



Alojz Gomola  
15<sup>th</sup> January 2018

## Obstacle avoidance based on Reach Sets

Short story how to keep your UAV safe

**Honeywell**  
THE POWER OF CONNECTED

# What I am going to Talk about

## **Easy introduction to Obstacle Avoidance**

- Follow Lojzo on his journey to Pub
- Obstacle Avoidance challenges related to Everyday situation

## **Obstacle avoidance framework**

- Brief introduction to obstacle avoidance framework
- What are components how they work together ?

## **Some results**

- Framework performance comparison with other methods
- Intruder avoidance (Reactive level)
- Static obstacle avoidance (Reactive level)

## **Actual challenges**

- Weather incorporation into obstacle avoidance, Rules of the Air

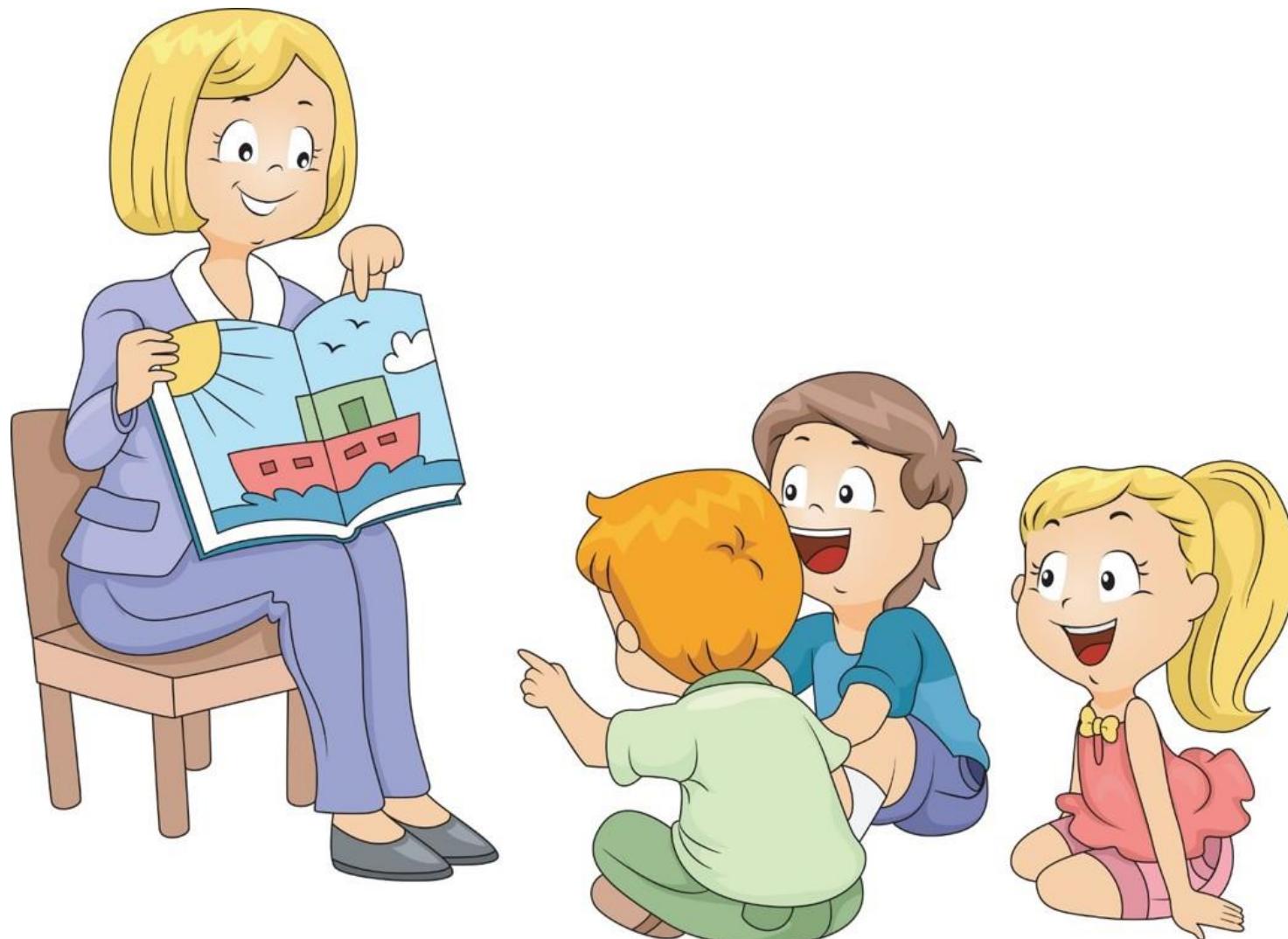
# Global research topic

## **Obstacle avoidance for UAV/UAS/RPAS systems:**

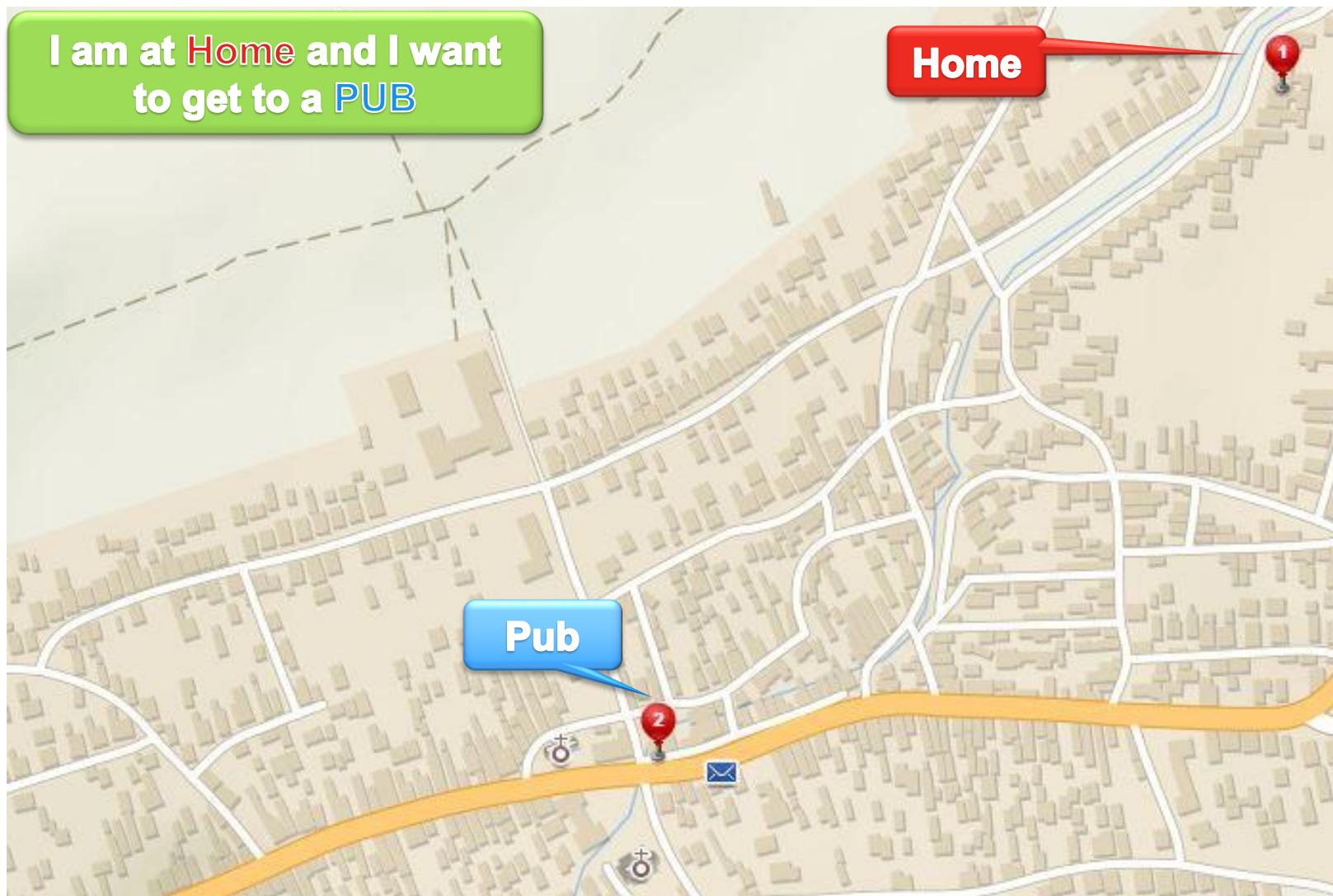
Given:

- partially known world, changing weather, information sources
- mission plan,
- small UAV equipped with a LiDAR and ADS-B sensors
  
- Safe and robust approach to real-time reactive obstacle avoidance
  - small computational footprint
  - guarantees safe path
  - guarantees turn return checkpoint
  
- Main applications:
  - terrain avoidance for low altitude flights
  - avoidance of intruders

# Easy introduction to Obstacle Avoidance



# How to get to a pub ?



# No problem, I know a map of my surroundings

## [1] Path planning problem:

1. Get map of surrounding area
2. Find shortest path between waypoints  
Use of safe routes Optional

Path to PUB



# Forget your map, know PUB location

You are in front of your Home

You can sense some of your surroundings  
(Field of Vision)

You know where is PUB (GPS)



## [2] Evolving world:

1. You are obtaining more knowledge as you move
2. You need to make Decisions when you reach your FOV boundary

# I get to the first intersection, now what ?

**[2] Evolving world:**

Decision Point – point of safe return (I can go home), where I do new sensor scan and decide follow up trajectory

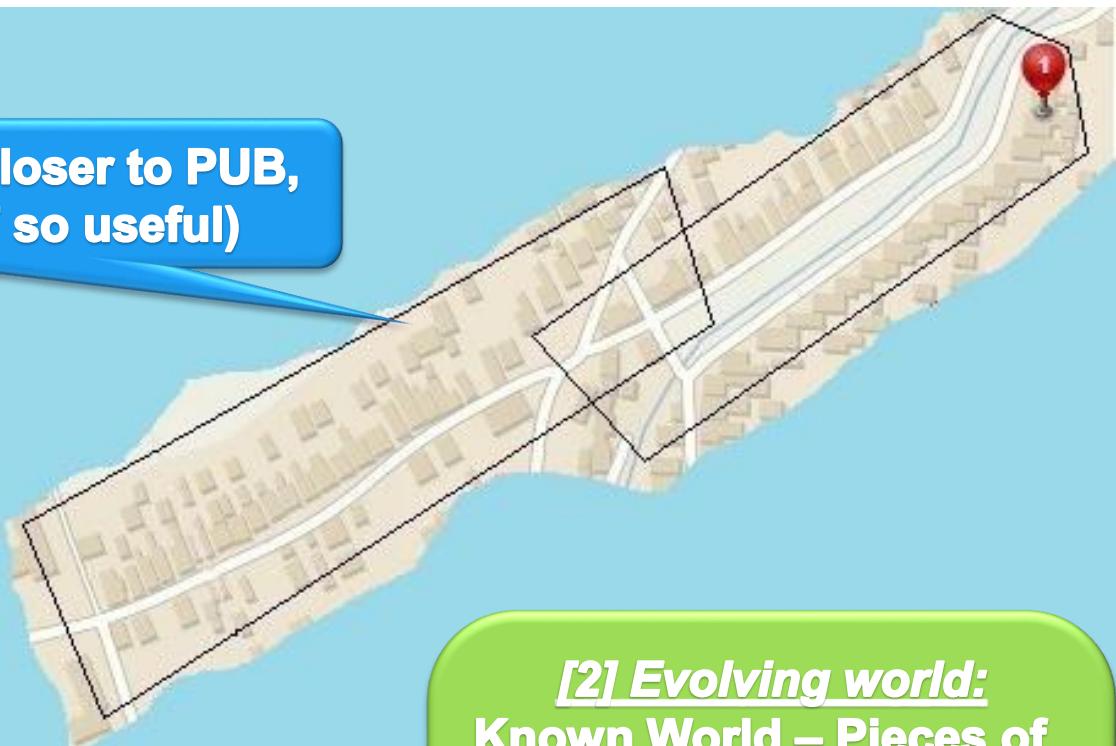
You know where is PUB (GPS)

You will probably apply some sort of COST FUNCTION in your decision

Passed trajectory

# Look around, get know to your surroundings

This patch can get me closer to PUB,  
(Other sightings NOT so useful)



You know where  
is PUB  
(GPS)



**[2] Evolving world:**  
Known World – Pieces of observations which are useful for the mission goal, you can define own set management, based on performance requirements

# Walk a little more, get closer to PUB

**[2] Evolving world:**

Rinse and repeat the previous strategy



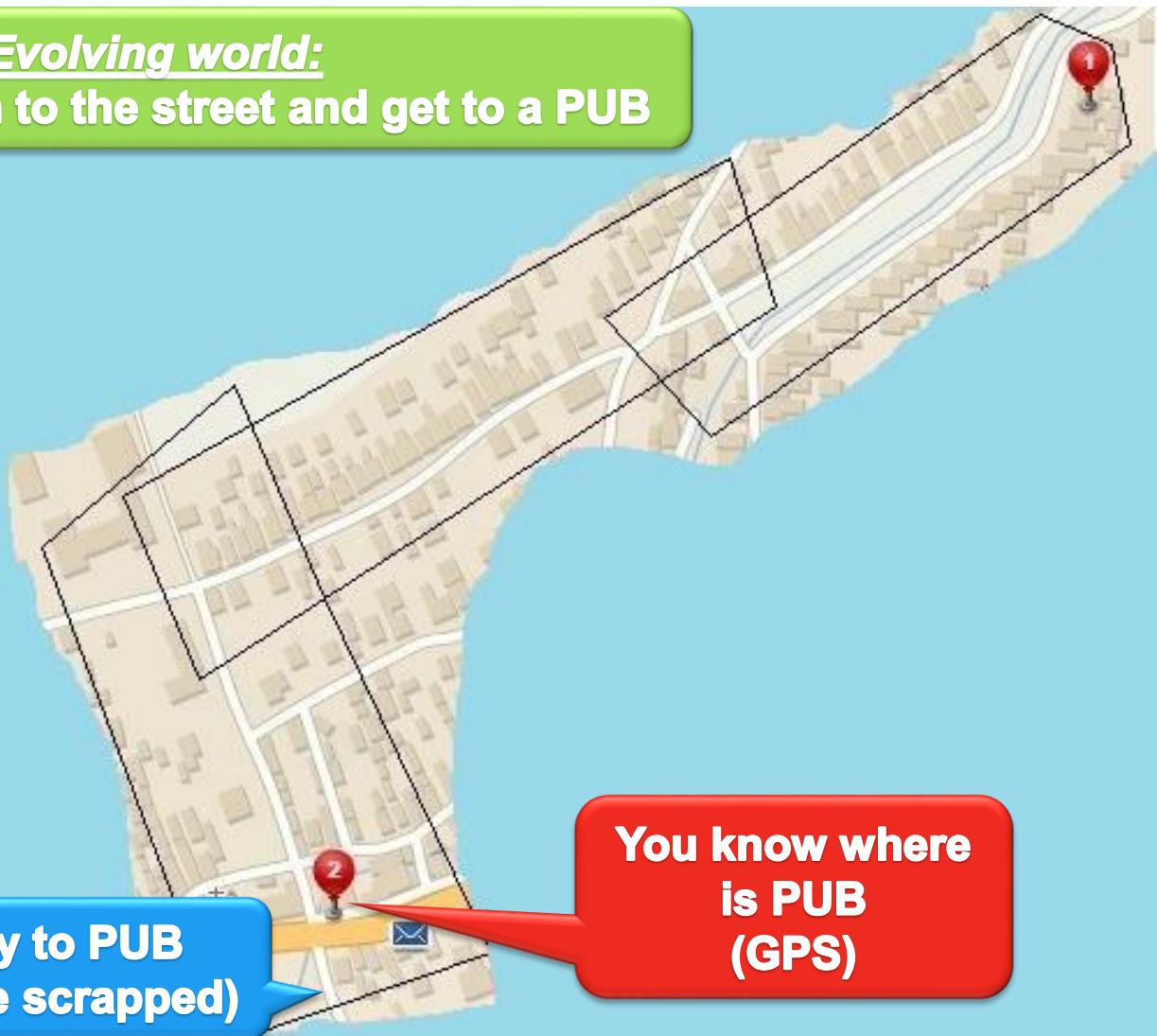
You know where  
is PUB  
(GPS)



# Now I can see the PUB

**[2] Evolving world:**

Now just walk down to the street and get to a PUB



I can see free way to PUB  
(Other sightings are scrapped)

You know where  
is PUB  
(GPS)

# I can finally enjoy the Cola with my friends

## [2] Evolving world:

1. Optimal paths in given FOVs
2. Evolving world cause suboptimal paths  
(Note that path is different)



# You are NOT alone on the streets !

## [3] Adversarial:

While you are moving to PUB  
you need to be cautious about:

### Cars and Other people

Because hitting them can cause  
damage to you or other people  
or property



## [3] Adversarial:

You need to implement:

### Reactive obstacle avoidance

You can assume:

1. Position of Adversary
2. Its Velocity vector
3. Own position and velocity

# Multiple source of information – Data Fusion

## **[4] Data fusion:**

You are well prepared, you ask locals, you got map, you can see a world, but to make decision you need to process that information:

### **Data Fusion**

Is there to help



## **[4] Data Fusion:**

You need to implement:

### **Decisions under uncertainty**

1. Determine what is truth
2. Consider all information
3. Rank your sources worth

# Gopniks & Babushkas – Static restriction areas

## [5] Static restrictions:

Well there are inhabited areas which you like to avoid, like Gopnik Squat Area where you can be beaten or Local Surveillance system which will tell your grand mother



## [5] Static restrictions:

1. Breakable – You can tell your grand mother you were on your way to local church
2. Unbreakable - Gopniks will beat you anyway

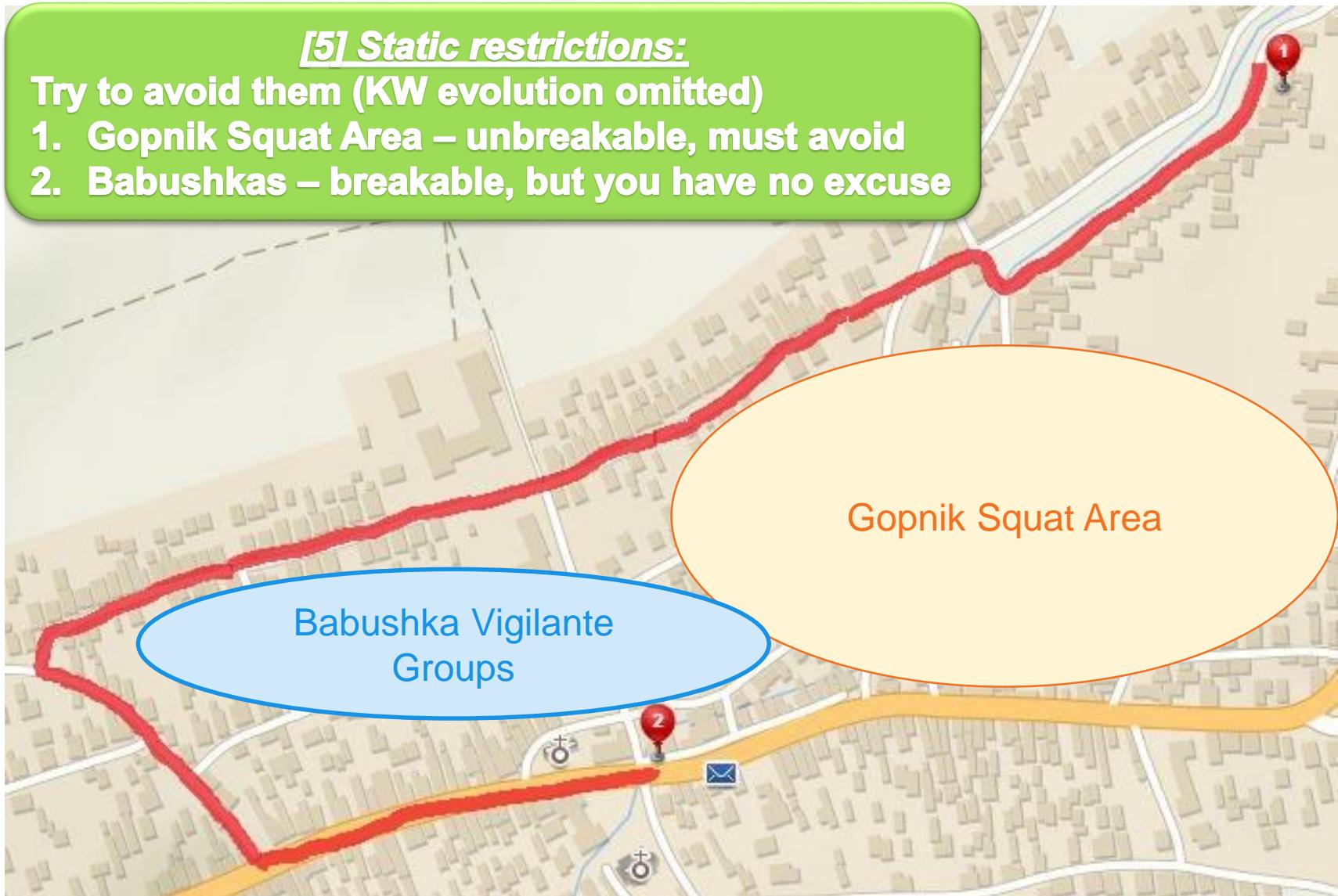
**Try to avoid all restricted areas**

# Try to avoid static restrictions areas !

## [5] Static restrictions:

Try to avoid them (KW evolution omitted)

1. Gopnik Squat Area – unbreakable, must avoid
2. Babushkas – breakable, but you have no excuse



# How about weather ?

## [6] Weather:

**Slav winter cold harsh mistress,  
If you don't treat her well she will  
strike with sickness for 7 days**



## [6] Weather:

- 1. Weather can impact energy consumption**
- 2. Weather is changing with time**
- 3. Decision making with additional layer**
- 4. Impact on your eyes and ears**

# Rather get caught, than catching cold ...

## [6] Weather:

Try to avoid unfavorable weather conditions:

1. Heatwave at EAST (changing with time)
2. Snowstorm at WEST (changing with time)

**Snowstorm**



I will explain  
somehow...

Babushka Vigilante  
Groups

Gopnik Squat Area

Heatwave

# Keep good manners !

## **[7] Rules:**

Keep rules while walking,  
keep yourself on right  
side of the road and look  
after other attendants ...

## **[7] Rules:**

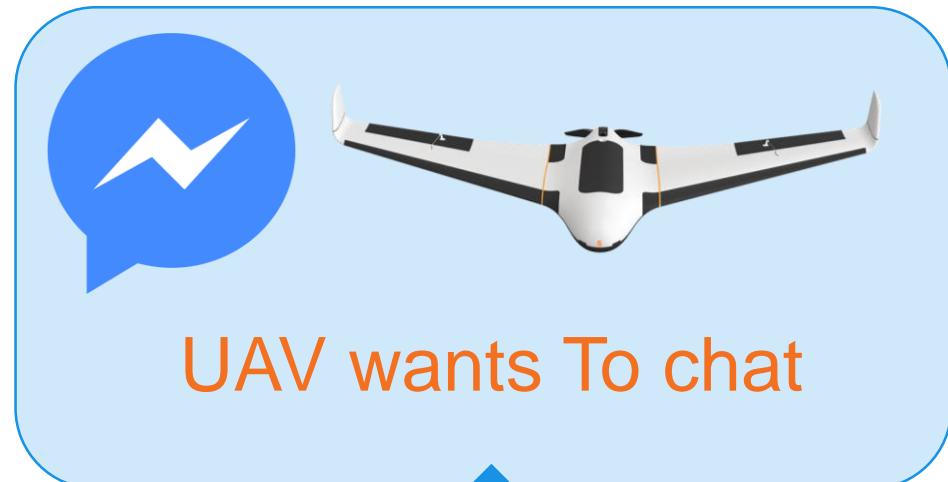
- You stick to written and unwritten rules during your commuting to PUB**
- Rules are necessary for successfull integration into traffic, society, air space**



After going all this challenges, you can finally enjoy that nice beer cola with friends...



Look at your phone



# A little chat with UAV...



Well Lojzo that is nice story, but how is it going to help me to avoid obstacles and execute mission?

You can use same principles of preemptive and reactive obstacle avoidance in your case



Slavic magic and Vodka ?



No, we can use Mathematics (Slavic magic) and my implementation skills (Coffee) to make you safe



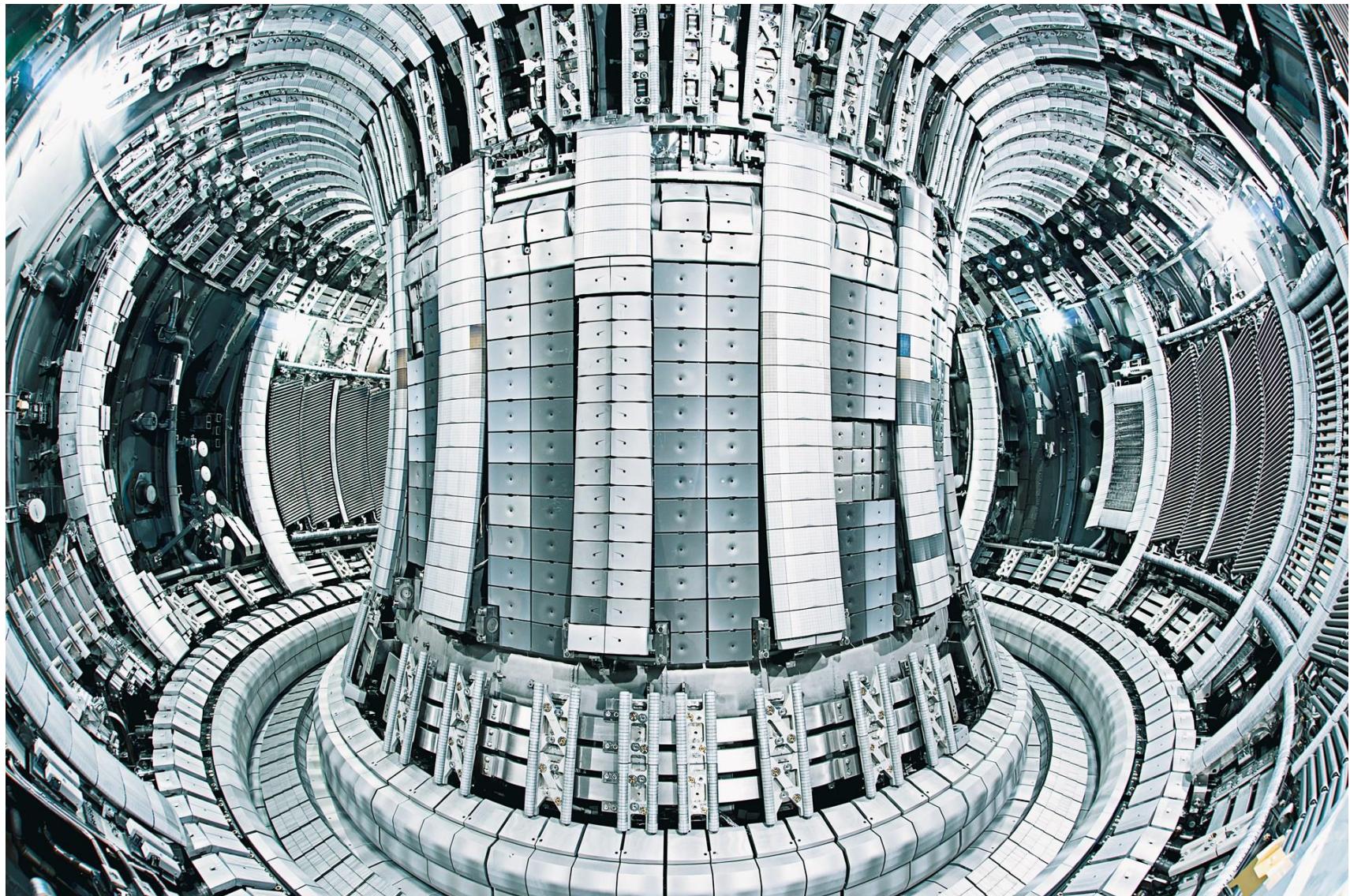
Like what ?



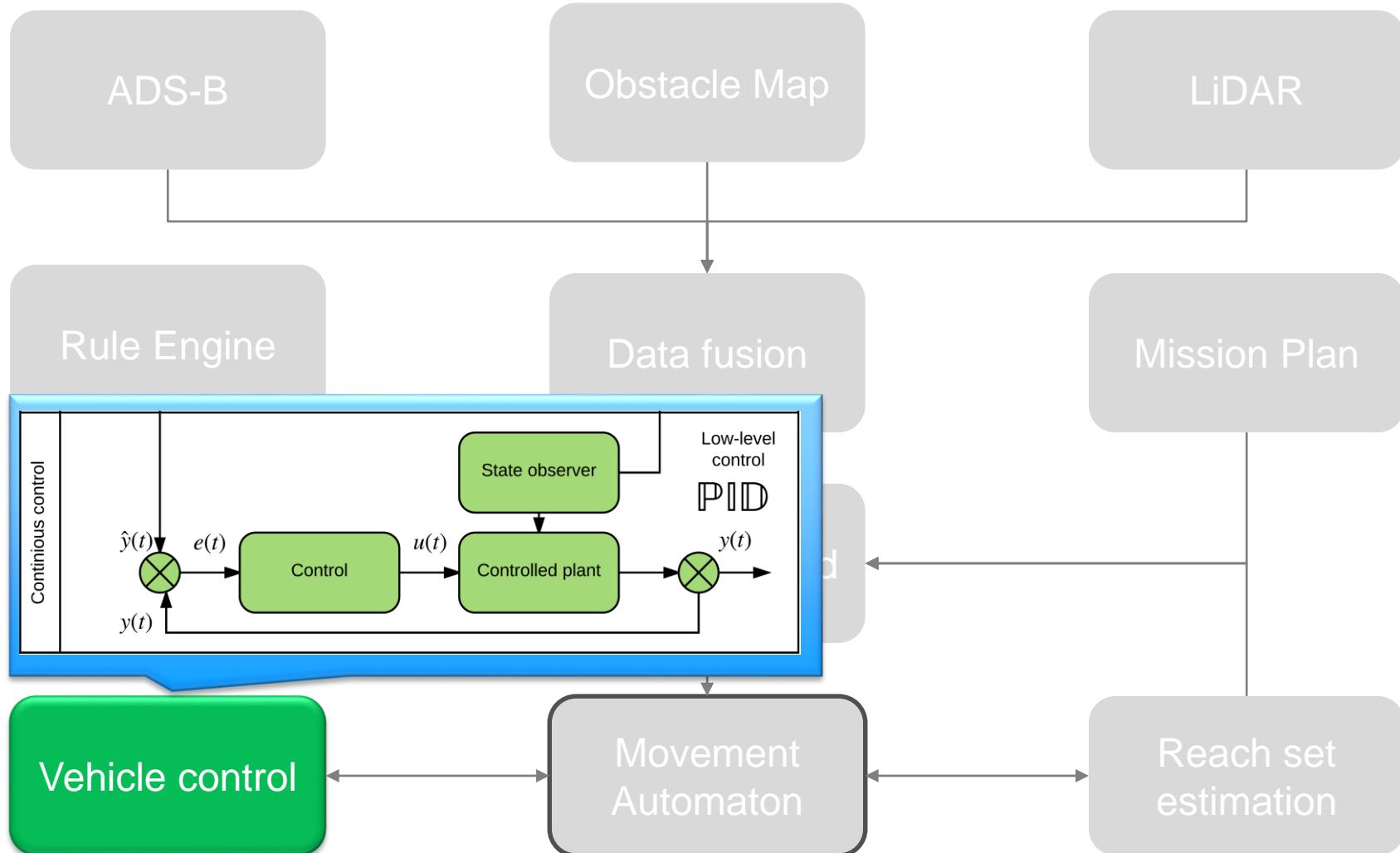
Some kind of elaborate Framework covering:  
 [1] Path planning, [2] Evolving world, [3] Adversarial, [4] Data fusion,  
 [5] Static restrictions, [6] Weather, [7] Rules



# Obstacle Avoidance Framework



# Vehicle control



# Vehicle control

You have some nice *vehicle model*:

The dynamic model of the *UAV* is given by a nonlinear first order state-space model:

$$\dot{x} = f(t, x, u) \quad (3)$$

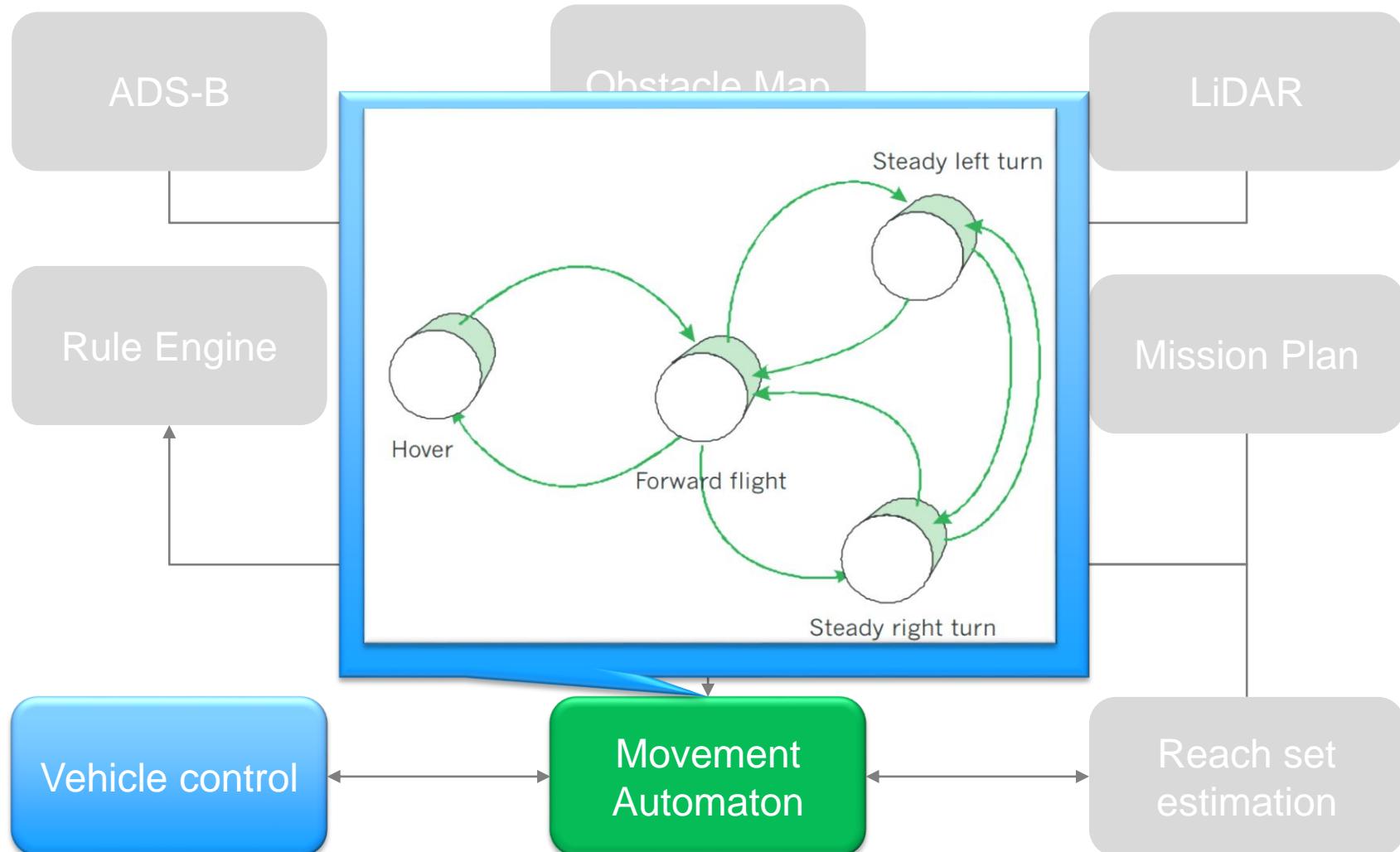
Where  $x \in \mathbb{R}^{6+n}$ ,  $n \in \mathbb{N}^{0+}$  is *system state* containing minimal information about vehicle position  $[x, y, z]$  and orientation  $[\theta, \varphi, \rho]$  (roll, pitch yaw), and  $u(t)$  is *control signal* belonging to  $\mathbb{R}^k$ ,  $k \in \mathbb{N}^+$ , bounded by control set  $u(t) \in U$ .

Then check *following conditions*:

- Observability of the system (Mandatory)
- Controllability of the system (Optional)
- Trajectory tracking control implementation (Optional)

Then you are ready to *construct movement automaton*

# Movement Automaton



# Movement Automaton, not so simple ...

*InitialState* :  $\in \mathbb{R}^h, h \in \mathbb{N}^+$

*System* :  $State = f(Time, State, Input)$  or *vectorField*

$$\text{Primitives} = \left\{ MovementPrimitive_i \begin{pmatrix} \text{vectorField}, \\ \text{minimalDuration}, \\ \text{parameters} \end{pmatrix} \right\} i \in \mathbb{N}^+$$

$$\text{Transitions} = \left\{ Transition_j \begin{pmatrix} MovementPrimitive_l, \\ MovementPrimitive_k \end{pmatrix}_{k \neq l} \right\} j \in N^+$$

$$\text{Movements} = \left\{ Movement_m \begin{bmatrix} Transition_o[0..*], \\ MovementPrimitive_p \\ Transition_r[0..*], \end{bmatrix}_{o \neq r} \right\} m \in N^+$$

$$\text{Buffer} = \{Movement_s(duration_s, parameters_s)\} s \in \mathbb{N}^+$$

$$\text{Executed} = \{Movement_s(duration_s, parameters_t)\} t \in \mathbb{N}^+$$

*Builder* :  $Movement \times MovementPrimitive \rightarrow Movement$

*Trajectory* :  $InitialState \times Movements^u \rightarrow State \times Time, u \in N^+$

*StateProjection* :  $Trajectory \times Time \rightarrow State(Time)$

# Movement Automaton, simple

I want to interface control law  $u(t)$  from function to *discrete command chain:*

$$\mathcal{MA}(\text{command sequence}) \rightarrow u(t)$$

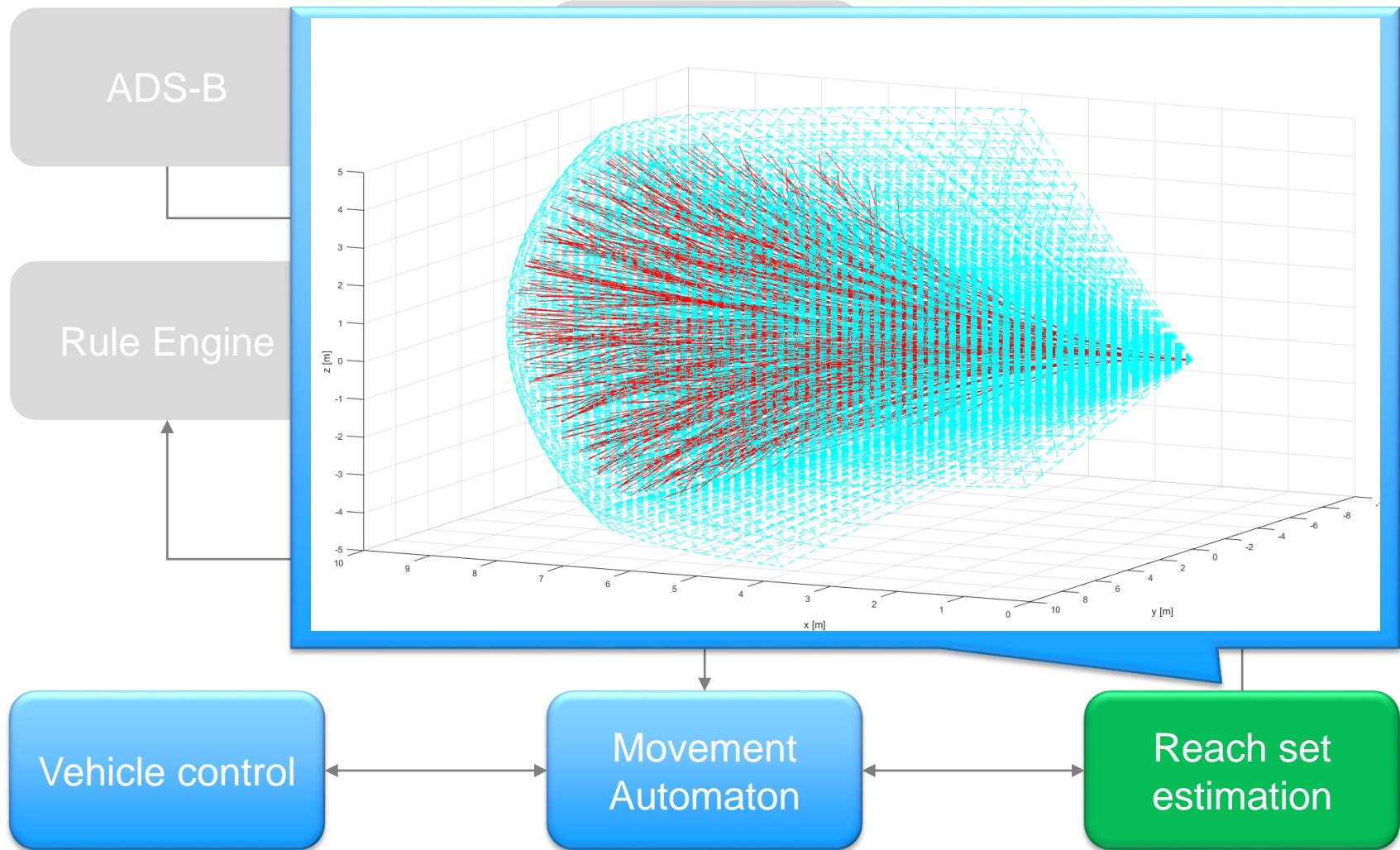
*Movement Automaton*  $\mathcal{MA}$  translates command sequence consisting from movements with applied duration to generate control signal  $u(t)$  or reference trajectory  $x(t)$

## References:

**[Concept]** FRAZZOLI, Emilio. *Robust hybrid control for autonomous vehicle motion planning*. 2001. PhD Thesis. Massachusetts Institute of Technology.

**[Stability, Controllability, Operation Modes]** Alojz Gomola, João Borges de Sousa, Fernando Lobo Pereira, Model Predictive Control of Unmanned Air Vehicles with Obstacle Avoidance Capabilities, FEUP 2016, Exam report,

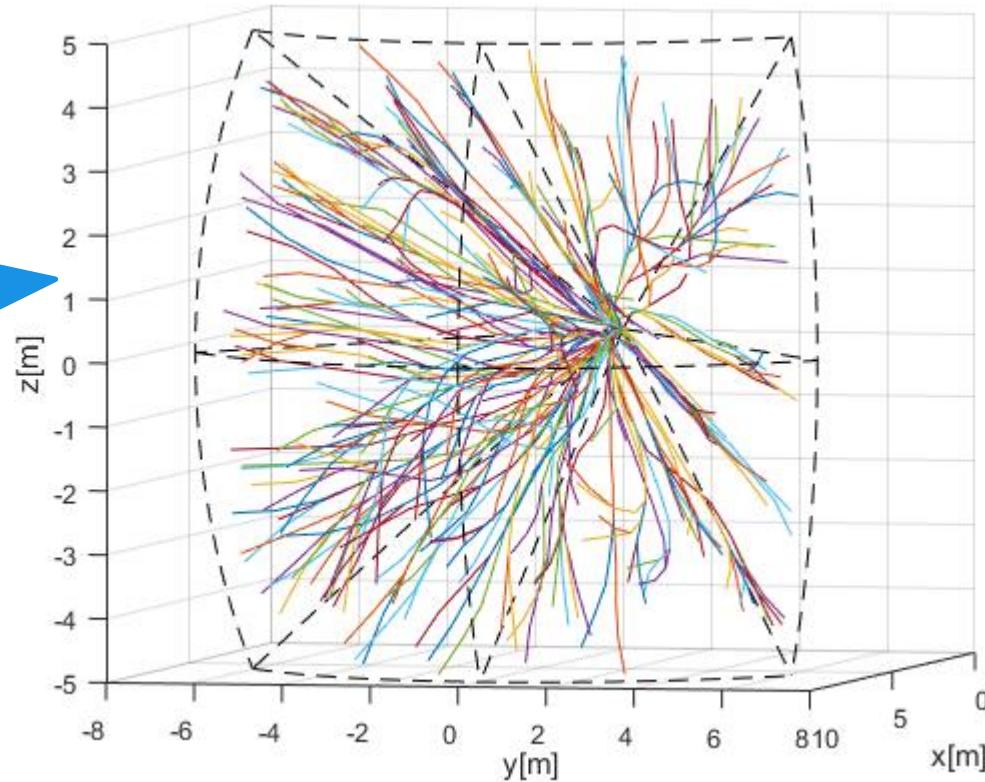
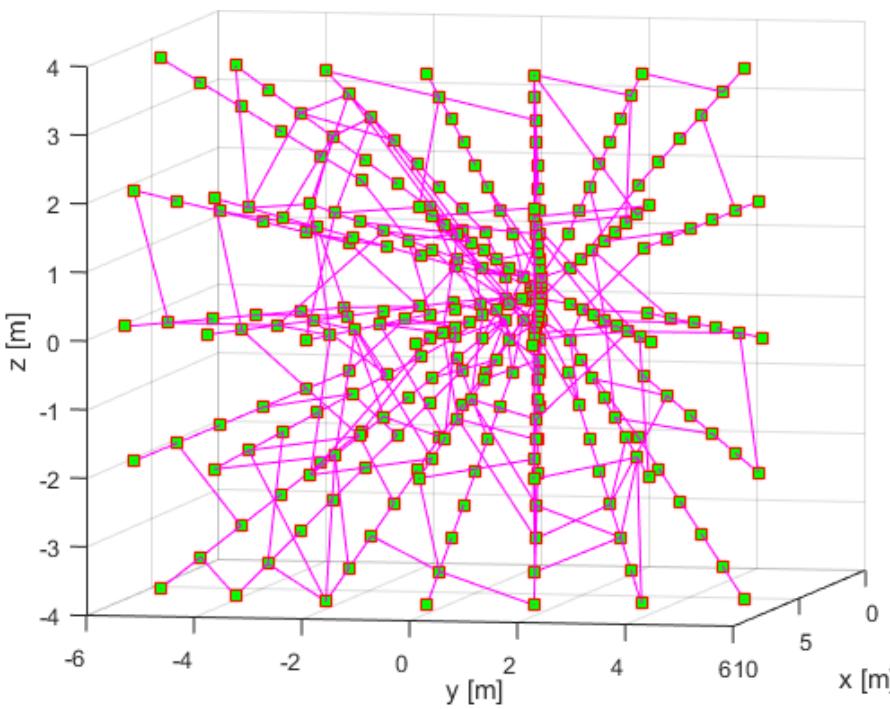
# Reach set estimation



# Reach set -Trajectory set & Graph

Trajectory set approximation:

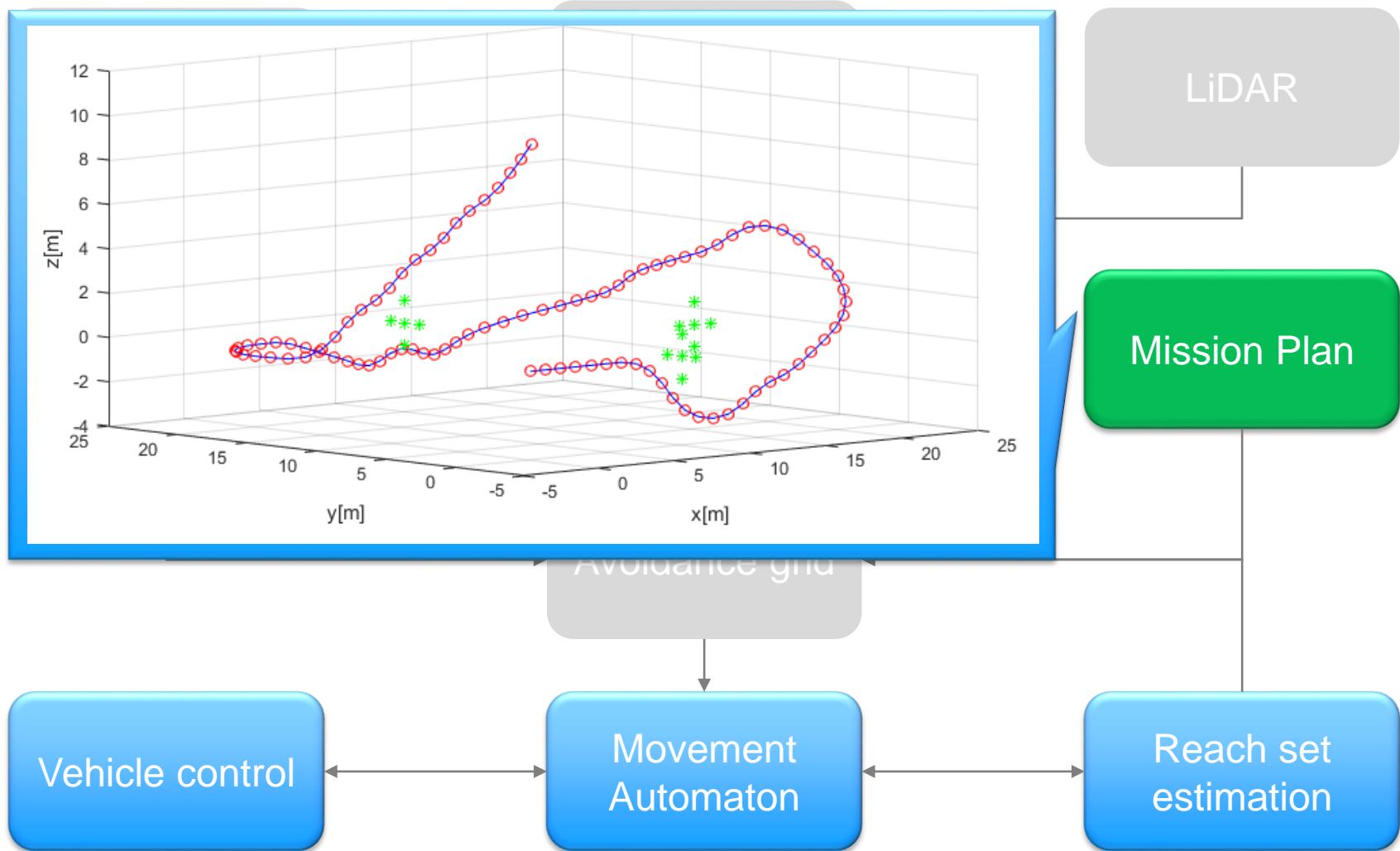
- 1350 movements
- 270 paths



Graph approximation:

- 351 nodes
- 270 paths

# Mission plan



# Some mathematical background:

## What you need ?

- Define mission as sequence of waypoints
- Define waypoint passing condition to evaluate performance,

*Mission* (6) which UAV should fly is given as set of *ordered, feasible in terms of vehicle dynamic , waypoints* in subspace of *Space*.

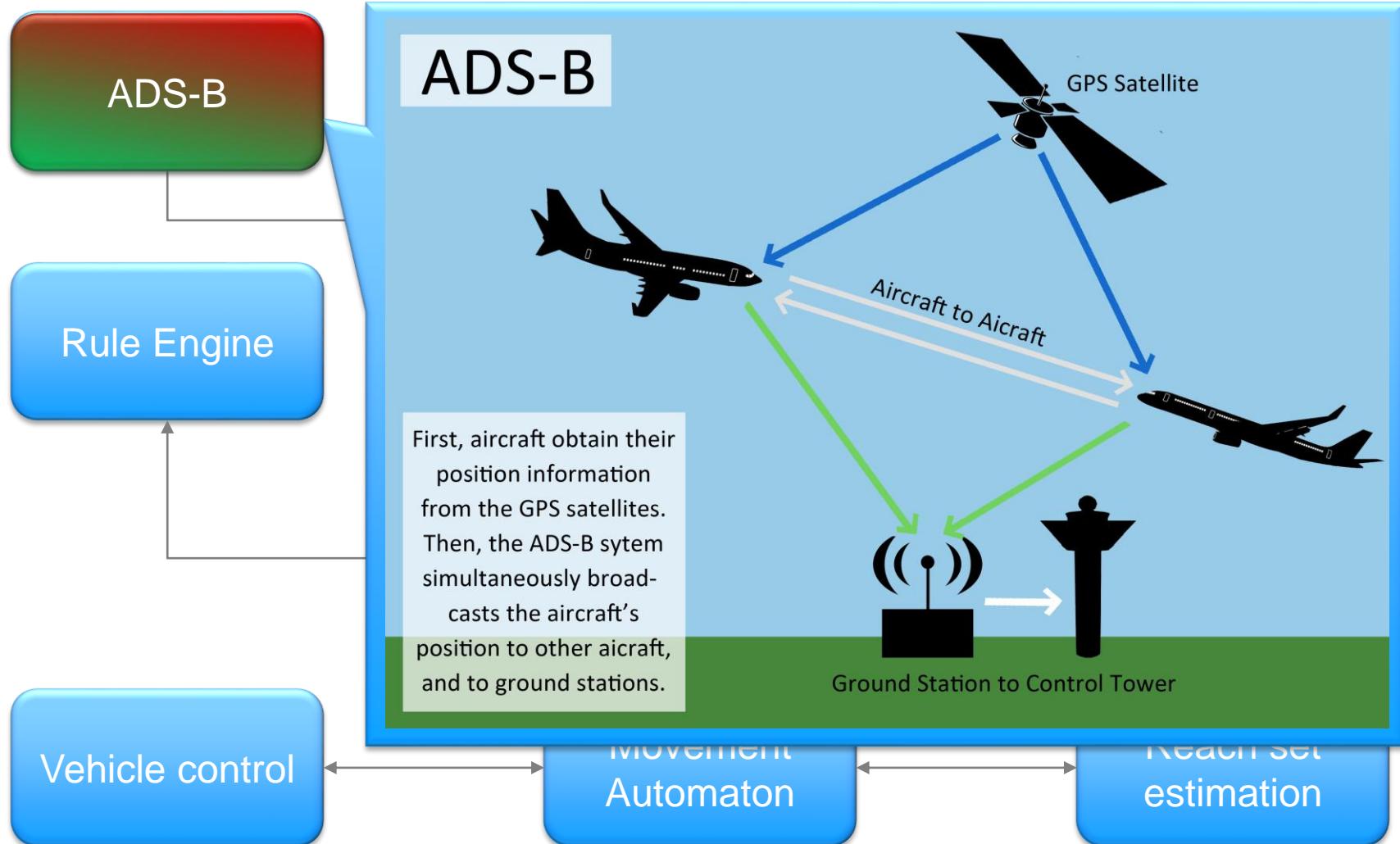
$$\text{Mission} = \left\{ \begin{array}{l} \text{waypoint}_1, \text{waypoint}_2, \dots, \text{waypoint}_m : \\ \forall_{i=1 \dots m} \text{waypoint}_i \in \text{Space} \end{array} \right\}, \quad m \in \mathbb{N}^+, m \geq 2 \quad (6)$$

*Waypoint Passing* (7) function maps system trajectory *projection in Space* and *Mission* waypoints (6) to a vector of *passing times*.

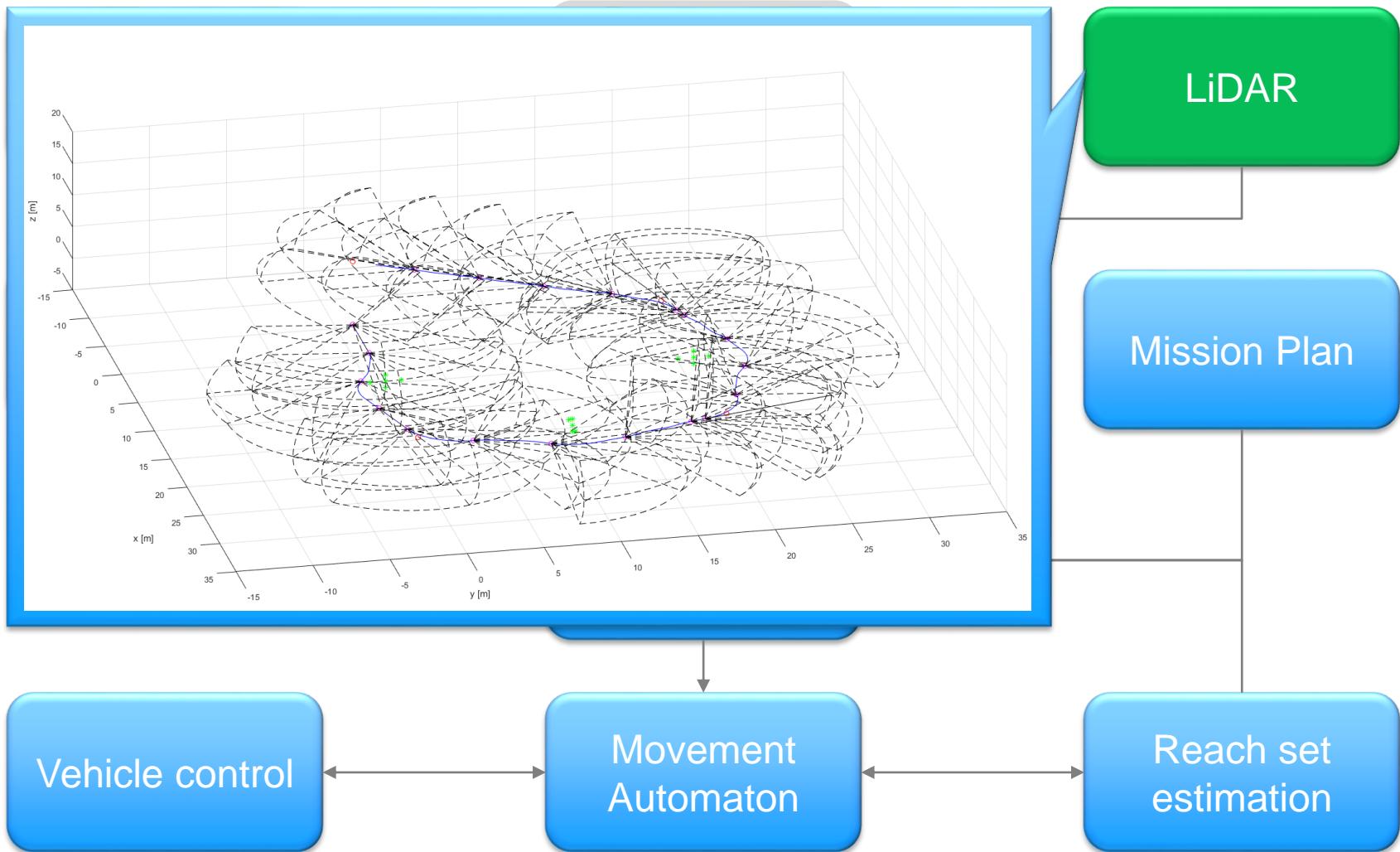
$$\text{WaypointPassing} : \text{TrajectoryProjection} \times \text{Mission} \rightarrow \text{Time}^m \quad (7)$$

*Note.* The *Mission* (6) is considered as successfully completed if and only if  $\forall$  waypoints are reached and in given order (check output of 7).

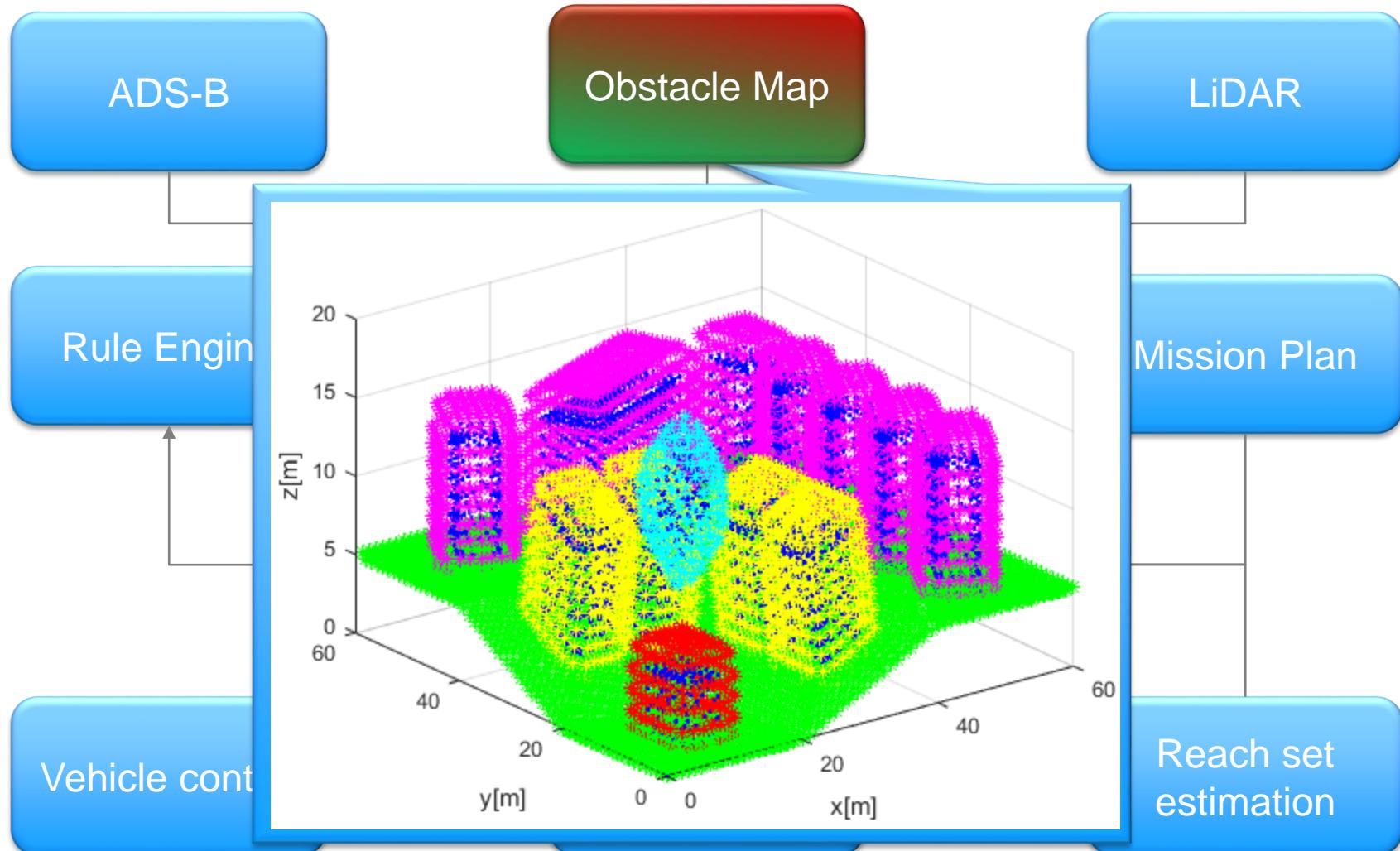
# ADS-B

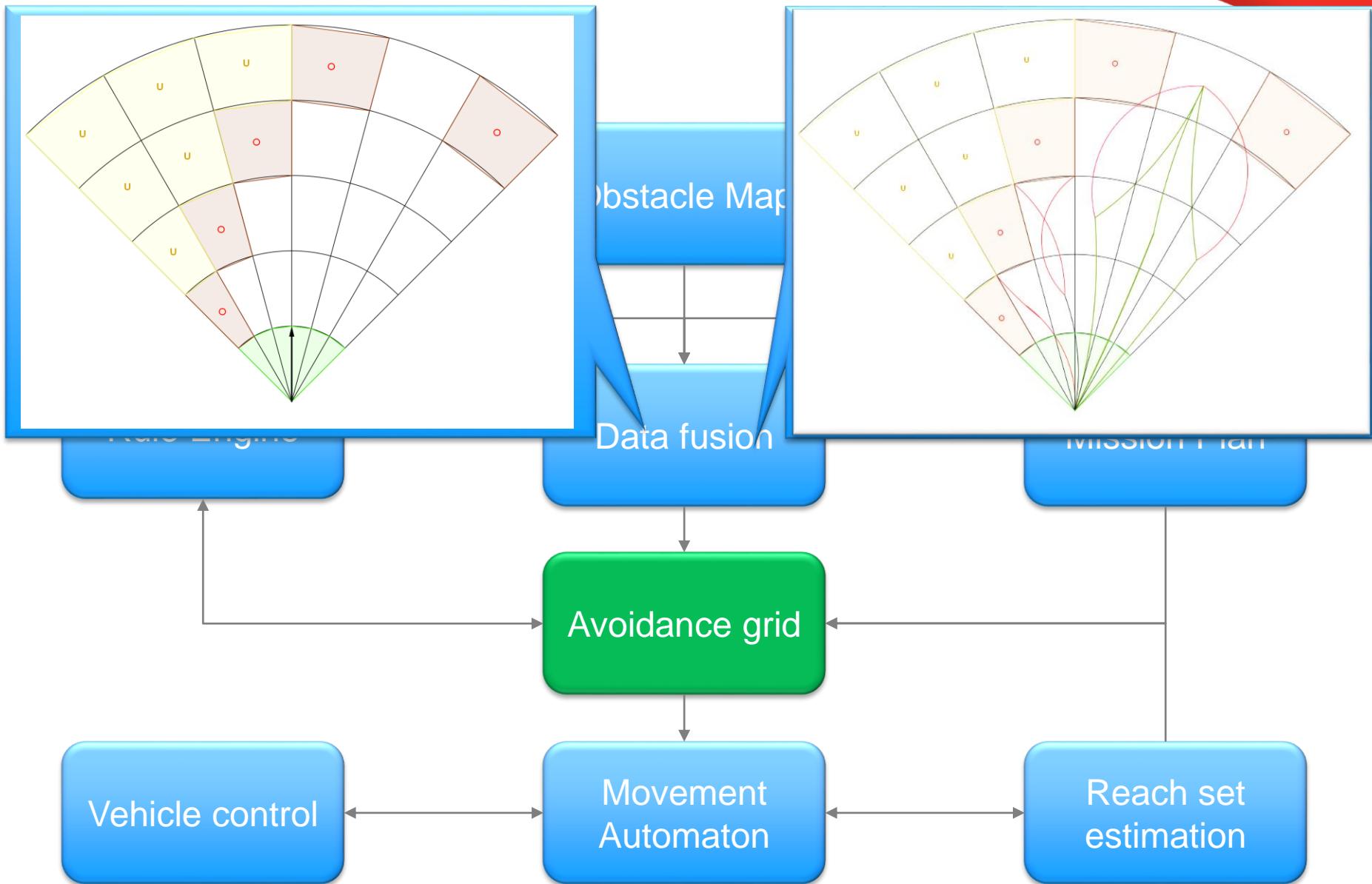


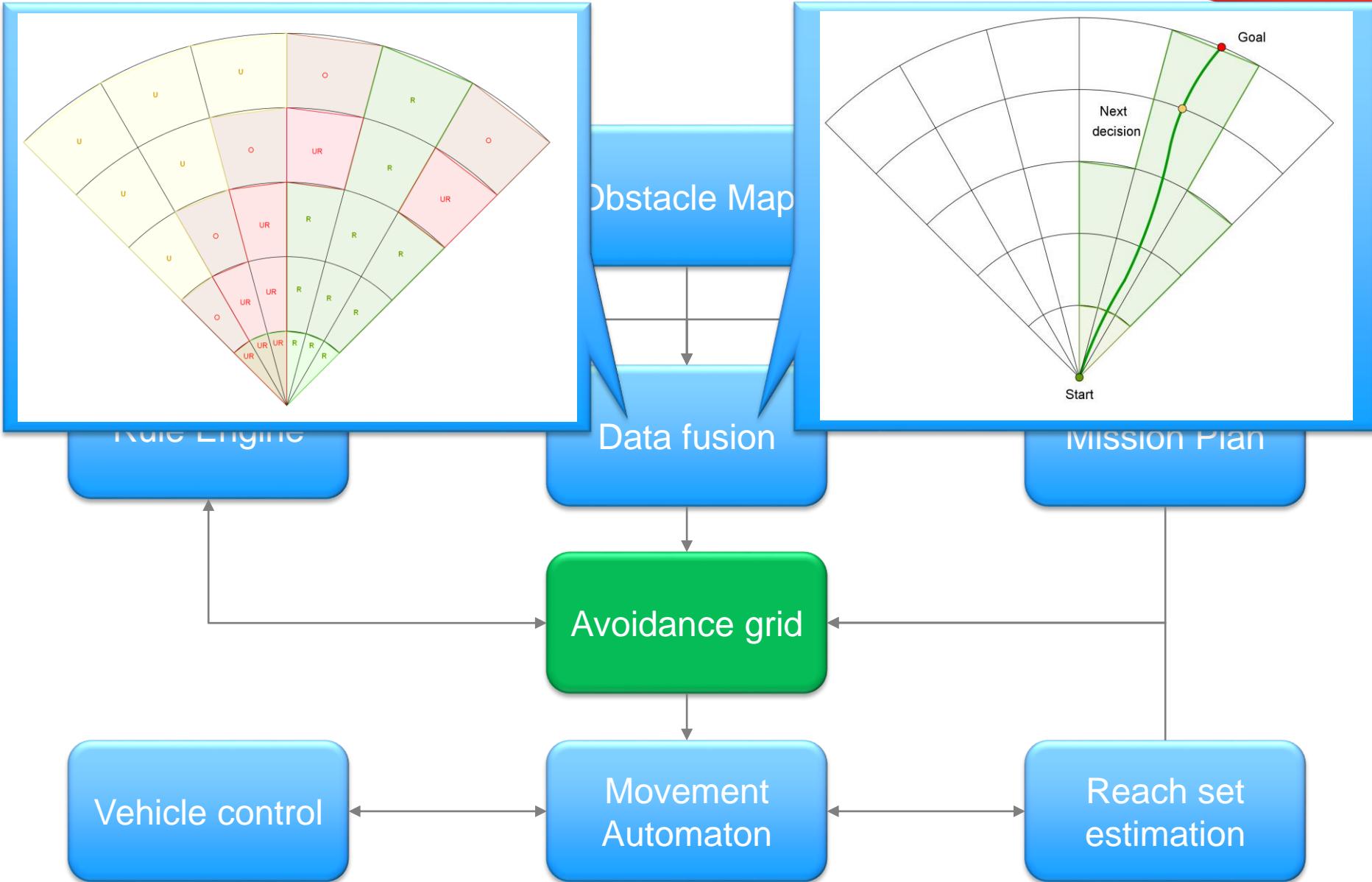
# LiDAR



# Obstacle Map







# Conditions

## Avoidance grid sizing (Distance/Horizontal range/Vertical range)

1. Distance should not be greater than combined median range of sensors
2. Horizontal/Vertical range is bounded by:
  1. Sensor field median ranges from observation point
  2. Reach set boundary for selected distance
3. The minimal avoidance grid size is given as:

$$\text{Distance} = \text{DecisionMaximumTime} \times \text{VehicleMaximumVelocity}$$

$$\text{HorizontalRange} = \pm \text{MaxHorizontalDeviation}(\text{ReachSet})$$

$$\text{VerticalRange} = \pm \text{MaxVerticalDeviation}(\text{ReachSet})$$

## Cell size (Length, Horizontal step, Vertical step):

1. Length ~ average length of executed movement in given layer
2. Horizontal/Vertical step ~ average deviation of movements in given cells

$$\text{Length} = \text{Average}(\text{Length}(\text{movementExecution}))$$

$$\text{HorizontalStep} = \text{Average}(\text{Deviation}(\text{Horizontal}(\text{movementExecution})))$$

$$\text{VerticalStep} = \text{Average}(\text{Deviation}(\text{Vertical}(\text{movementExecution})))$$

# Example: Grid Scaling

*Decision point 10m  
LiDAR range 100m  $\Rightarrow$  Range: 10m*

*Calculate ReachSet  $\Rightarrow$  Lines in Picture*

$$\text{MaxHorizontalDeviation} = \frac{\pi}{4} \text{ [rad]}$$

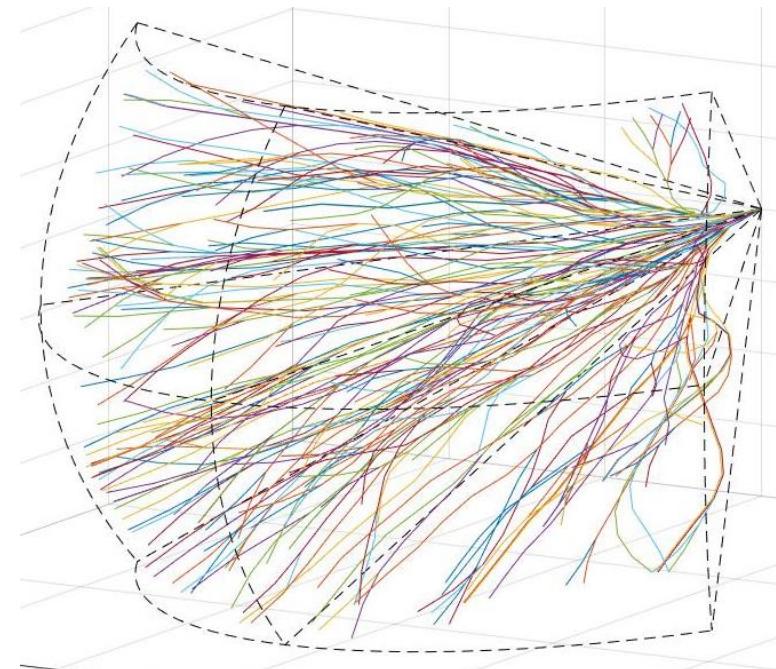
$$\text{MaxVerticalDeviation} = \frac{\pi}{6} \text{ [rad]}$$

## Avoidance Grid boundary:

Distance: 10m

$$\text{Horizontal range: } \left( -\frac{\pi}{4}, \frac{\pi}{4} \right)$$

$$\text{Vertical range: } \left( -\frac{\pi}{6}, \frac{\pi}{6} \right)$$



# Framework Summary

You should check my article: GOMOLA, Alojz, et al. Obstacle Avoidance Framework Based on Reach Sets. In: Iberian Robotics conference. Springer, Cham, 2017. p. 768-779.



## Key notes:

- Movement Automaton – interface between control and avoidance grid,
- Sensor/Data Fusion – interface between sensors/information sources and avoidance grid,
- Reach set – tells UAV where it can go from initial position with given maneuvering capabilities from Movement Automaton
- Avoidance grid – combines known world from Sensor/Data Fusion with Reach set for current vehicle state and applies Rules to find best escape trajectory in Field of Vision,
- Rule engine – contains set of Rules,

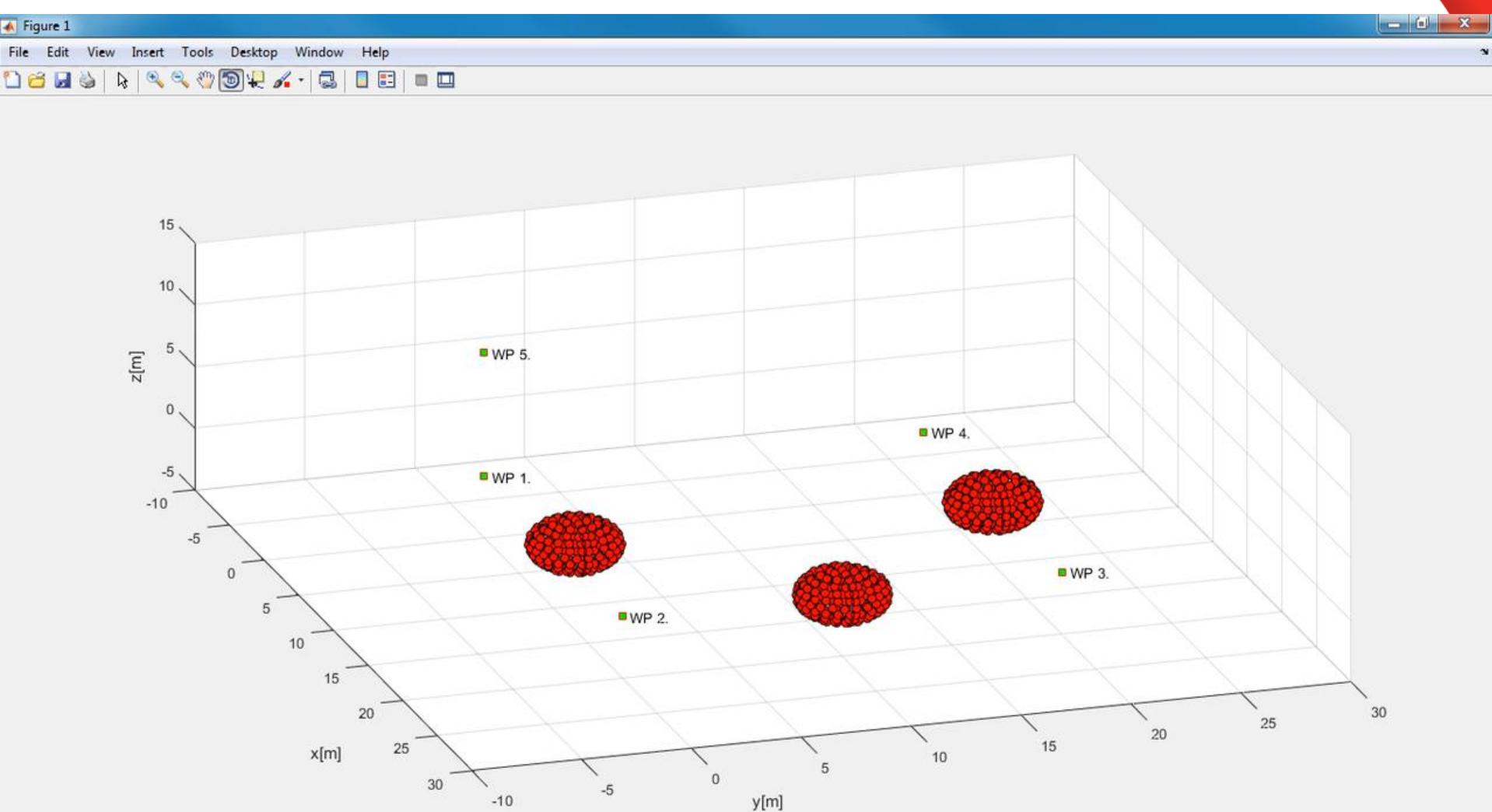
## Addressed issues:

- [1] Path planning, [2] Evolving world, [3] Adversarial, [4] Data fusion

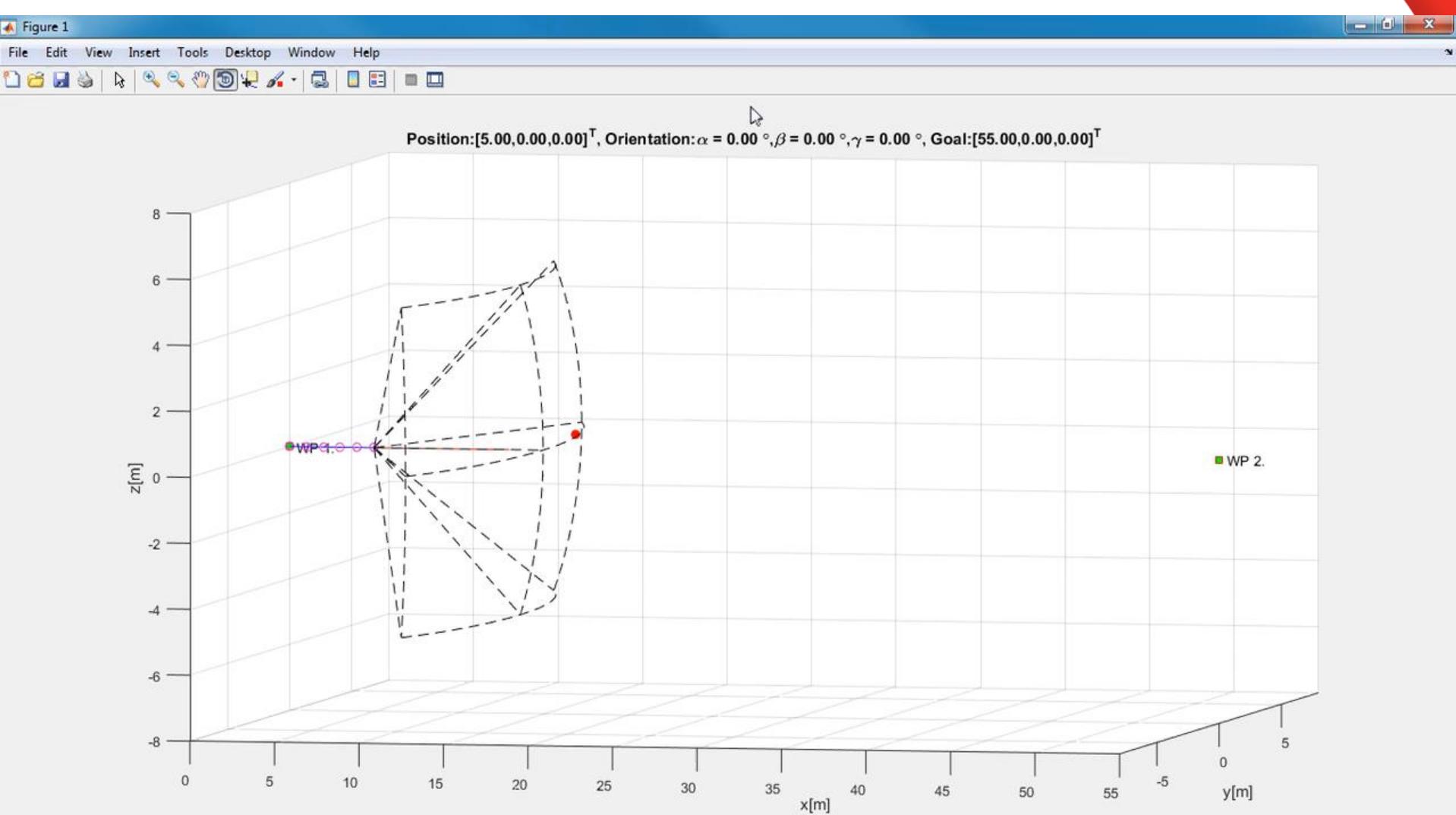
## Some Results ...



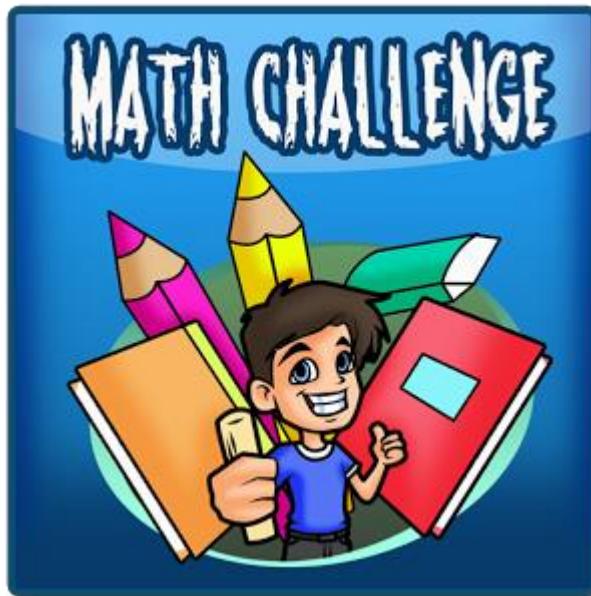
# Static obstacle avoidance



# Intruders avoidance



# Actual Challenges



## Weather ATC



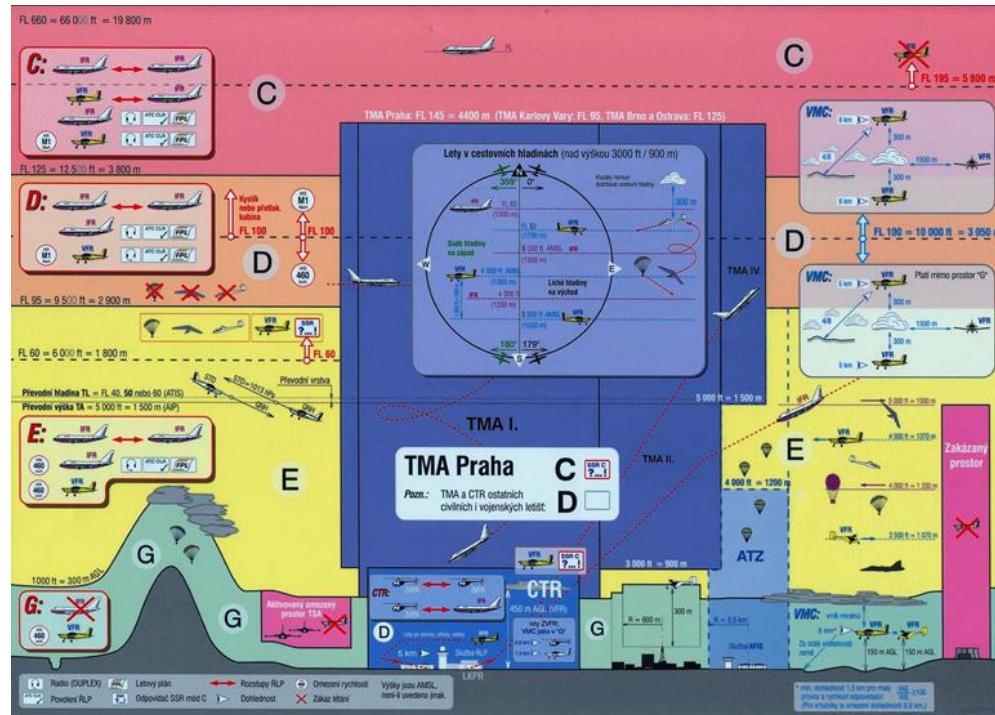
# Airspace classification

We want to Fly in *Non-Segregated Airspace*

Space is separated into levels defined by boundaries:

Level C-F – altitude in Flight Levels (FL)

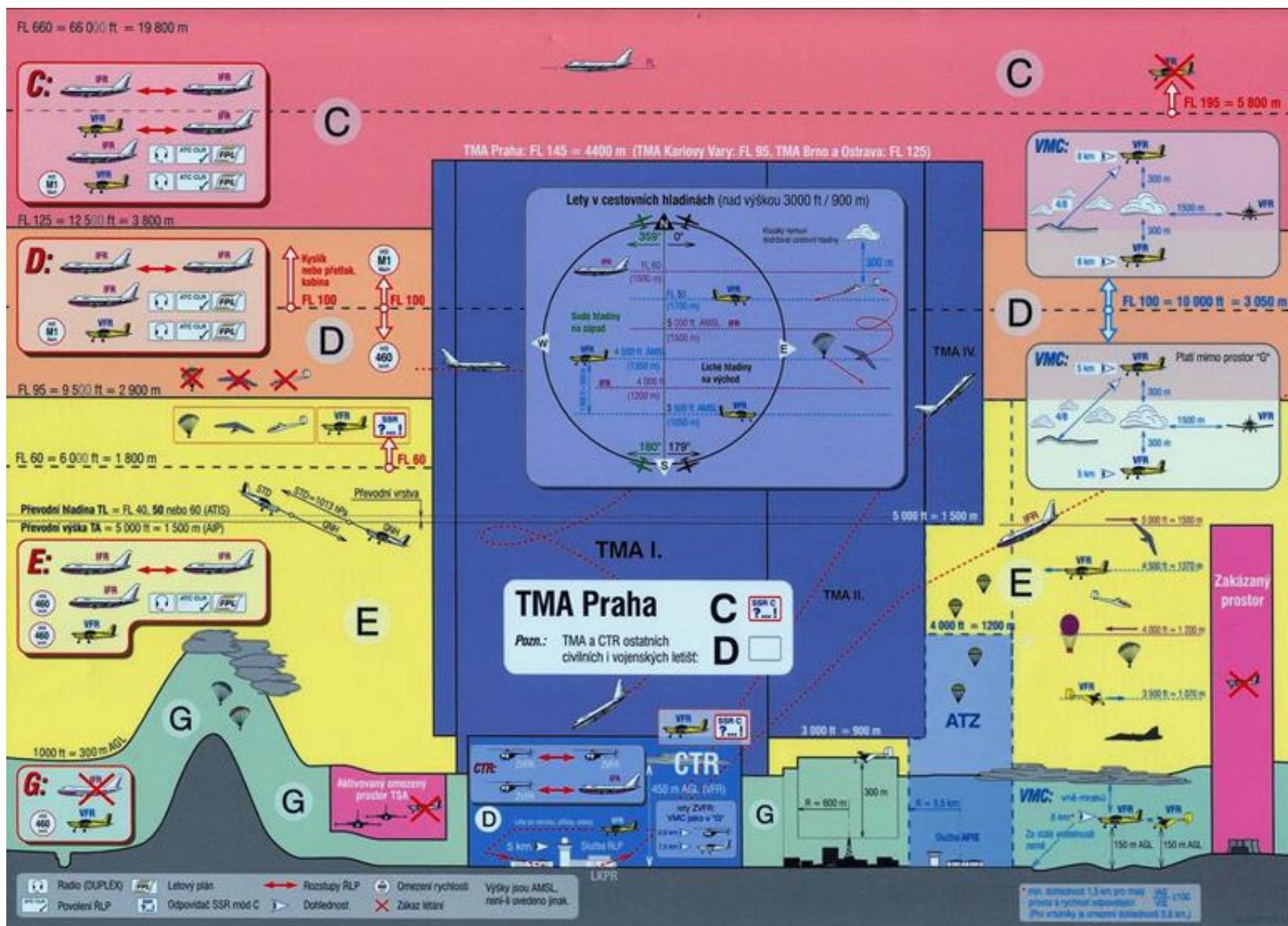
Level G – altitude in Above Ground Level (AGL)



Example of European  
Airspace classification  
(Czech republic)

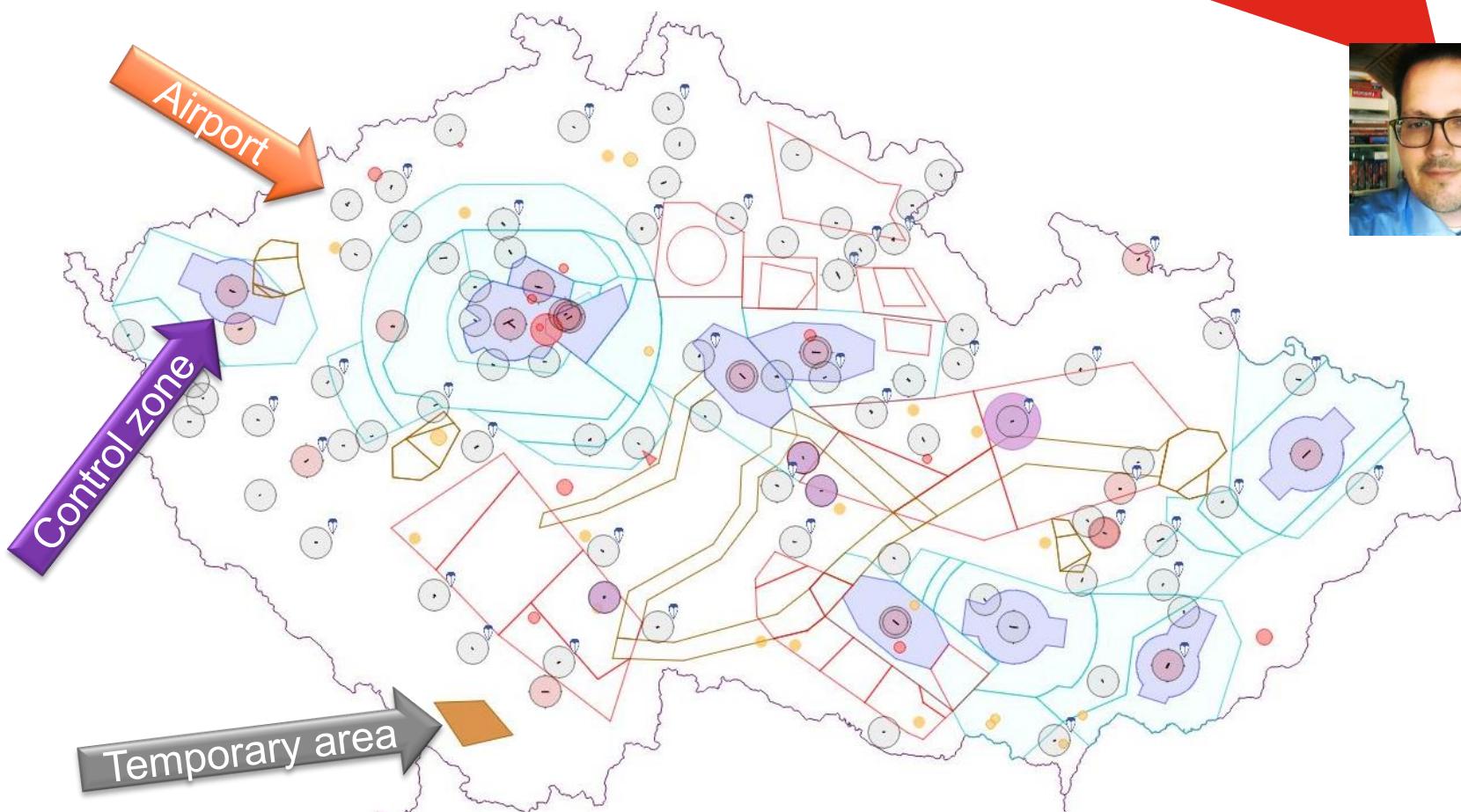
Next slide

# Airspace classification (Detail)



# ATC constraints – Czech Republic

Example of air traffic constraints is taken from Czech national air traffic control portal (<http://aisview.rlp.cz/>)



# Weather constraints

- UAV system in general is more sensitive to weather conditions than standard manned aviation (icing, storms, turbulences)
- Clouds can generally render UAV sensing system
- Especially thick fog renders LiDAR sensor range tremendously
- Modern manned aviation system can provide data necessary for weather situations avoidance



Example of advanced weather warning system developed by Honeywell.  
Extracted data will be used as a source of restrictions for UAV

# Intruder avoidance priority

Our vision is based on various projects:

- Amazon Drone platform – fully autonomous UAV
- **NASA – UTM** (UAV Traffic Management) – rules in development
- JARUS – SAA regulation framework (in development)

The rules of the air are formulated (ICAO annex II.) around avoidance priority, depending on technical capabilities of manned aircraft type (Jumbo jet should avoid glider due higher maneuverability).

The expected behavior of UAV platform is given by following table:

Intruder	Our vehicle	Priority
Manned aircraft	Any UAV	Manned aircraft
VLOS piloted UAV	VLOS piloted UAV	Coordinated avoidance
VLOS piloted UAV	Autonomous UAV	Autonomous UAV
Autonomous UAV	Autonomous UAV	Coordinated avoidance

# Aviation classes for immediate avoidance

**FIRST**: Manned aviation in distress



Mayday, mayday, mayday!

For more Information: ICAO ANNEX II.

**SECOND** Balloons



**THIRD** Gliders

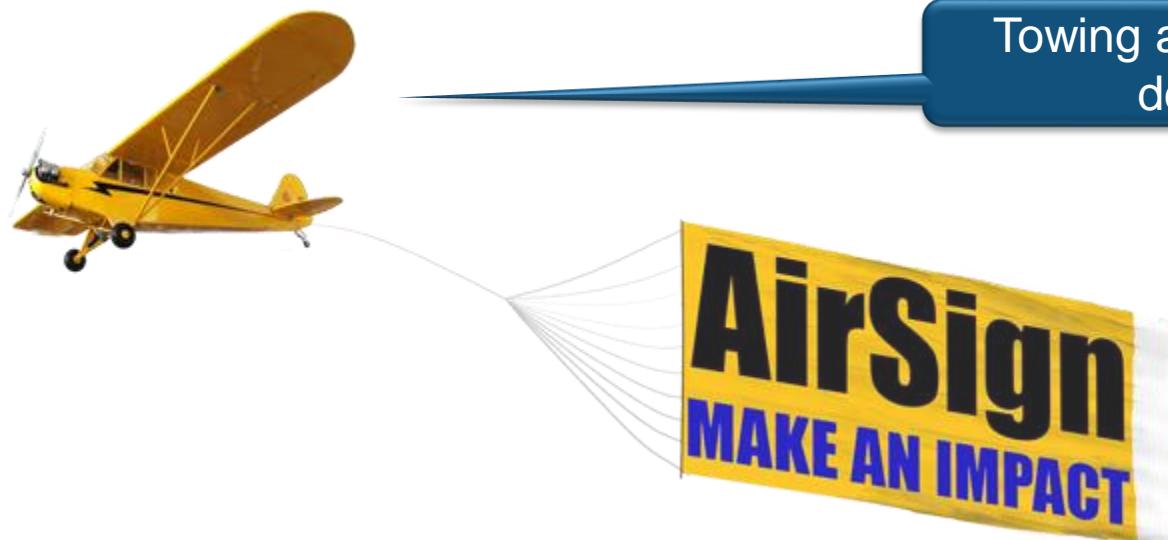


# Aviation classes for immediate avoidance

## **FOURTH:** Aerial fueling and towing



Fueling of aircraft



Towing a sign, aircraft or other dependable load

# Aviation classes for immediate avoidance

## **FIFTH:** Airships



But Lojzo, I am not on  
this list...

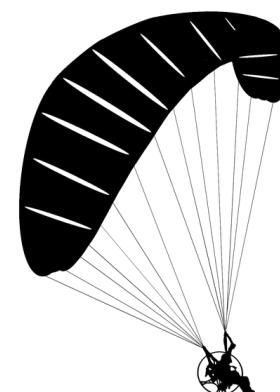
You are here on the bottom of  
course



Pilots don't like you, you need to  
avoid all manned aviation

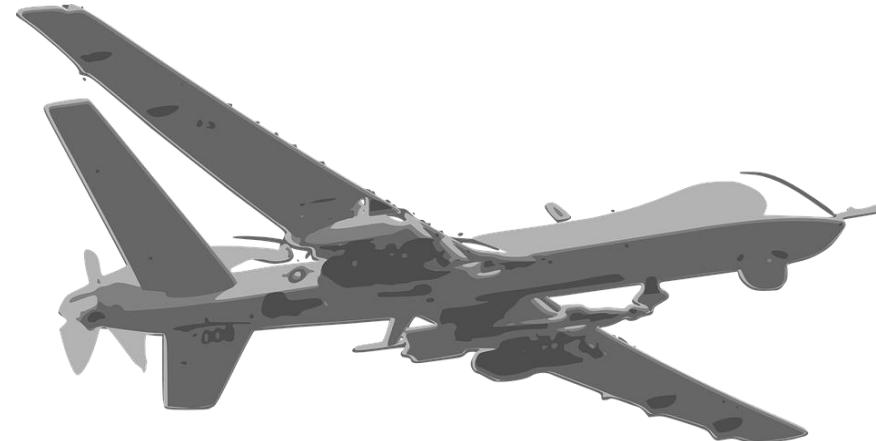


## **Sixth:** Manned aviation with propeller



# Proposed unmanned aviation classes

**Seventh:** Unmanned aviation with standardized avoidance system



Aww thanks buddy !

No problem mate!



That was sarcastic...

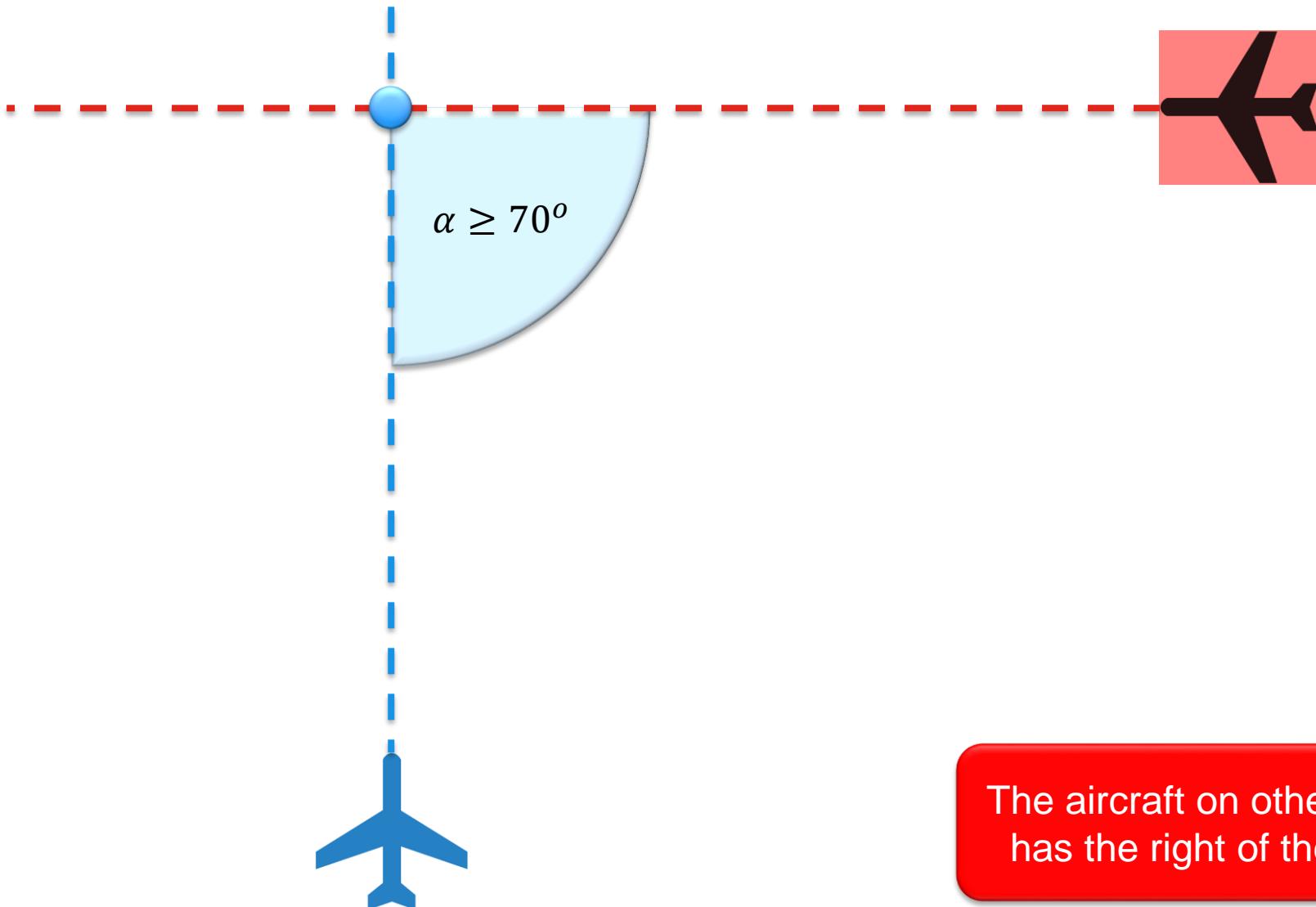
**Eight:** Unmanned aviation for VLOS operations piloted from ground



# Right of the way rules

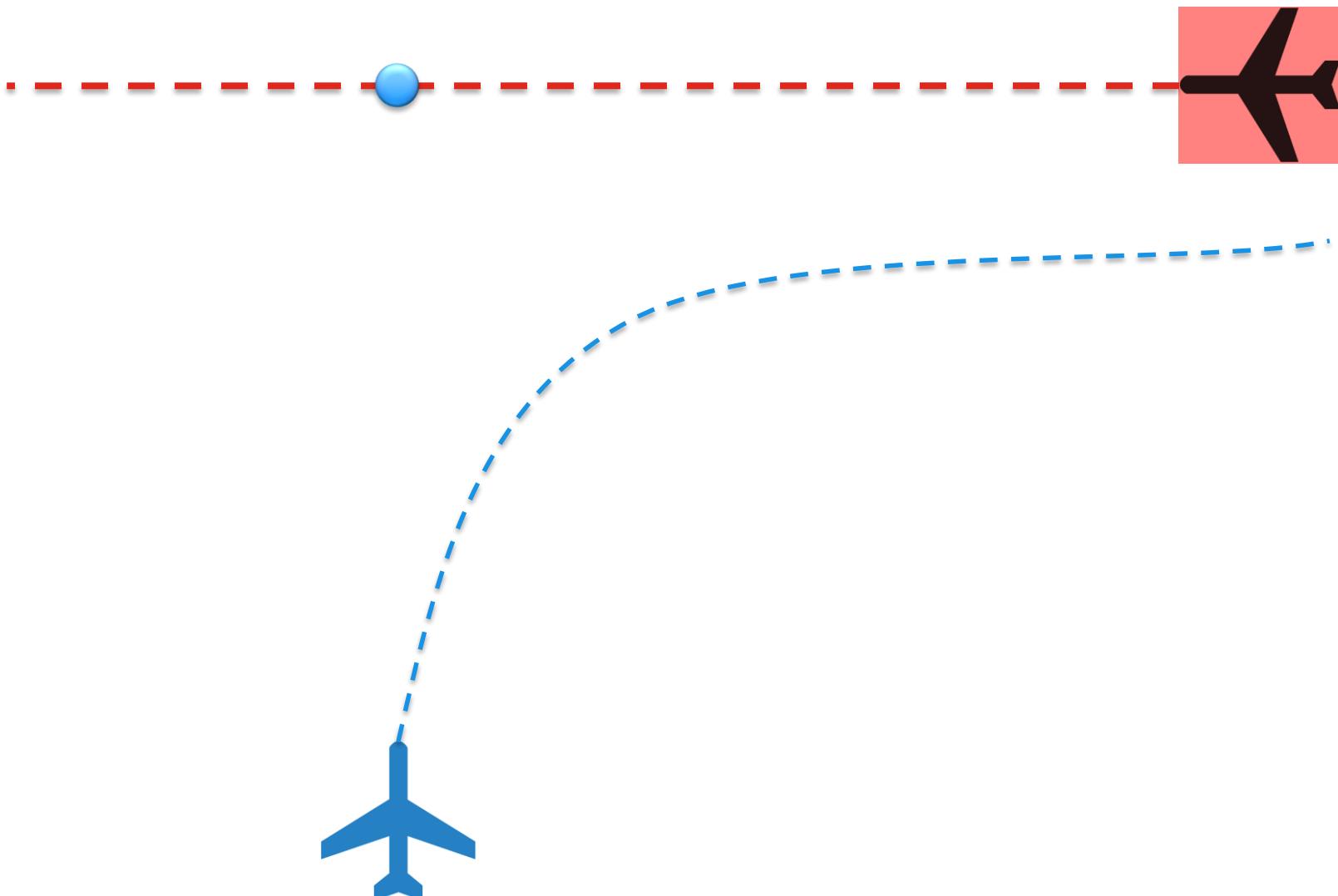
- Right of the way rules aid pilots in avoiding each other visually
- The rules are found:
  - Part 91 Federal Aviation Regulation (USA)
  - ICAO Annex 2 -> Chapter 3 - General Rules -> 3.2 Avoidance of collisions -> 3.2.2 Right of way (International)
- When aircrafts are converging, the right of way is generally given to the least maneuverable aircraft (higher class)
- The aircraft now having the right of the way is to pass well clear of the aircraft not having the right of the way
- Application for rules of the air in case of UAV seems to be reduced to immediate avoidance in case of manned aviation adversary
- The avoidance rules can be applied between the same class of UAV

# Converging aircrafts of the same category

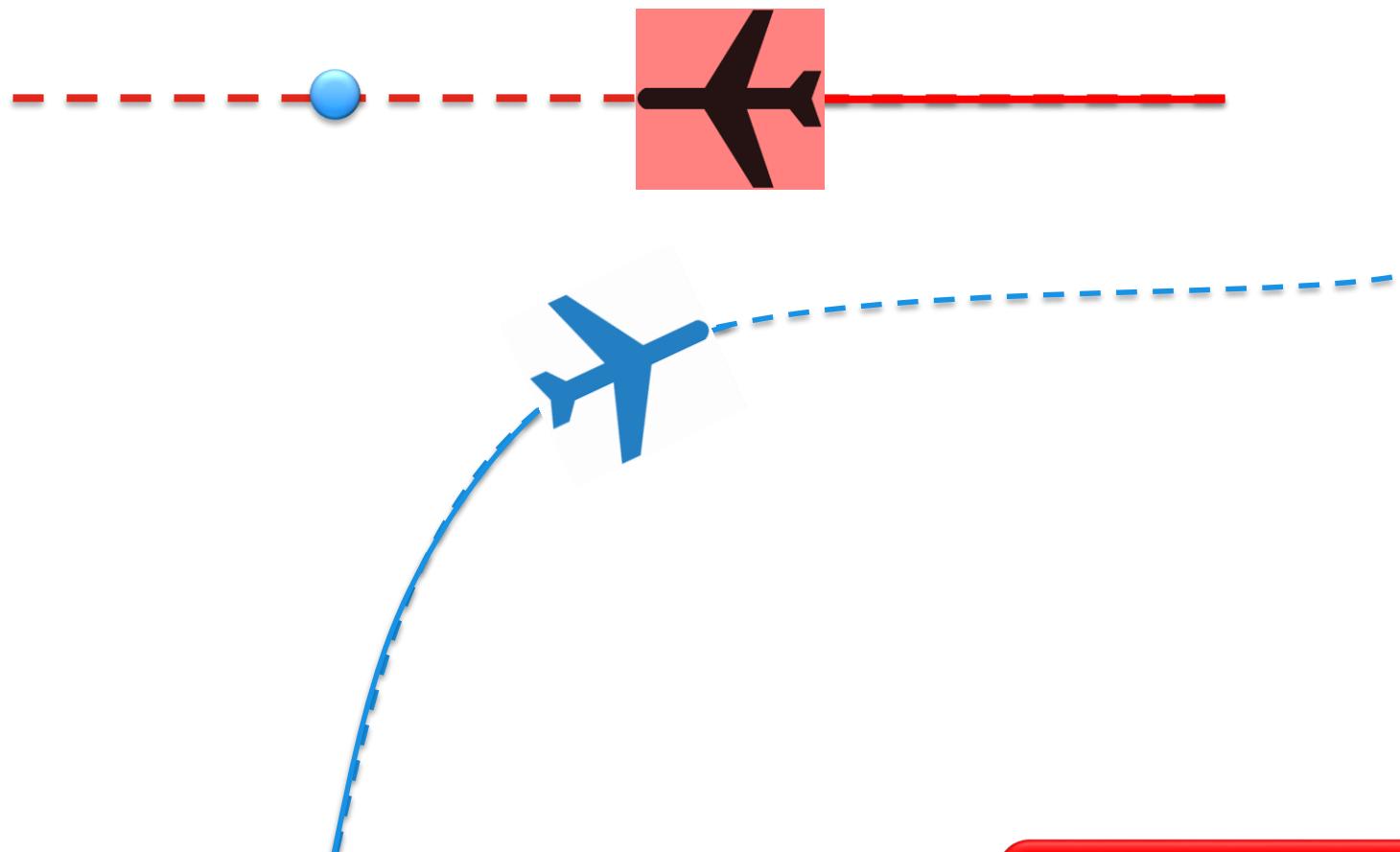


The aircraft on other right,  
has the right of the way

# Converging aircrafts of the same category

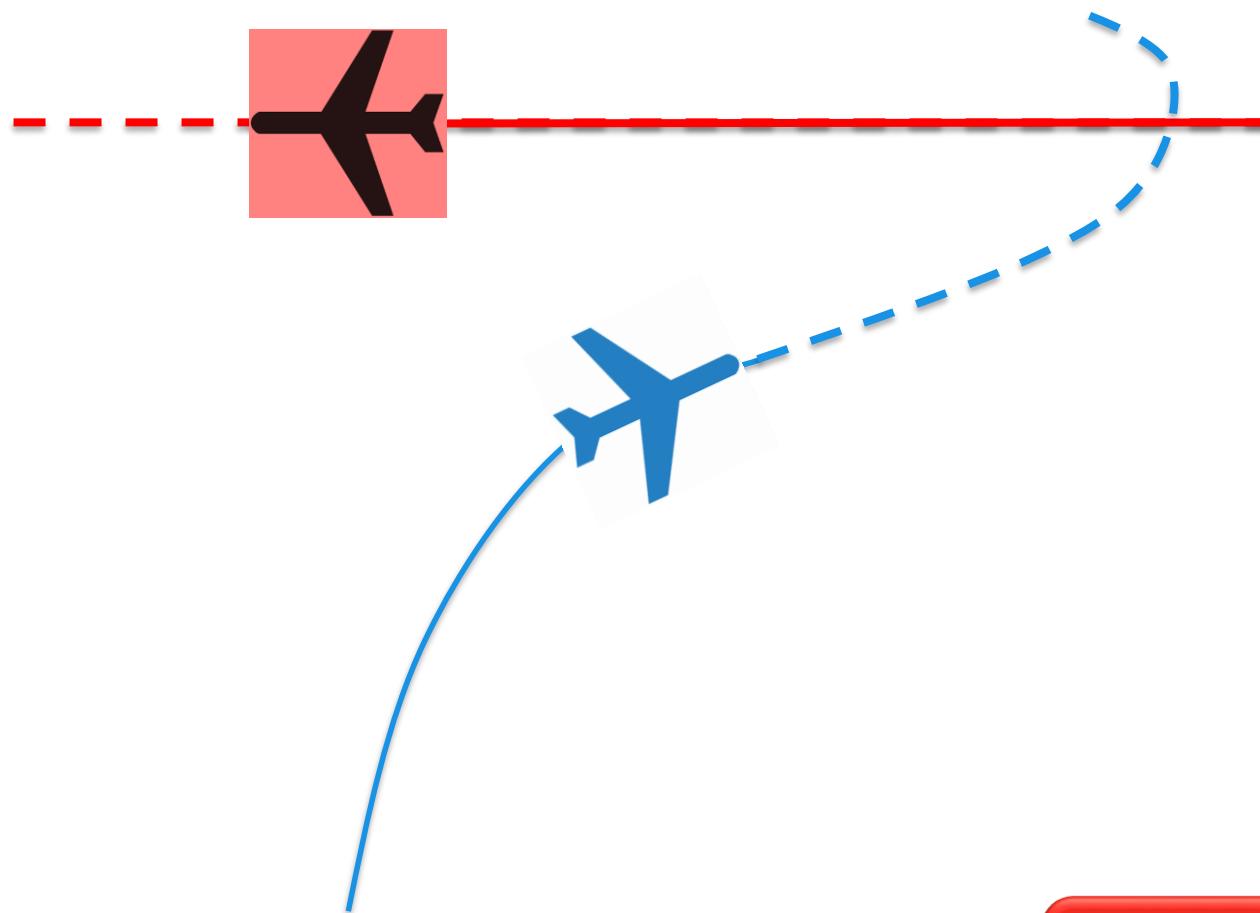


# Converging aircrafts of the same category



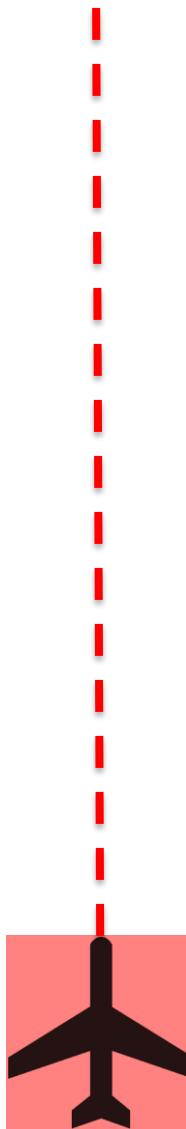
Turn right and stay away  
from approaching aircraft

# Converging aircrafts of the same category



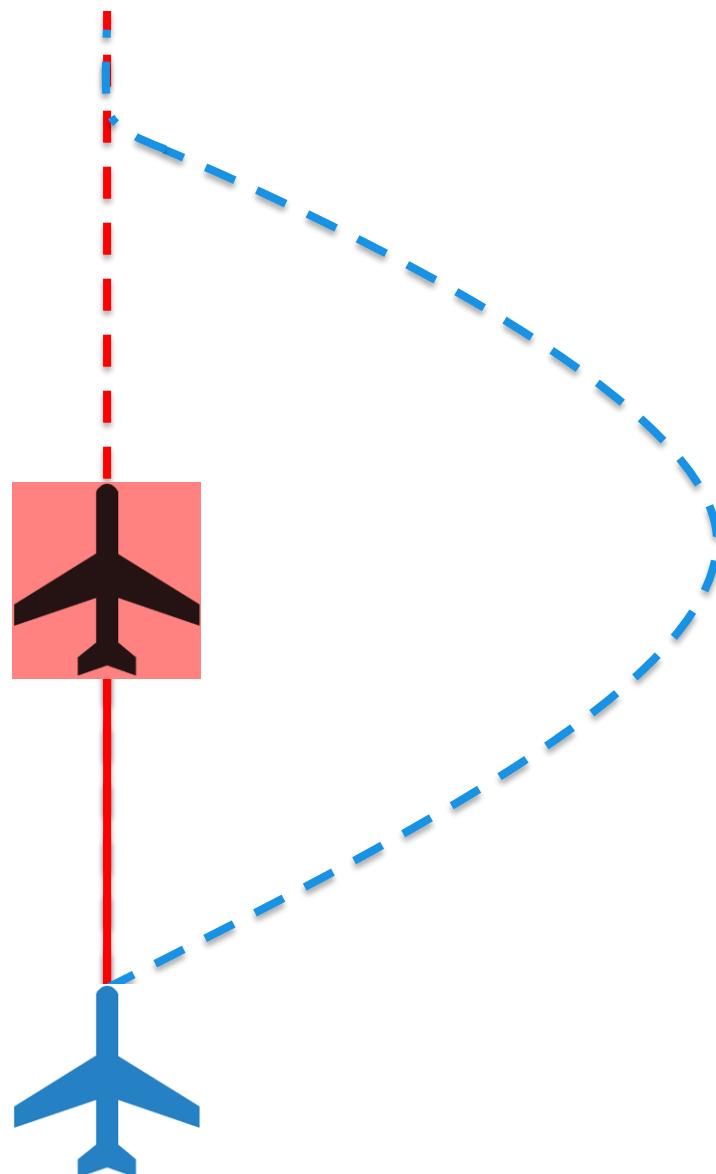
Return to original path  
behind other aircraft

# Overtaking aircraft of the same category



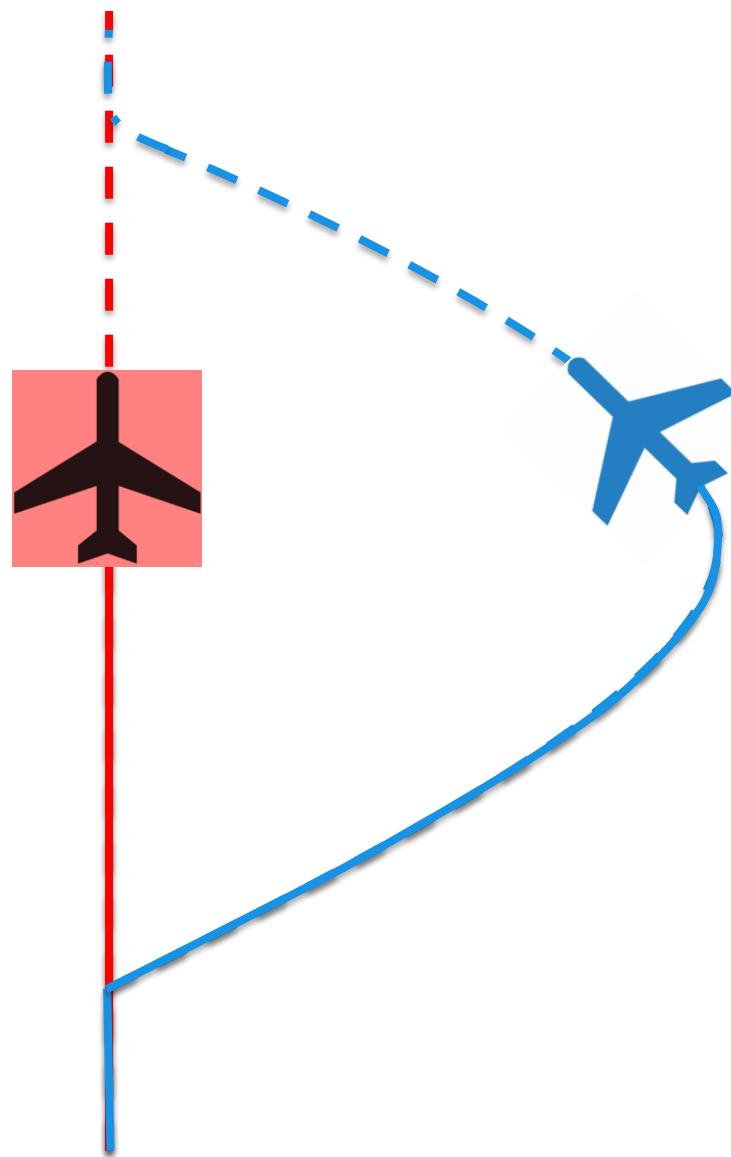
Lower speed aircraft is  
being overtaken

# Overtaking aircraft of the same category



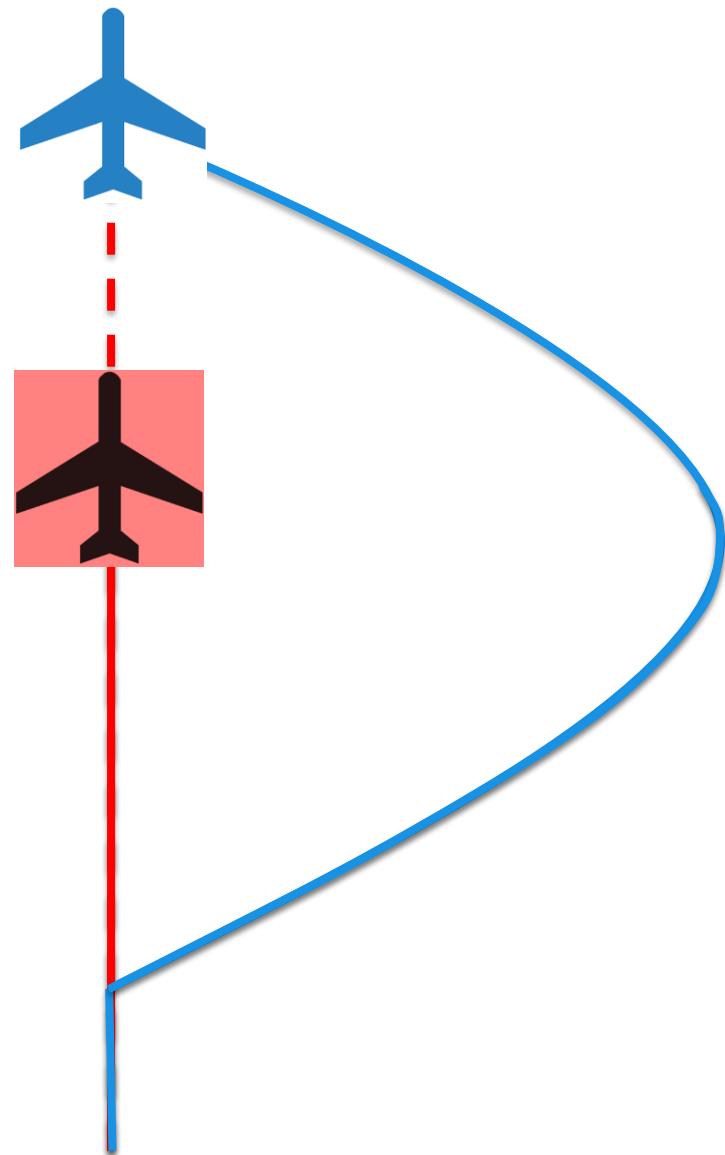
The aircraft being overtaken has the right of the way

# Overtaking aircraft of the same category



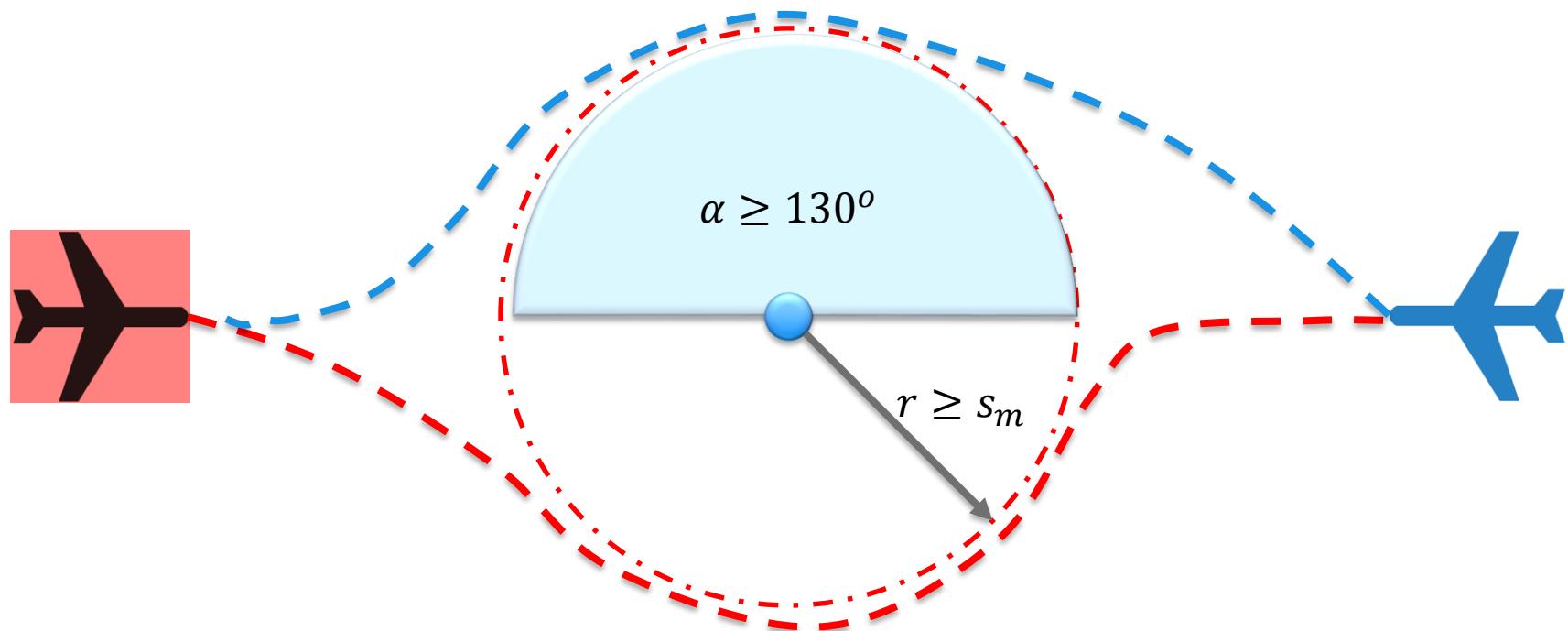
Overtake is executed from  
right

# Overtaking aircraft of the same category



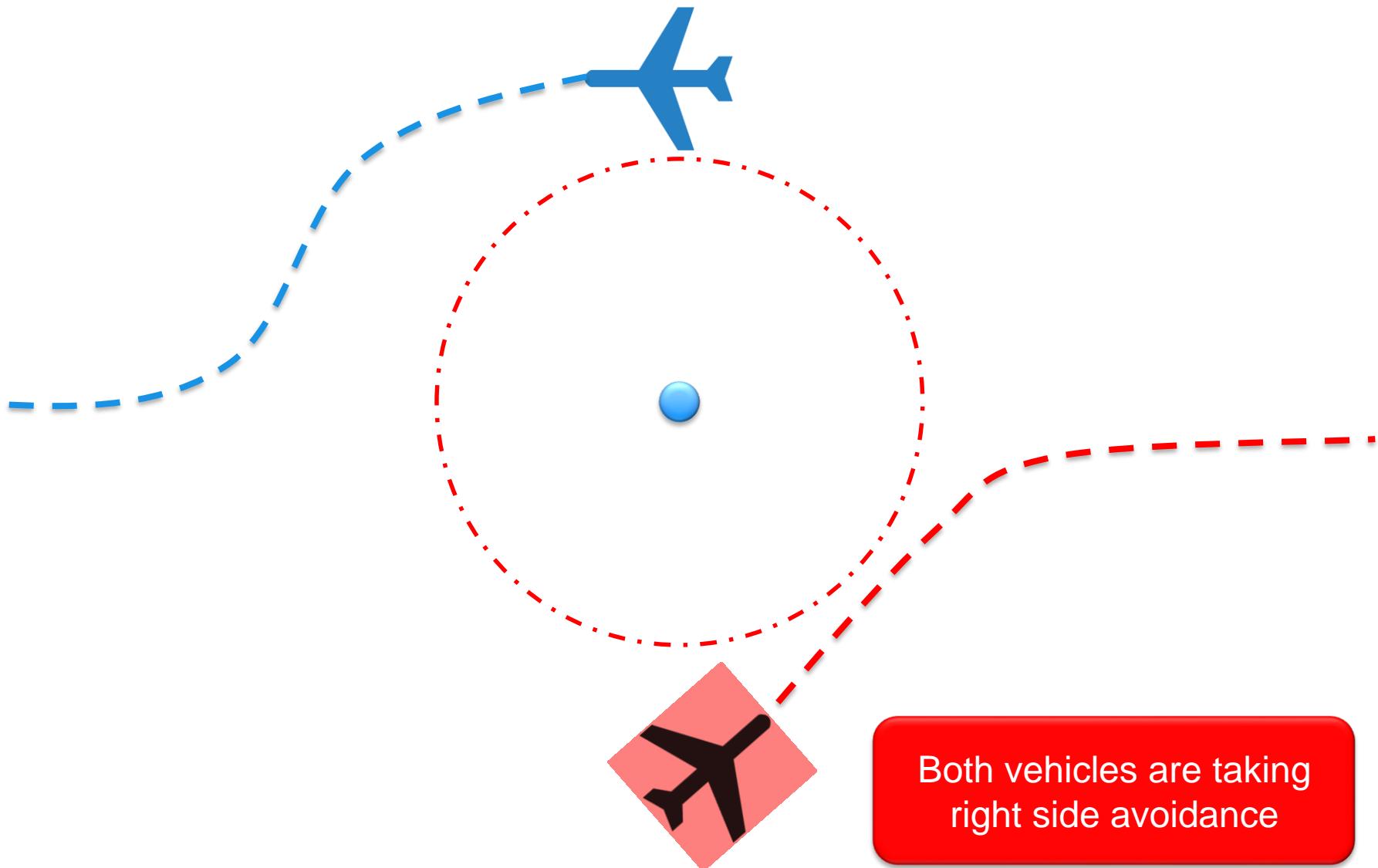
Overtake rule apply also  
for convergence under  
 $\alpha < 70^\circ$

# Head on approach of same category



Applies when angle of approach  $\alpha \geq 130^\circ$ , Safety margin  $s_m$  must be defined

# Head on approach of same category



# Summary - Right of the way

- Manned aviation defines only horizontal avoidance, vertical avoidance is considered in TCAS, horizontal avoidance will be added in ACAS-Xu
- Manned aviation does not define angle of approach  $\alpha$  strictly, it is usually used following margins:
  - $180^\circ \geq \alpha \geq 130^\circ$  Head on avoidance (both avoiding)
  - $130^\circ > \alpha \geq 70^\circ$  Covering avoidance (the left avoiding)
  - $\alpha < 70^\circ$  Overtake (faster avoiding)
- Safety margin  $s_m$  for Head on avoidance is defined as sufficient distance to avoid vehicle induced turbulence or obstruction of airflow, in case of UAV it must be defined based on UAV class size (bigger UAV), this can induce additional problems
- Overall “rules of air” and “right of the way” should be amended to be compatible with UAV integration into non-segregated airspace
- Pilot judgement is always the main consideration, therefore manned aviation behavior is unpredictable to some extent

# Last chat with UAV



Well Lojzo I see there is still a lot of challenges, to make me safe....  
But when will be this feasible ?

Our Framework is already covering:  
[1] Path planning, [2] Evolving world, [3] Adversarial, [4] Data fusion,  
[5] Static restrictions (somehow)  
Now we are working on:  
[6] Weather, [7] Rules



That seems all nice, when I learn proper manners and add my crafty skills to avoid [6] Weather, I can finally fulfill my destiny:  
To be useful for society



Well my friend it will take a lot of effort, but I will do my best to help you,  
At least on theoretical level, then you can fulfill your destiny:  
Deliver me that Pizza

# Summary

## **Computational feasibility:**

- Proposed method is computationally feasible, because its using only simplistic intersection and pruning algorithms for Reach Set estimation in bounded Avoidance Grid
- Full Reach set can be precomputed for finite number of initial states

## **Provided checks and functionality:**

- Checking system dynamic/static constraints with finite precision
- Checking environment dynamic/static constraints
- Prediction of feasible trajectory in limited space
- Selection of optimal trajectory in limited space

## **Planned checks and functionality:**

- Weather Flight restriction zones avoidance
- Rules of the air implementation

# Q&A Session

