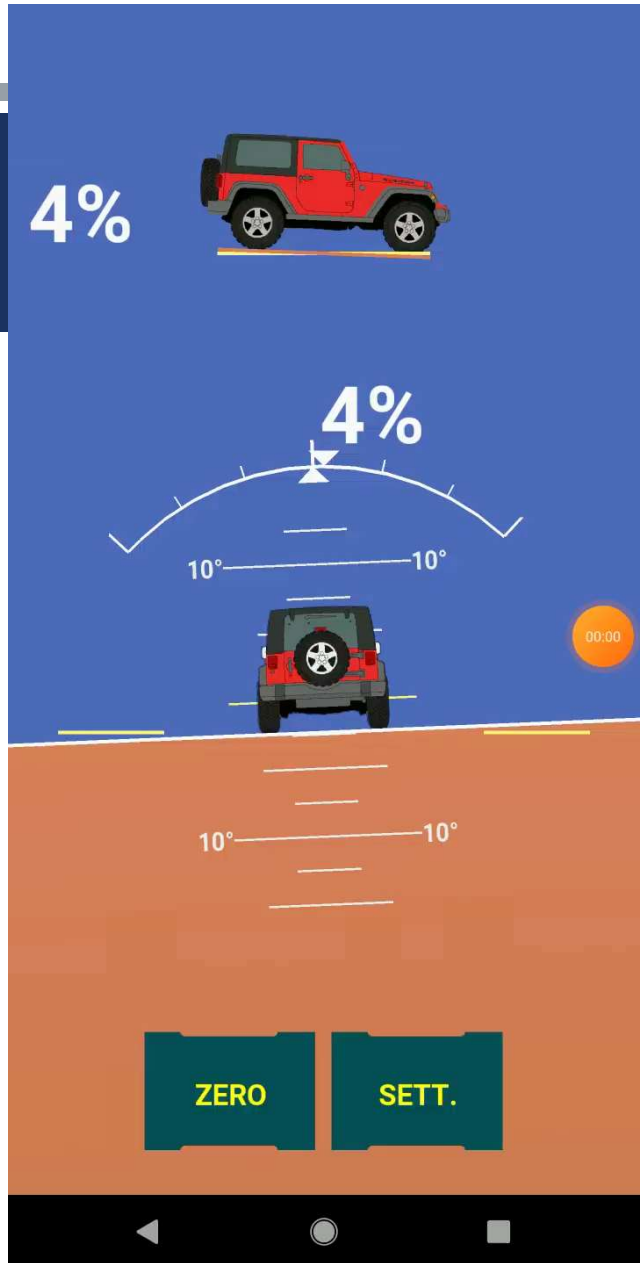


- The problem of determining **R** from sensor readings is known as **pose estimation** problem or attitude estimation
- One simple algorithm also used in android is the TRIAD algorithm that uses accelerometer and magnetometer readings (this is what android also uses)
- A more sophisticated estimation uses the gyroscope and an Extended Kalman filter for fusing all data and makes the orientation available as a rotation vector

INCLINOMITER

- In this example, the car rotates based on pitch and roll



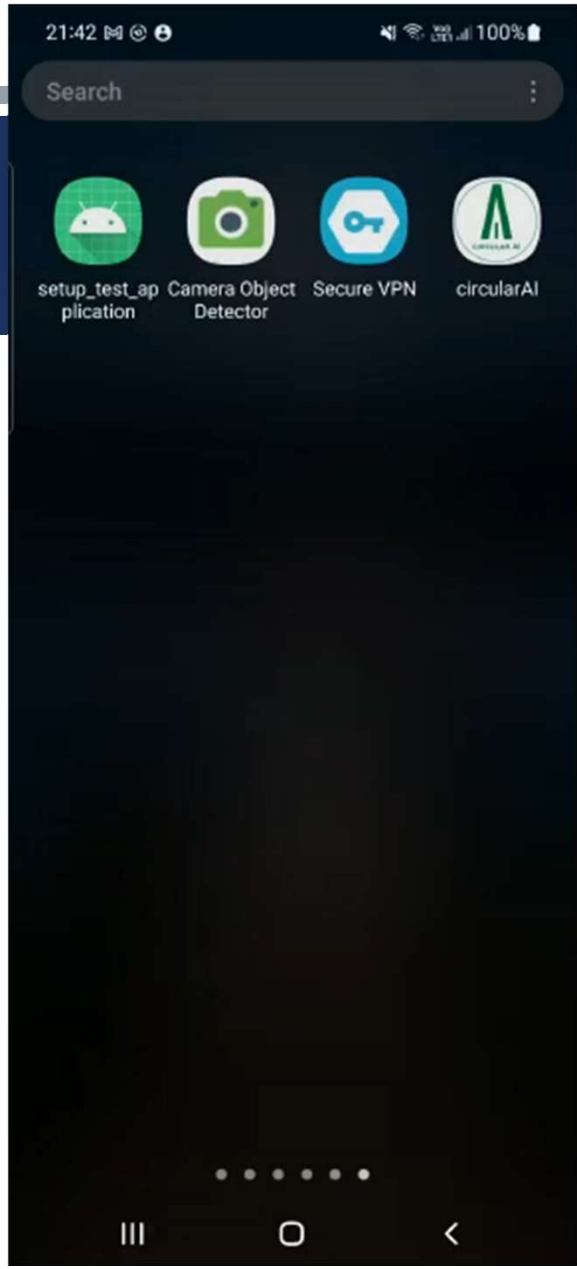


FLY2LIVE

- This is an example of a solo project

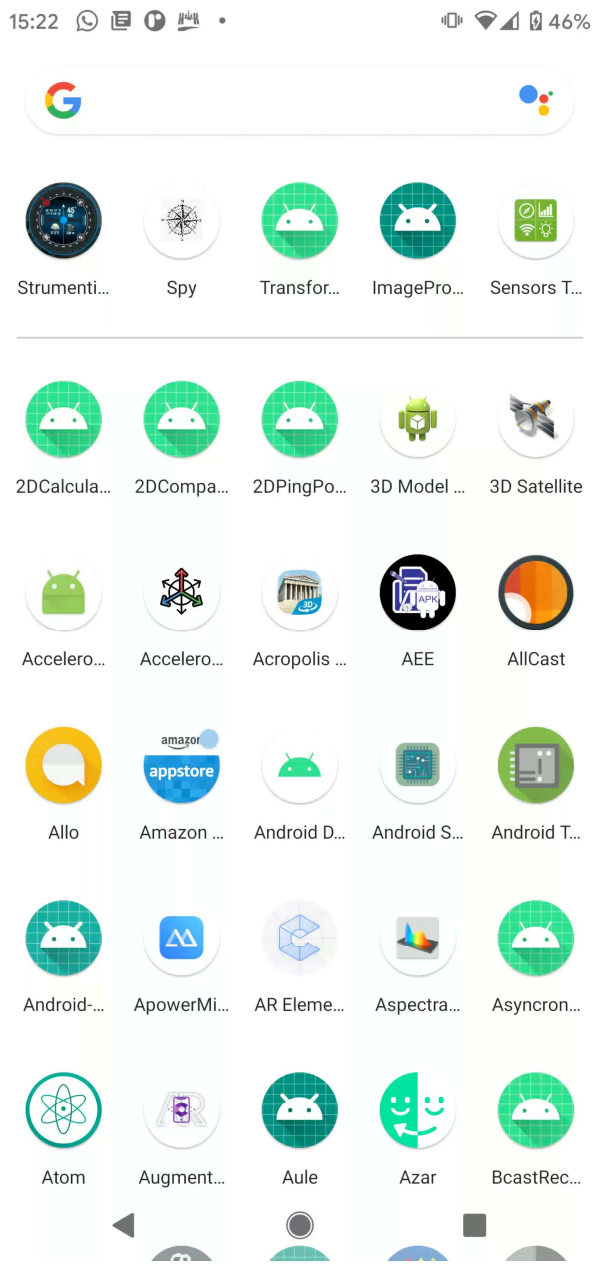
WASTE SORTING WITH AI

- Another example of past project



SIMPLE IMAGE PROCESSING

- This example shows come simple image processing algorithm running on board (taken from OpenCV) and applied to images



BEHIND IMAGE PROCESSING

- Image Access
- OpenCV (Open Source Computer Vision Library)
- Image processing is just the surface of the capabilities of the library
  - more than 2500 state-of-the art optimized algorithms