A Framework for UAV Sensor Fusion

E. Martí, J. García, J.M. Molina
Group of Applied Artificial Intelligence
Universidad Carlos III de Madrid

Contents

Introduction

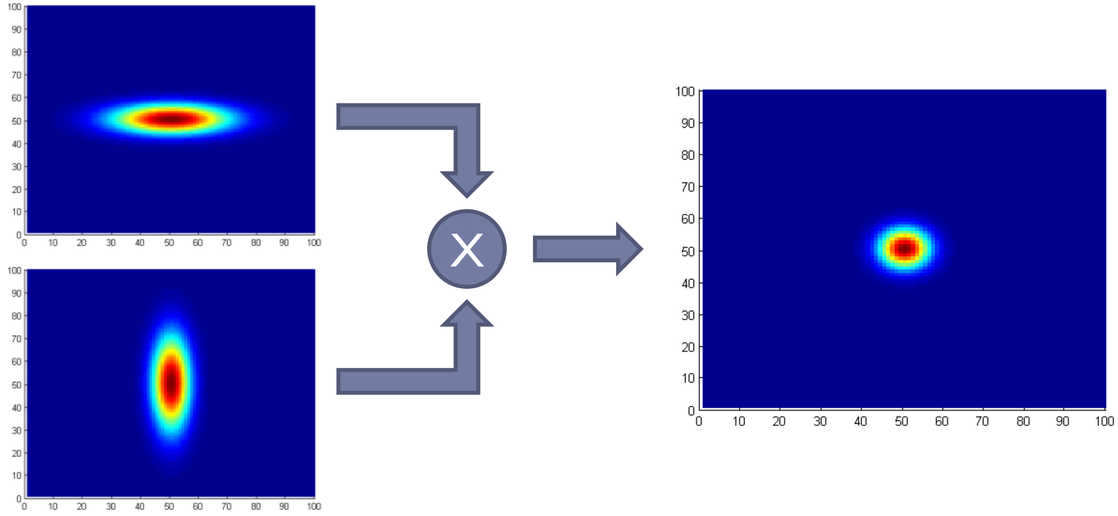
- UAV – Unmaned Aerial Vehicle
  - No human supervision during long time spans
  - **Autonomous**
    - know their own state, operate to achieve goals

Introduction

- UAV – Unmaned Aerial Vehicle
  - No human supervision during long time spans
  - <u>**Autonomous**</u>
    - know their own state, operate to achieve goals

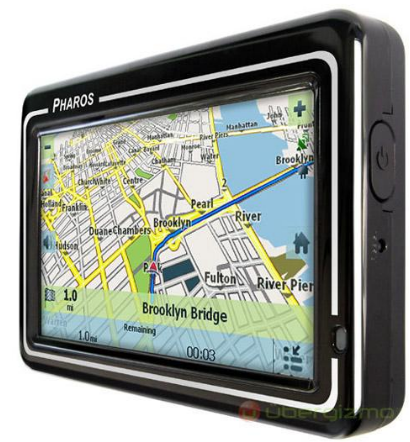
Introduction

- Sensor fusion
  - **Sensors:** imperfect observations about a system
  - **Fusion:** combine information from different sources, so that the final result is better than what we could achieve using each source separately



Introduction

- Sensor fusion
  - Include knowledge about observed system to improve results

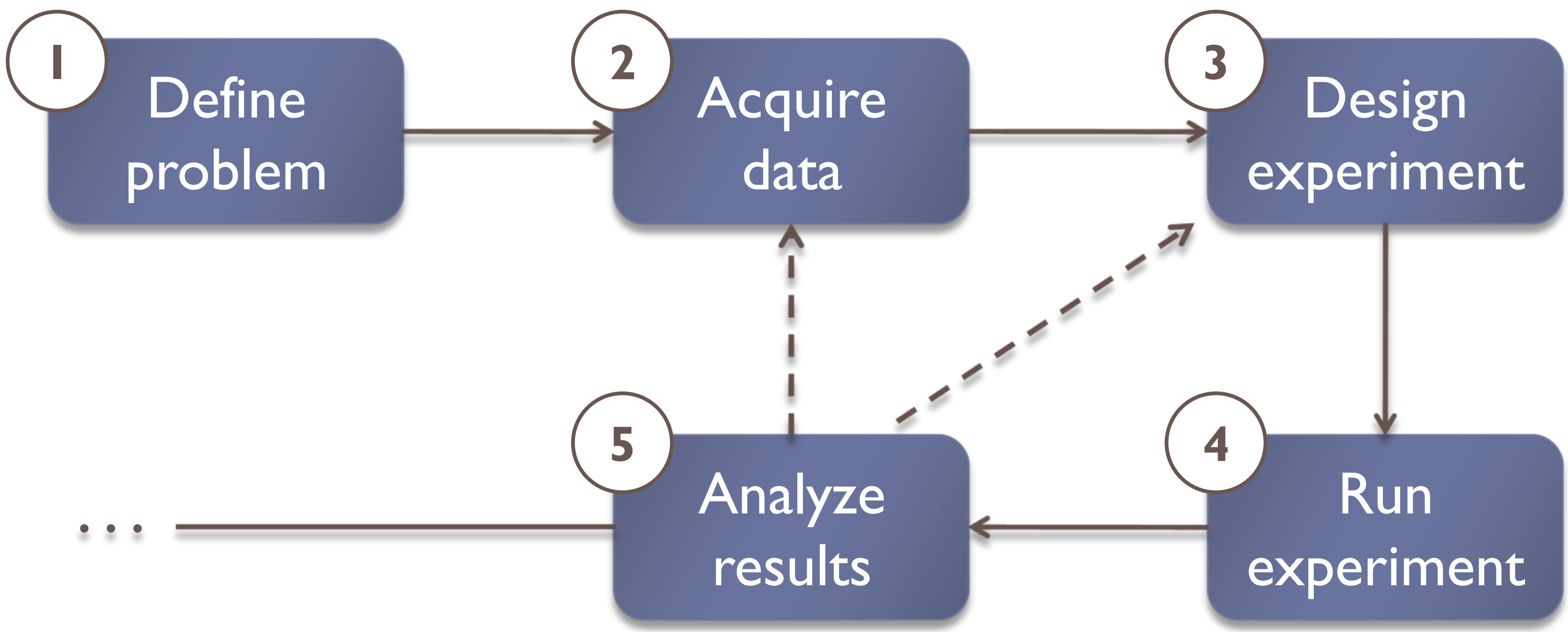


Introduction

- Sensor fusion
  - In our case, this knowledge is a description of the dynamics of a plane
  - Typically, evolution modeled as a first order Markov process
  - Uncertainty sources:
    - Model inaccuracies
    - Sensor measures are not perfect

Why a framework? - Motivation

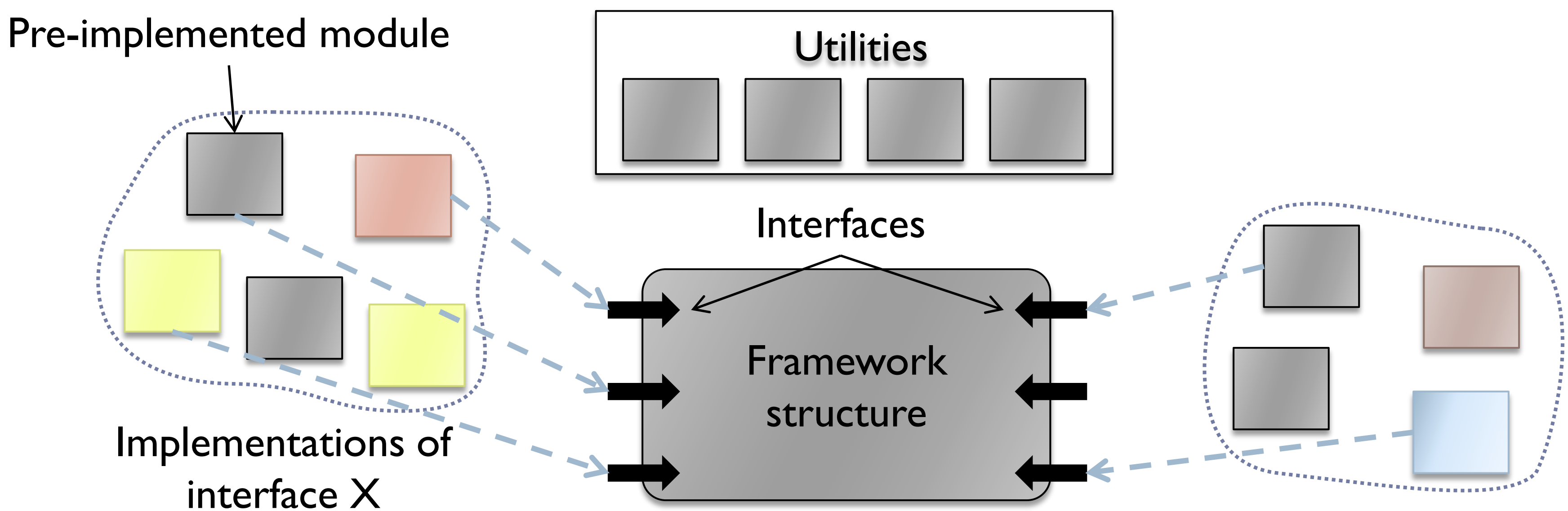
- Typical experimental methodology:



- Repeat steps 3–5 in a loopy fashion
  - Sometimes, new data is needed.

Why a framework? - Motivation

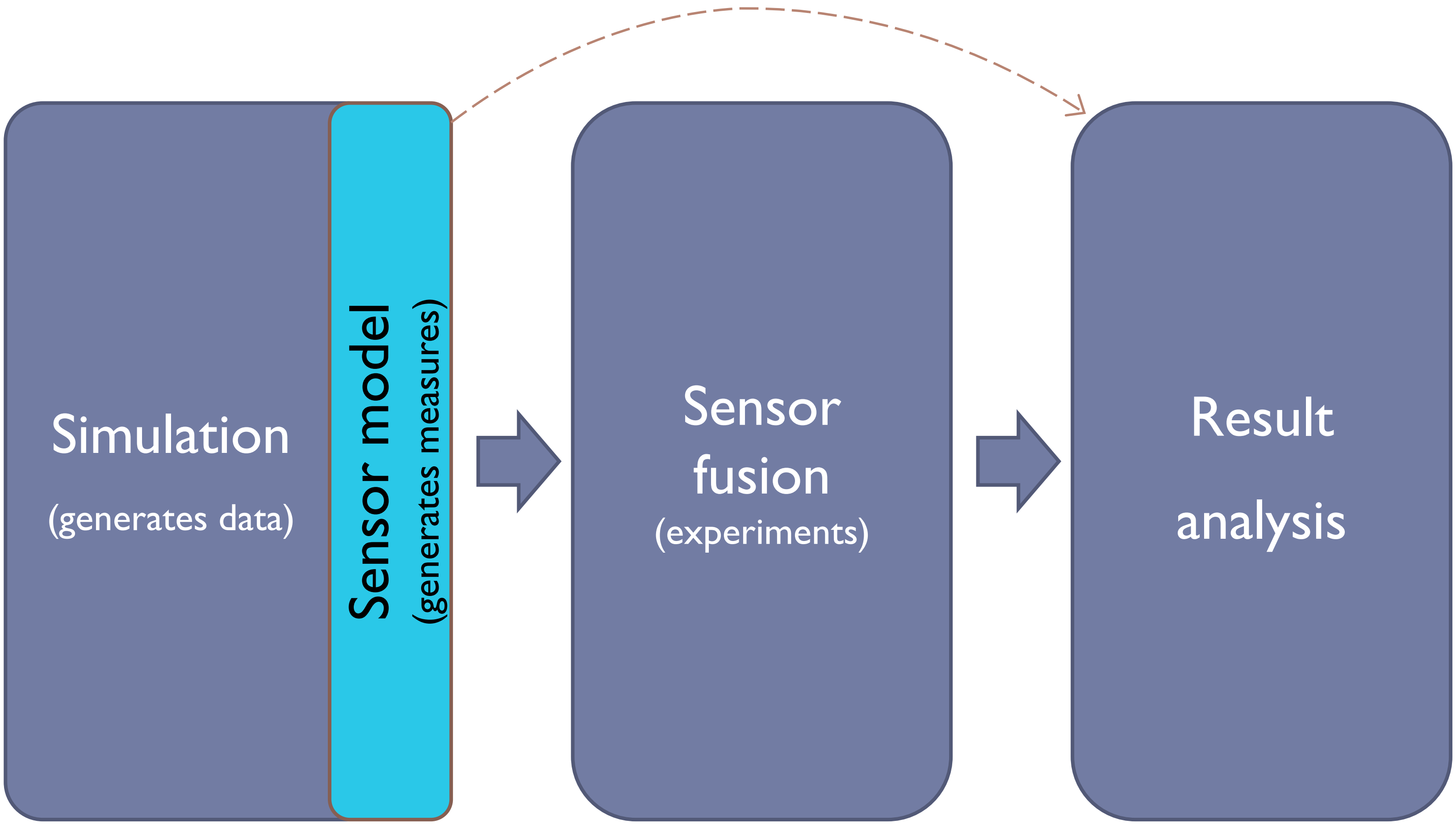
- Save work
  - Implemented parts (structure, utilities)
- Make collaborative efforts possible
  - Architecture, modularity, data specification



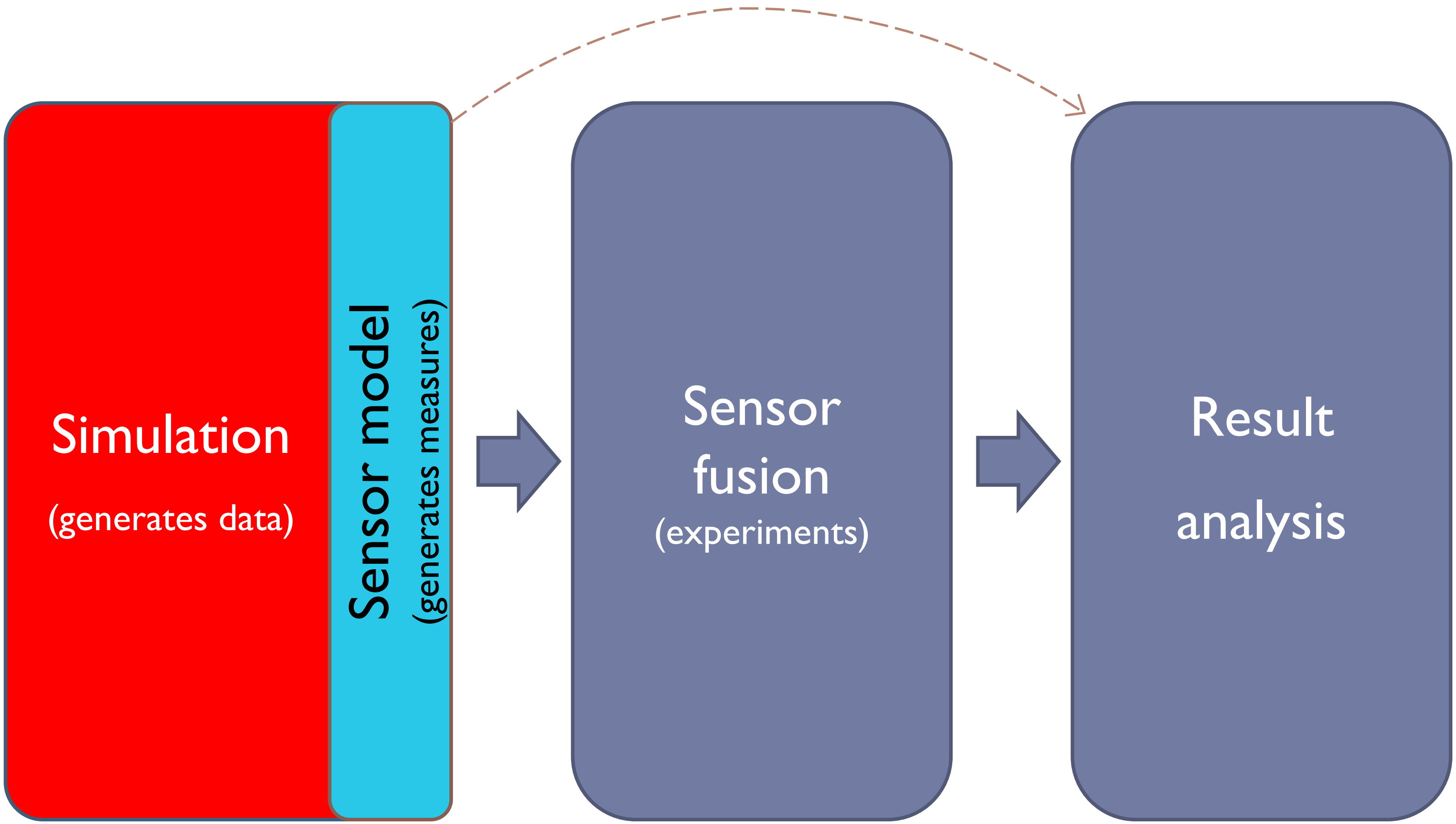
Framework features

- Centralized fusion
  - Single vehicle equipped with sensors
- Passive, event-driven
  - Do “computing stuff” on new events
  - Periodical sensor updates, not under request
- Implementation in MATLAB
  - Fast prototyping (for research)
  - Many useful tools

Architecture



Architecture

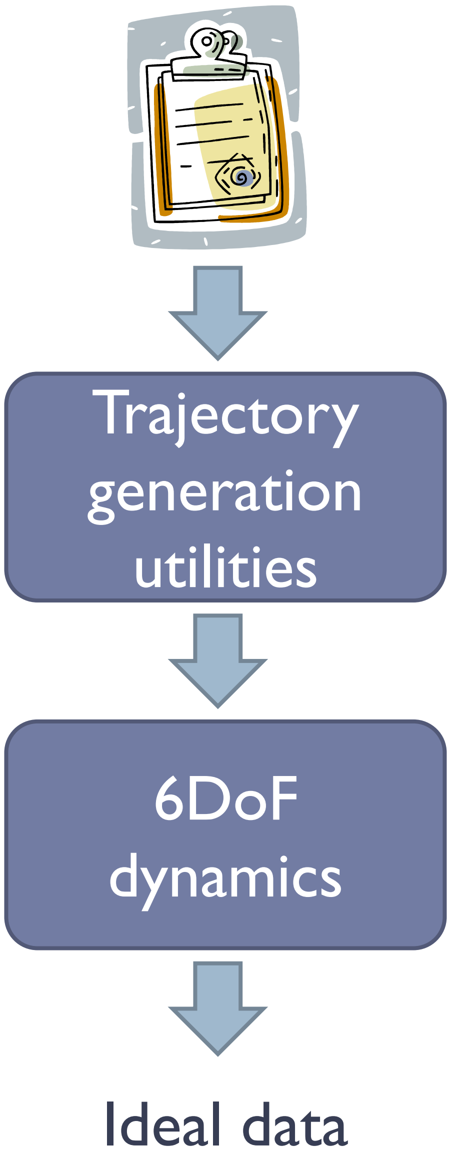


Architecture – simulation

- Why simulation?
  - Independent of external conditions
  - Not unexpected technical issues
  - Fast evaluation of changes
  - Study “extreme” cases (crashes)

Architecture – simulation

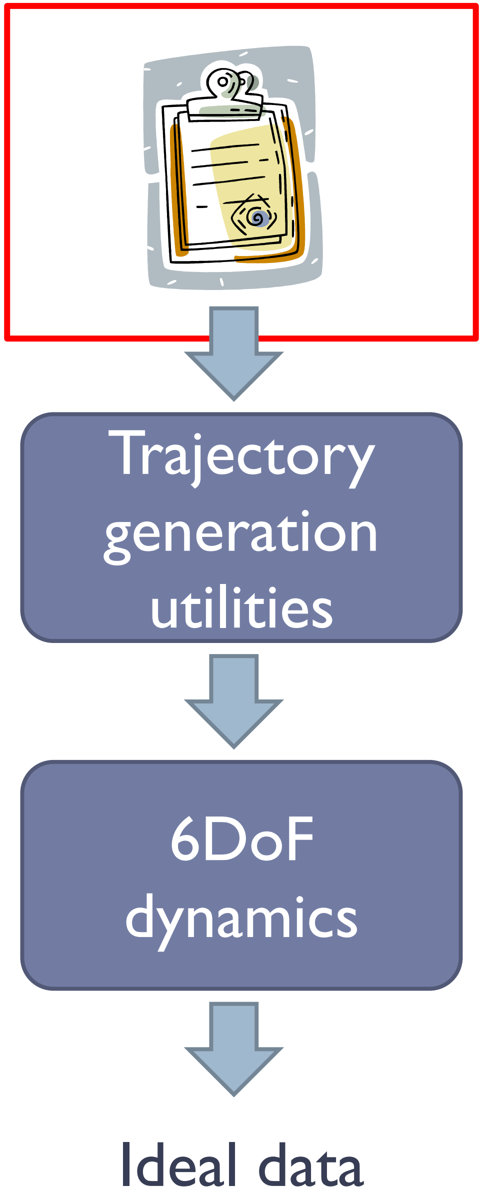
- Basic implementation:
  - High level specification of trajectory
  - Simple Six Degrees of Freedom (6DoF) system
    - Weight, inertia
    - Punctual mass in vacuum (no complex aerodynamics)
- Returns data about the UAV trajectory
  - Required by the sensor simulation module



Architecture – simulation

- Basic implementation:
  - Input to simulation module – high level specification of trajectory

…	
straight flight	(speed, duration)
linear acceleration	(acceleration, duration)
turn	(axis, angle, duration)
straight flight	(speed, duration)
random turns	(number, angle limit, …)
…	



Architecture – simulation

- Basic implementation:
  - Intermediate product: forces and moments
  - Array of tuples [time, values X,Y,Z]

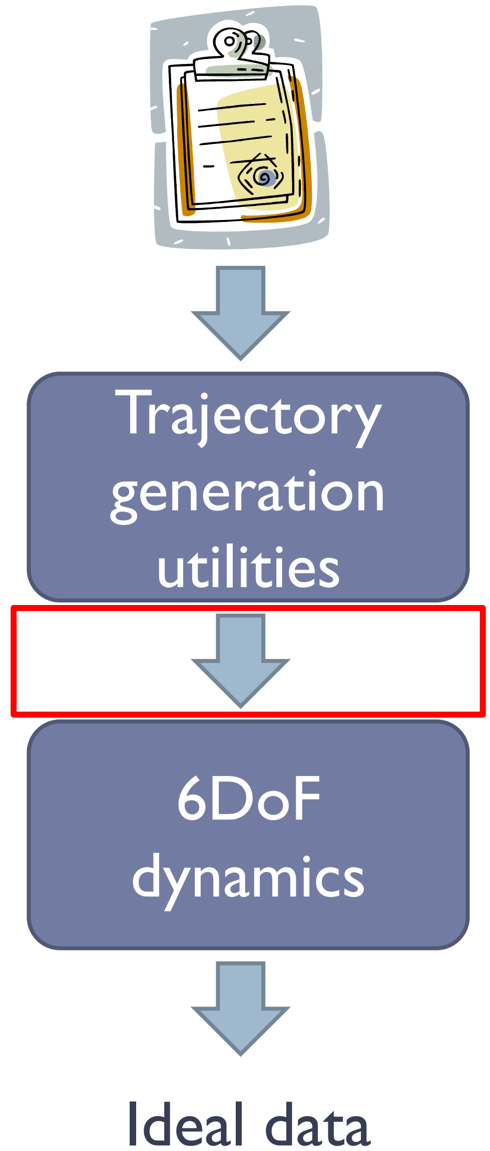
Forces:

time	force X	force Y	force Z
00.00	3.24	-0.03	0.00
02.40	-2.50	0.00	0.30
31.50	0.00	0.00	0.00

Moments:

time	mmntX	mmntY	mmntZ
05.00	0.00	0.00	0.00
07.36	0.00	2.27	-1.45

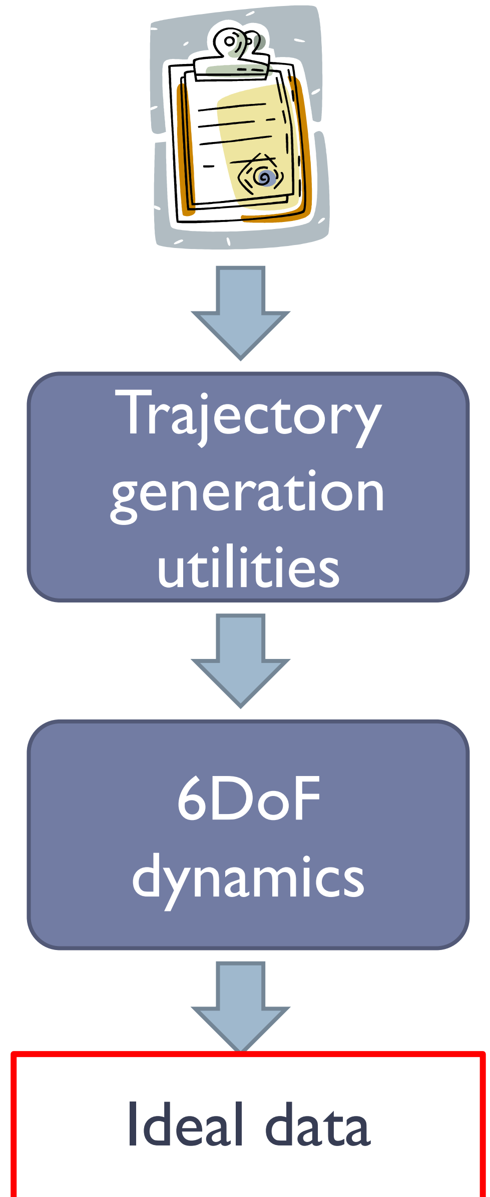
…



Architecture – simulation

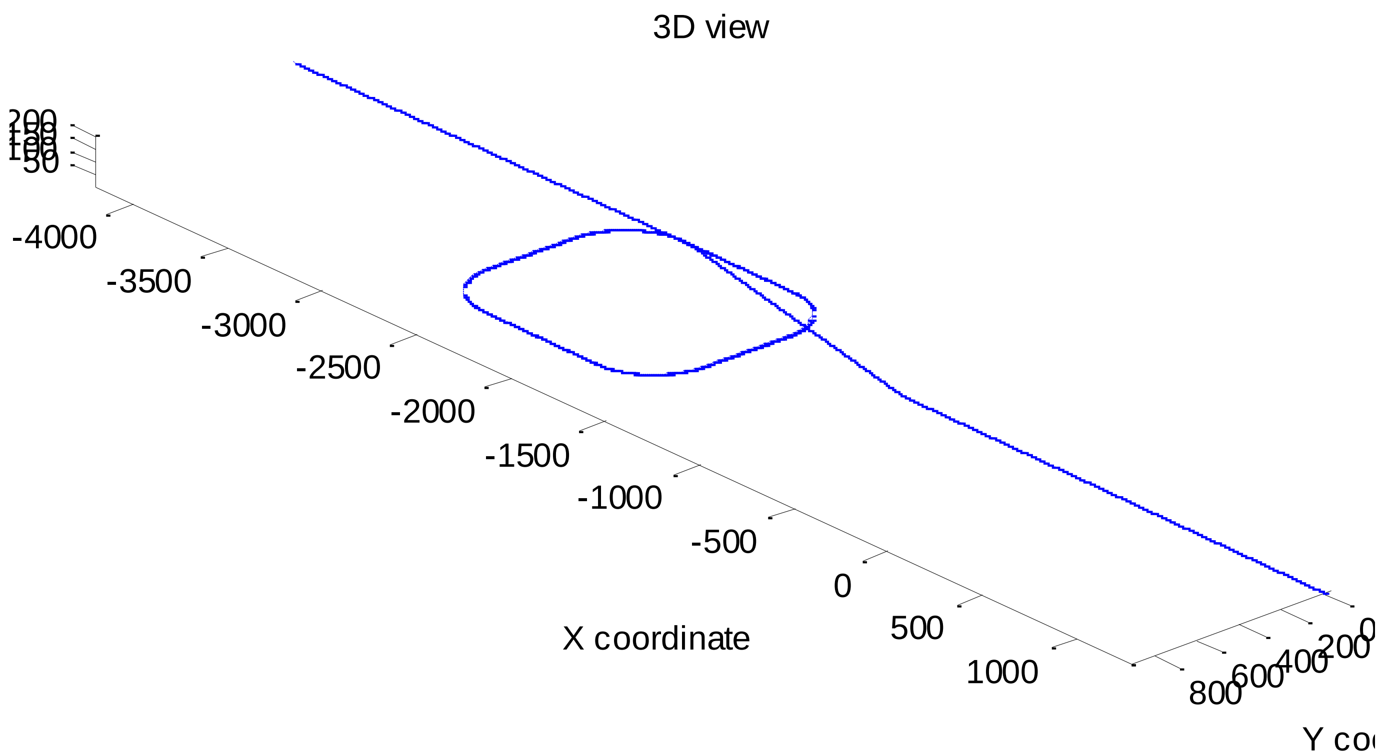
- Basic implementation:
  - Expected output: exact values for the physical attributes of interest
  - Fixed timestep, high frequency (adjustable)

Time	Position	Accel.	Speed	Ang. Rate
0.01	…	…	…	…
0.02	…	…	…	…
0.03	…	…	…	…
0.04	…	…	…	…
0.05	…	…	…	…
…				



Architecture – simulation

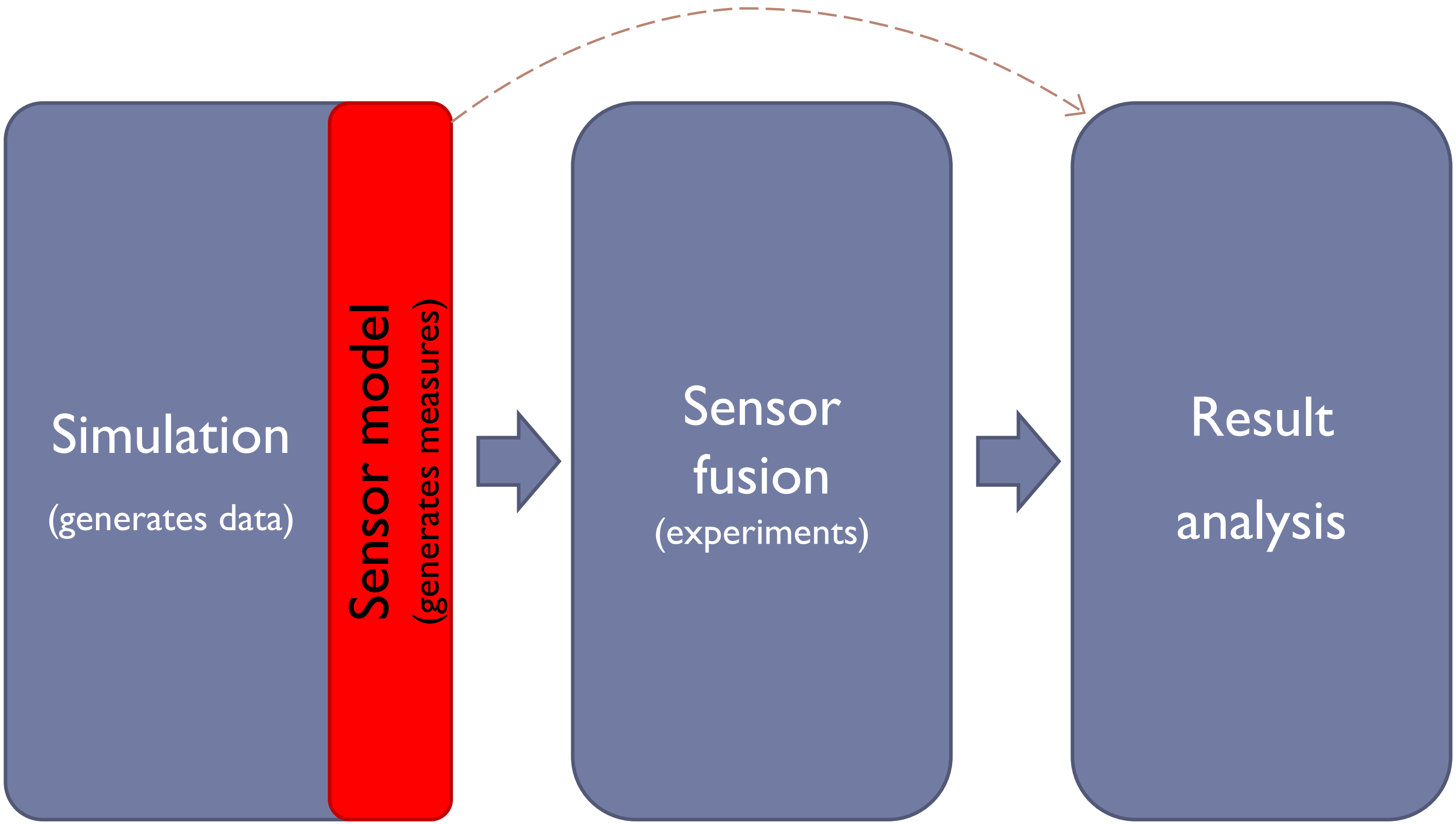
- Application: typical maneuvers in commercial flights



Architecture – simulation

- Can be easily extended/substituted
- FlightGear + JSBSim
  - Live trajectory generation
  - Realistic aerodynamics
  - Atmospheric conditions
  - Output to file
- …and we can use real data!

Architecture – sensor model

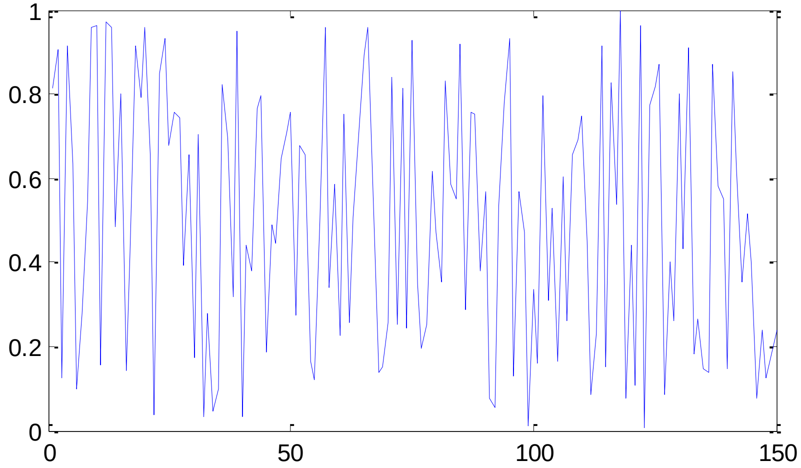


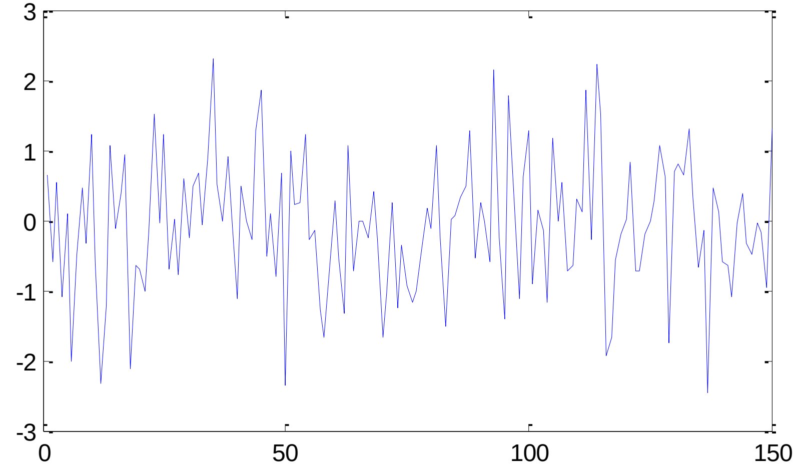
Architecture – sensor model

- **Input:** data from simulation module
- **Output:** the measures that a real sensor would display
- Several devices already implemented
  - Matlab classes/functions
- Configurable parameters
  - "amount of error", update rate…

Architecture – sensor model

- Emulate inaccuracies and other effects of real life sensors
- Examples:
  - **Noise:** independent random perturbations of measures



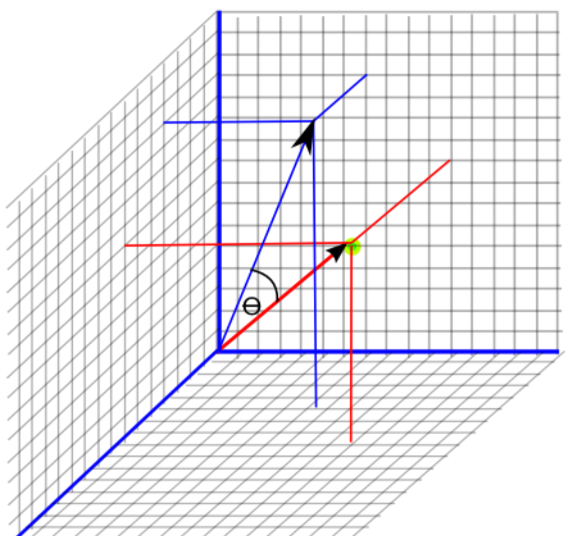


Architecture – sensor model

- Emulate inaccuracies and other effects of real life sensors
- Examples:
  - **Noise:** independent random perturbations of measures
  - **Bias:** persistent, slow-changing deviation between real value and measure

Architecture – sensor model

- Emulate inaccuracies and other effects of real life sensors
- Examples:
  - **Noise:** independent random perturbations of measures
  - **Bias:** persistent, almost-constant deviation between real value and measure
  - **Correlated errors:** physical effects that affect several components of a measure at the same time

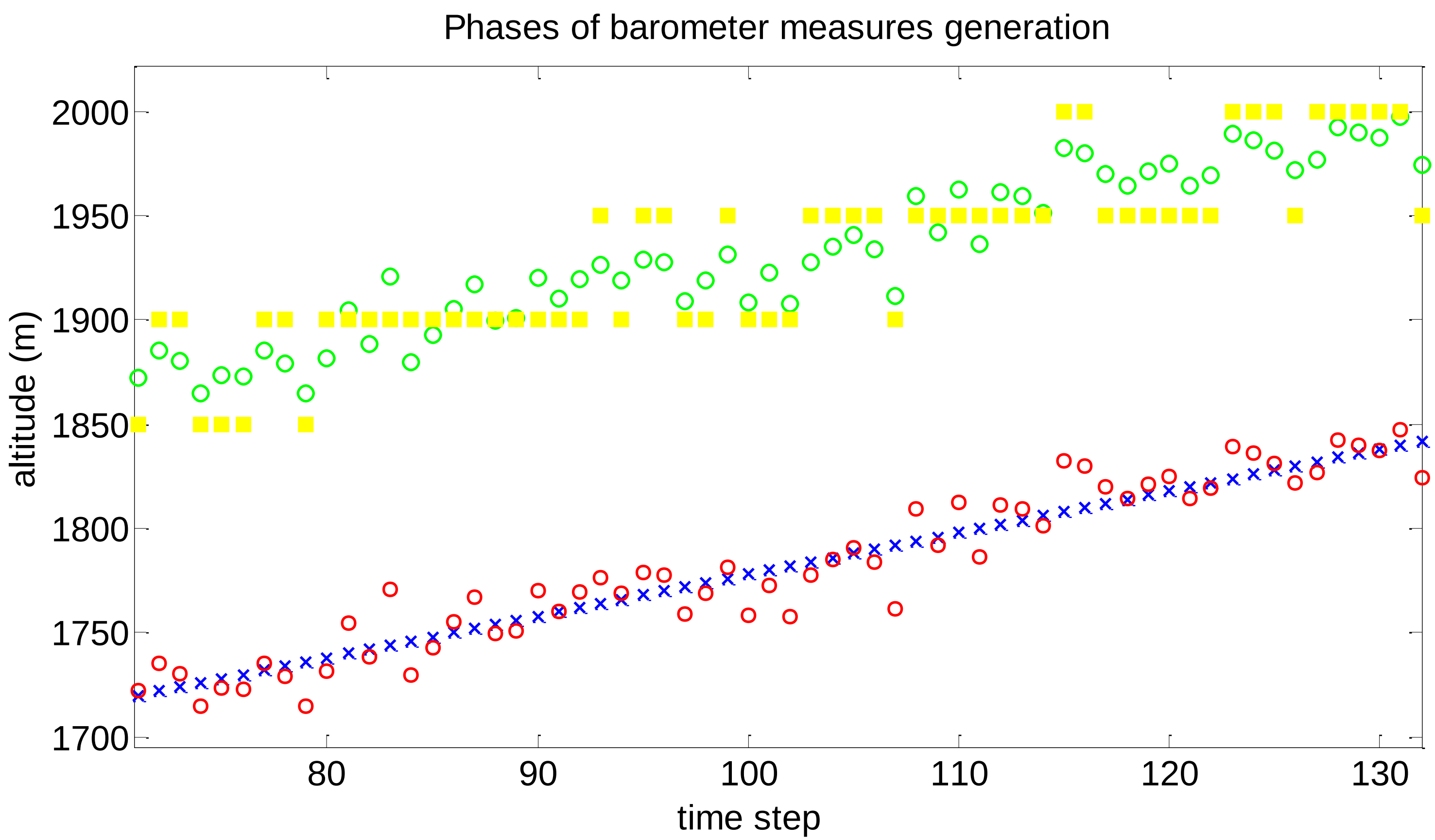


Architecture – sensor model

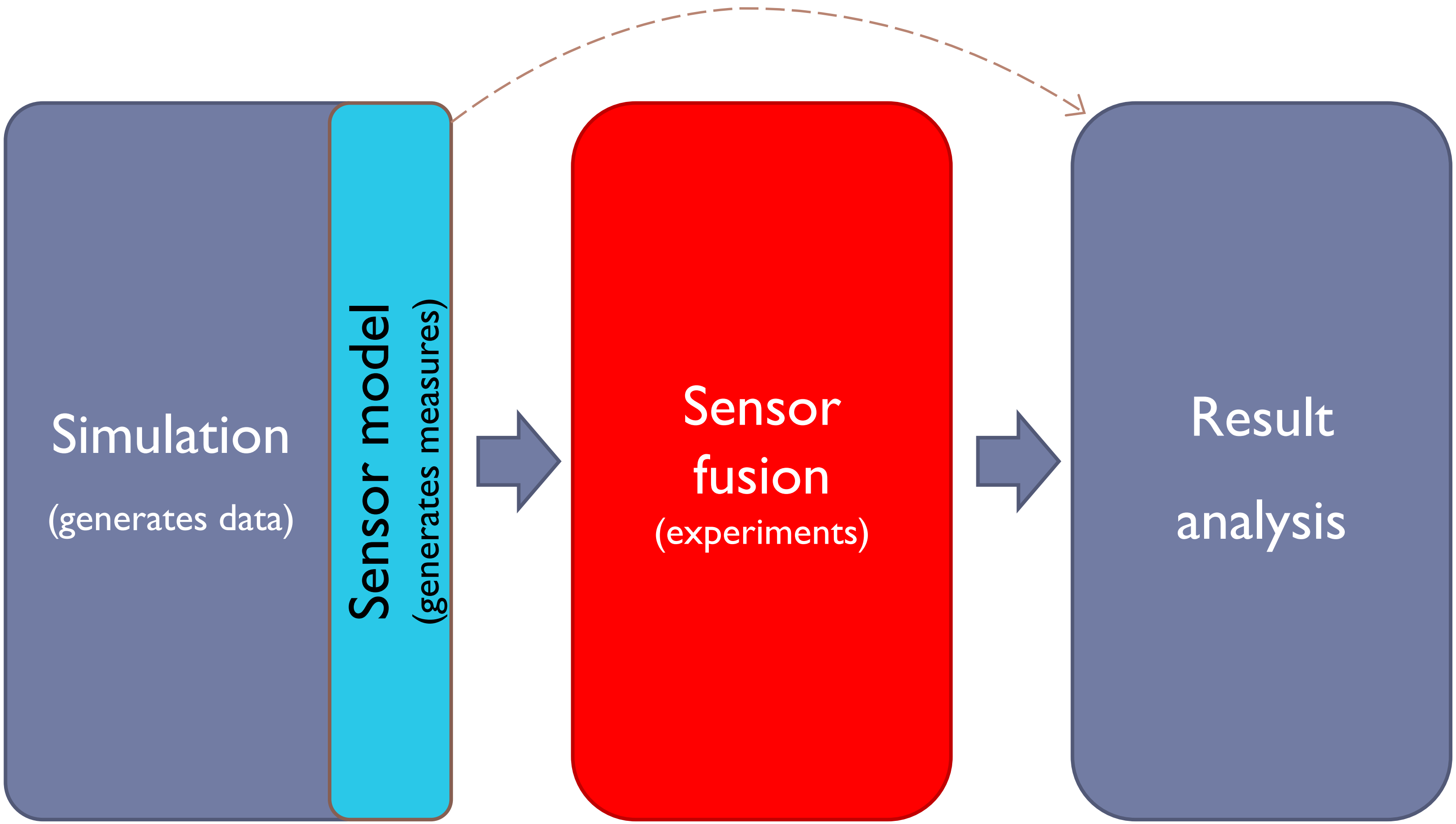
- Emulate inaccuracies and other effects of real life sensors
- Examples:
  - **Noise:** independent random perturbations of measures
  - **Bias:** persistent, almost-constant deviation between real value and measure
  - **Correlated errors:** physical effects that affect several components of a measure at the same time
  - **Quantization:** devices measuring a continuous attribute, but their output vary discretely

Architecture – sensor model

- Example: barometric altimeter simulation



Architecture – sensor fusion

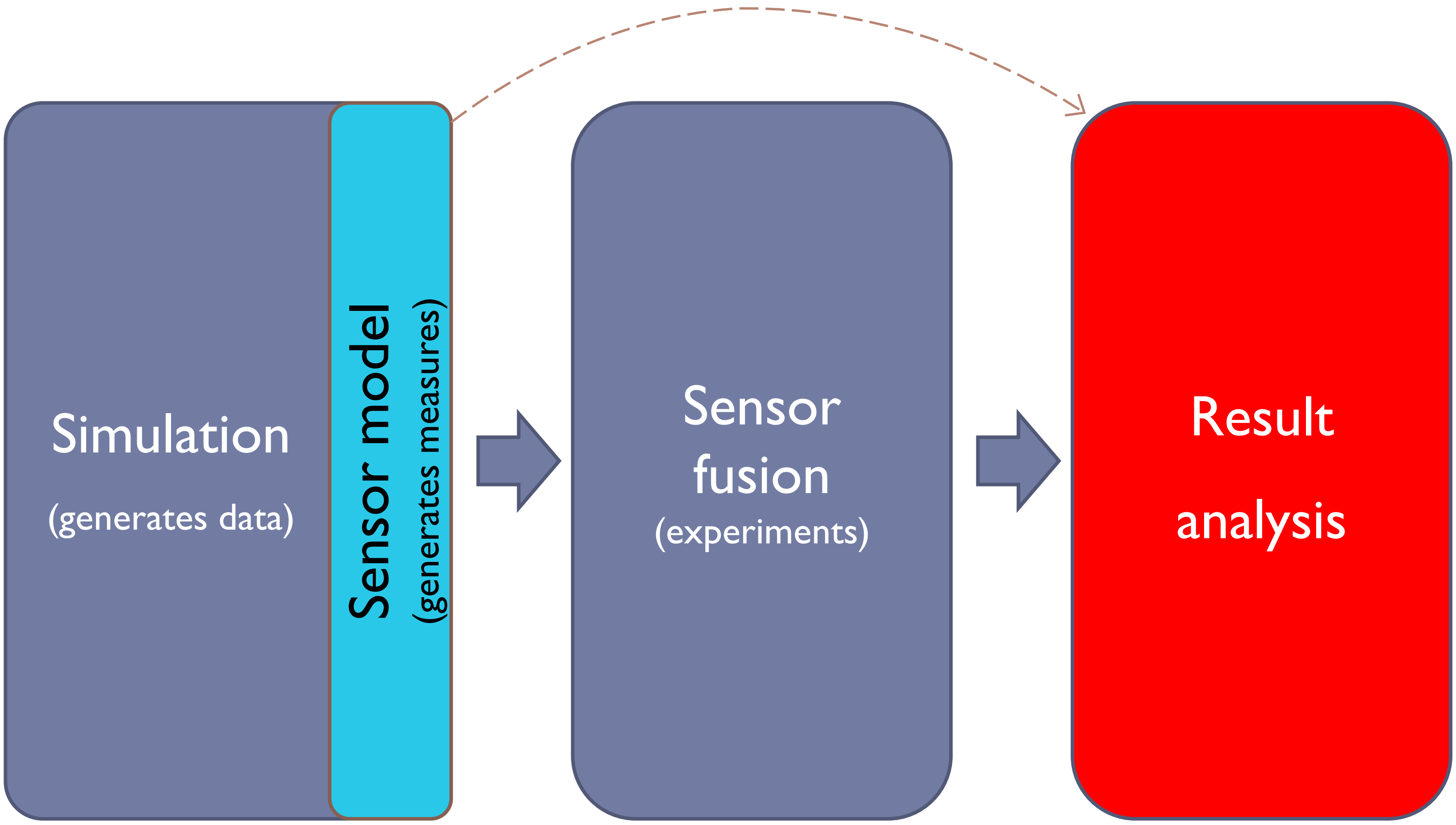


Architecture – sensor fusion

- Flexibility!
- Sample "template" scripts

- Implementation of classical techniques:
  - Kalman Filter (KF)
  - Extended KF
  - Unscented KF
  - Particle Filter
- "Soft interface":
  - Respect particularities of each technique
  - Minimize impact when switching amongst them

Architecture – result analysis

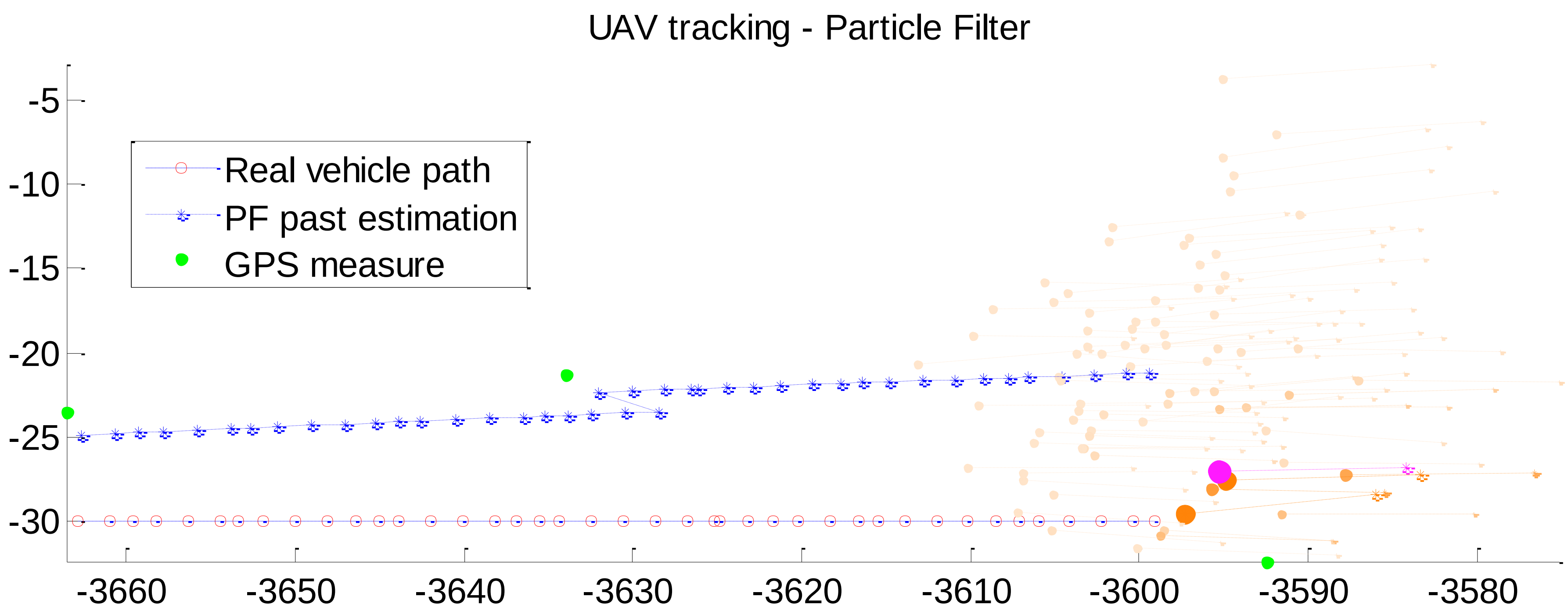


Architecture – result analysis

- Numerical output
  - Statistical validation
  - Performance indices
- Visual output
  - System evolution **during** experiments
  - Performance under special external conditions
  - Comparison of algorithms/configurations

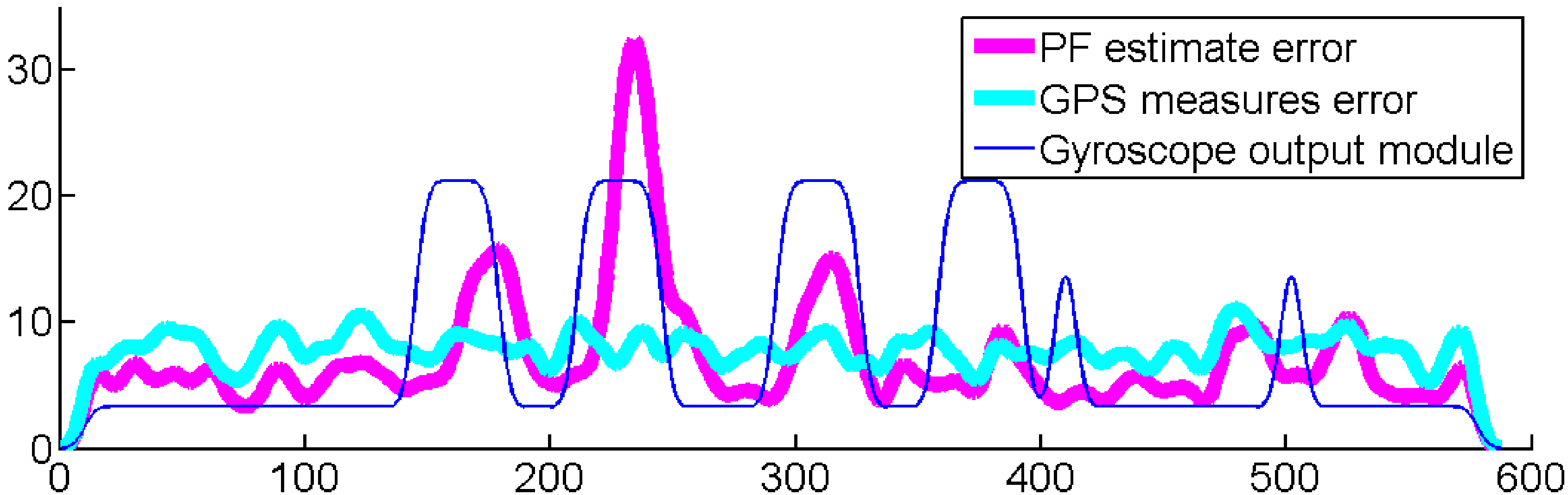
Architecture – result analysis

- System evolution during experiments



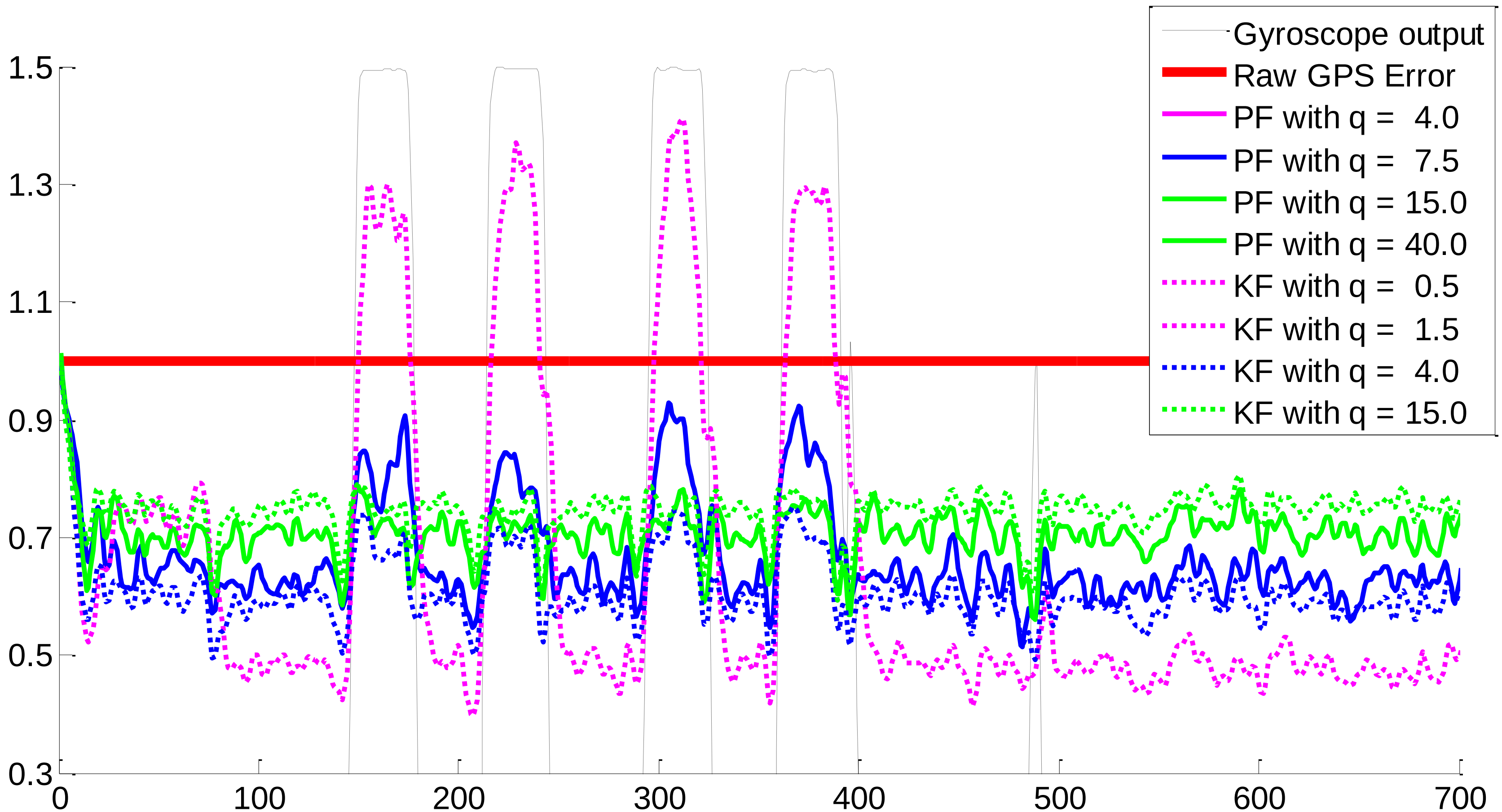
Architecture – result analysis

- Performance under special external conditions



Architecture – result analysis

- Comparison of different algorithms/configurations



Conclusions

- Framework covers the whole experimental process
- Easy to try different parameters/algorithms
- Adding new parts is quite straightforward
- Focused on visual performance evaluation

Questions & suggestions

Thank you very much!