Department of Control and Computer Engineering

Doctoral Dissertation
Doctoral Program in Computer and Control

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Autonomous Navigation for Mobile Robots in Crowded Environments

Challenges with ground and aerial robots

Stefano Primatesta

Supervisor
Prof. Alessandro Rizzo

Doctoral Examination Committee
Prof. Giunluca Ippoliti, Referee, Università Politecnica delle Marche, Italy
Prof. Kimon Valavanis, Referee, University of Denver, USA
Prof. Marina Indri, Politecnico di Torino, Italy
Prof. Bartolomeo Montrucchio, Politecnico di Torino, Italy
Prof. Domenico Prattichizzo, Università di Siena, Italy

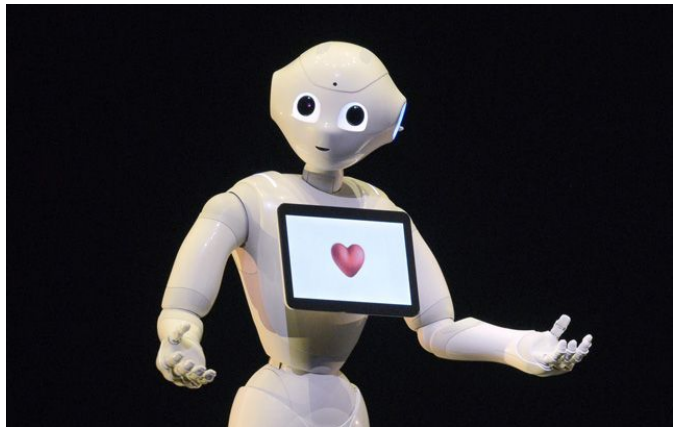
JOINTOPENLAB
TIM
POLITECNICO DI TORINO

Outline

- **Introduction**
- Safe Navigation for UAS in urban areas
  - Cloud-based framework for UASs
  - Risk-based map
  - Risk-aware path planning
  - Simulation
- Autonomous navigation for ground robots
  - Cloud-based architecture for Service Robotics Applications
  - Dynamic path planning
  - Motion control with PF-MPEPC
  - Service Robotics Applications
- Conclusions

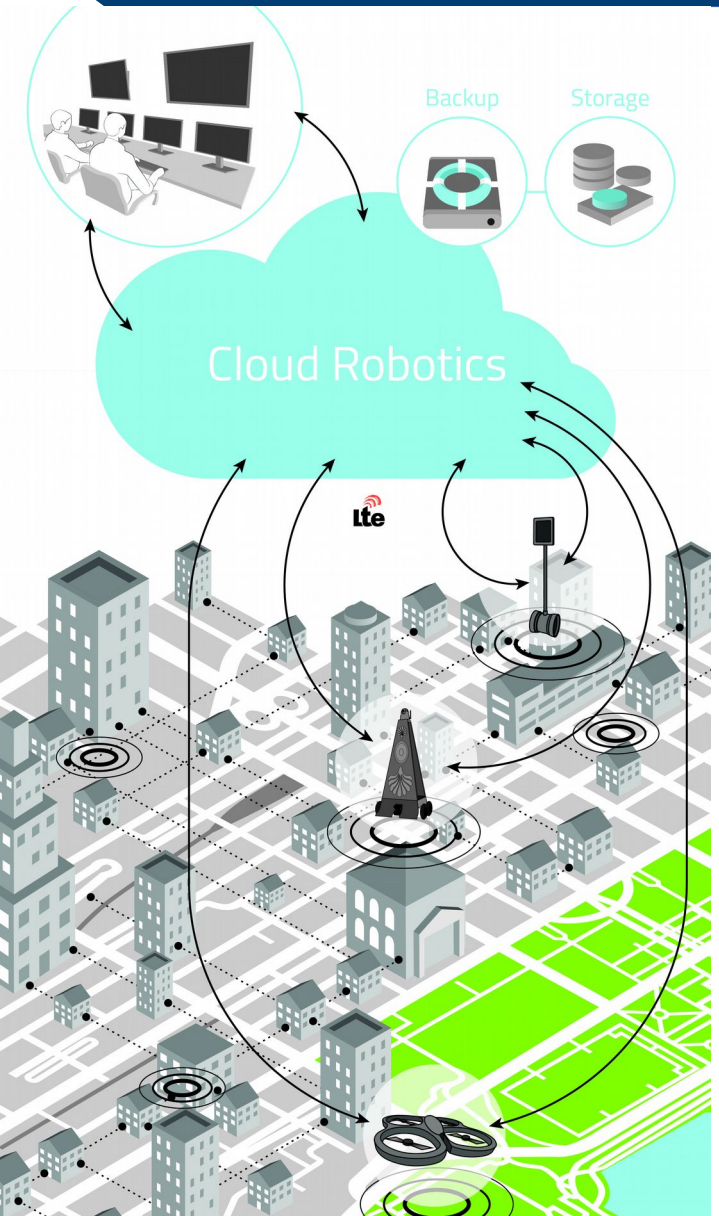
Introduction

- The number of robot acting in contact with people is growing
- Robots are used in a wide variety of applications
- Thanks to **Service Robotics**, robots are getting popular in our life
- A service robot is *“a robot that performs useful tasks to humans or equipment excluding industrial automation applications”* [1]



[1] ISO 8373. Robots and robotic devices–Vocabulary. International Standardization Organization (ISO), 2012.

Introduction

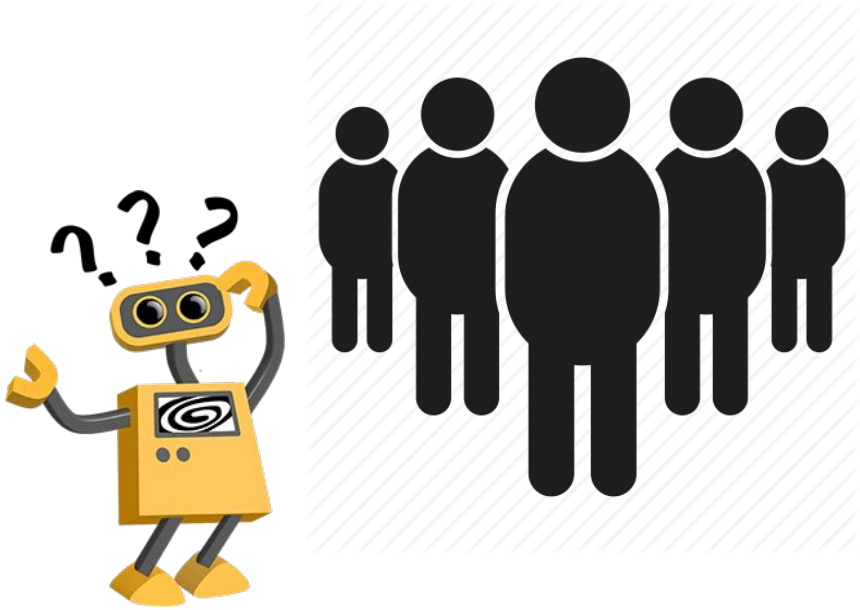


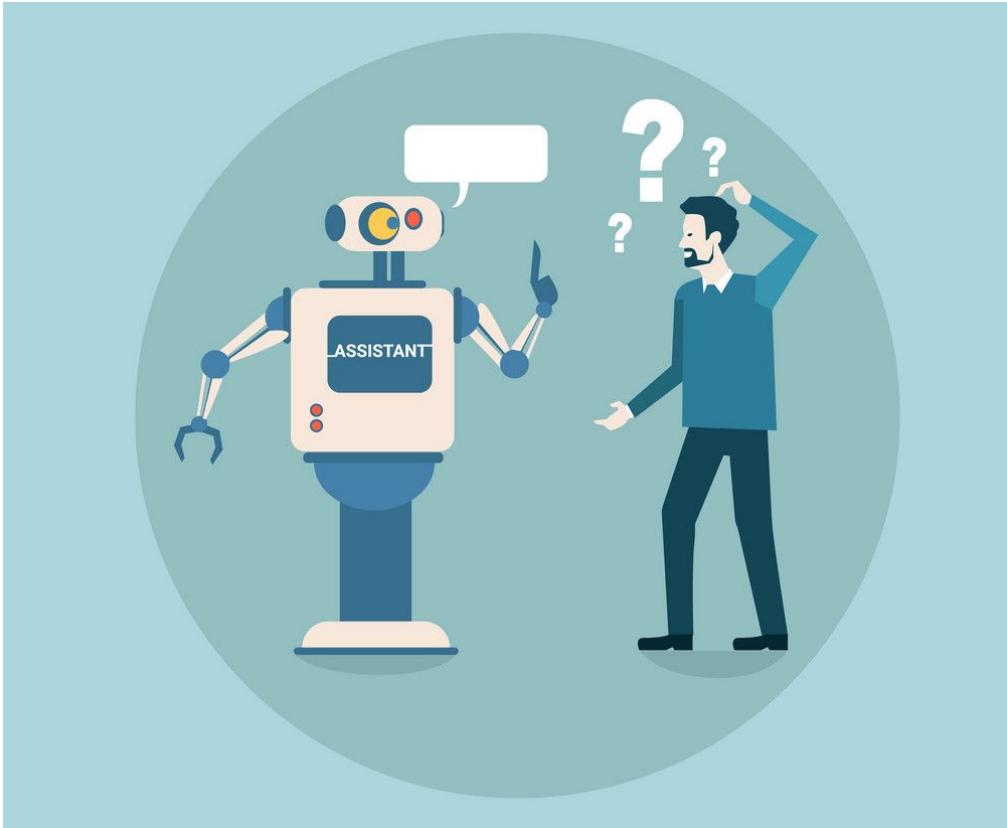
- The latest mobile technologies (e.g. 4G, 5G) connect the robot on Internet, opening Service Robotics to the Cloud Robotics paradigm
- A Cloud Robot is:
  "*Any robot that relies on either data or code from a network to support its operation, i.e., not all sensing, computation and memory is integrated into a single standalone system*" [2]
- The robot becomes a simple agent. Most of intelligence resides on-Cloud

[2] B Kehoe et al. "A survey of research on cloud robotics and automation". In: IEEE Transactions on automation science and engineering. 2015

Introduction: main problem

- The presence of humans in the operational environment is one of the major problem in Service Robotics
- A human environment is complex (dynamic and unstructured)
- Human safety must be guaranteed





Thesis Contribution

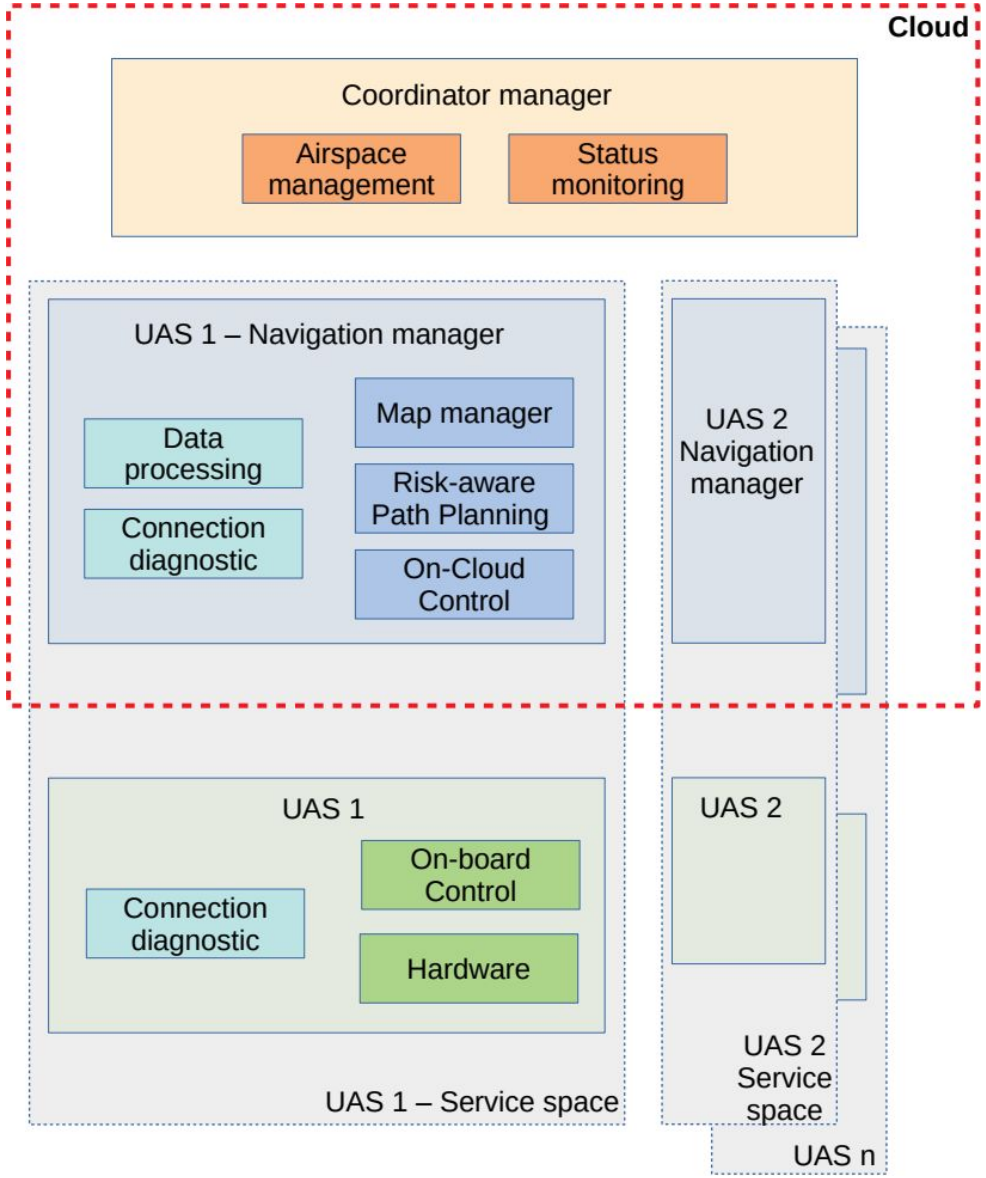
- The aim of the thesis is to study a ***safe and autonomous navigation for mobile robots*** to provide service robotics applications
- Two scenarios:
  - Unmanned Aerial System in urban areas
  - Ground robot navigation in crowded environments

Outline

- Introduction
- **Safe Navigation for UAS in urban areas**
  - **Cloud-based framework for UASs**
  - Risk-based map
  - Risk-aware path planning
  - Simulation
- Autonomous navigation for ground robots
  - Cloud-based architecture for Service Robotics Applications
  - Dynamic path planning
  - Motion control with PF-MPEPC
  - Service Robotics Applications
- Conclusions

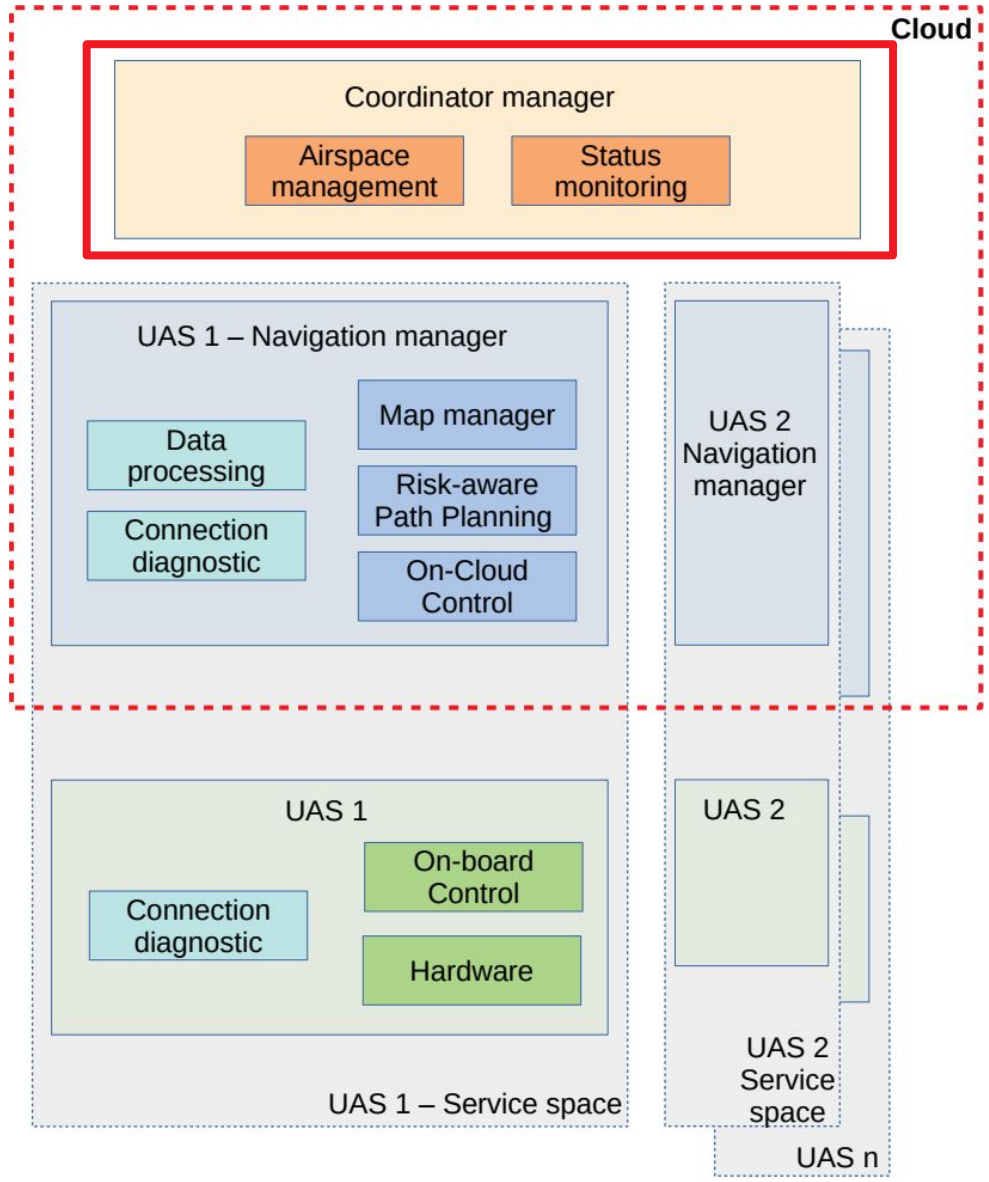
Cloud-based architecture for UASs

- The aim is to propose a reference framework to enable safe flight operations in urban areas
- The architecture is distributed between the Cloud and UASs
- The framework coordinates a fleet of UASs
- Minimization of the risk to the population on the ground
- UASs perform autonomous flight in BVLOS (Beyond Visual Line-Of-Sight)
- UASs are connected with the Cloud using mobile networks



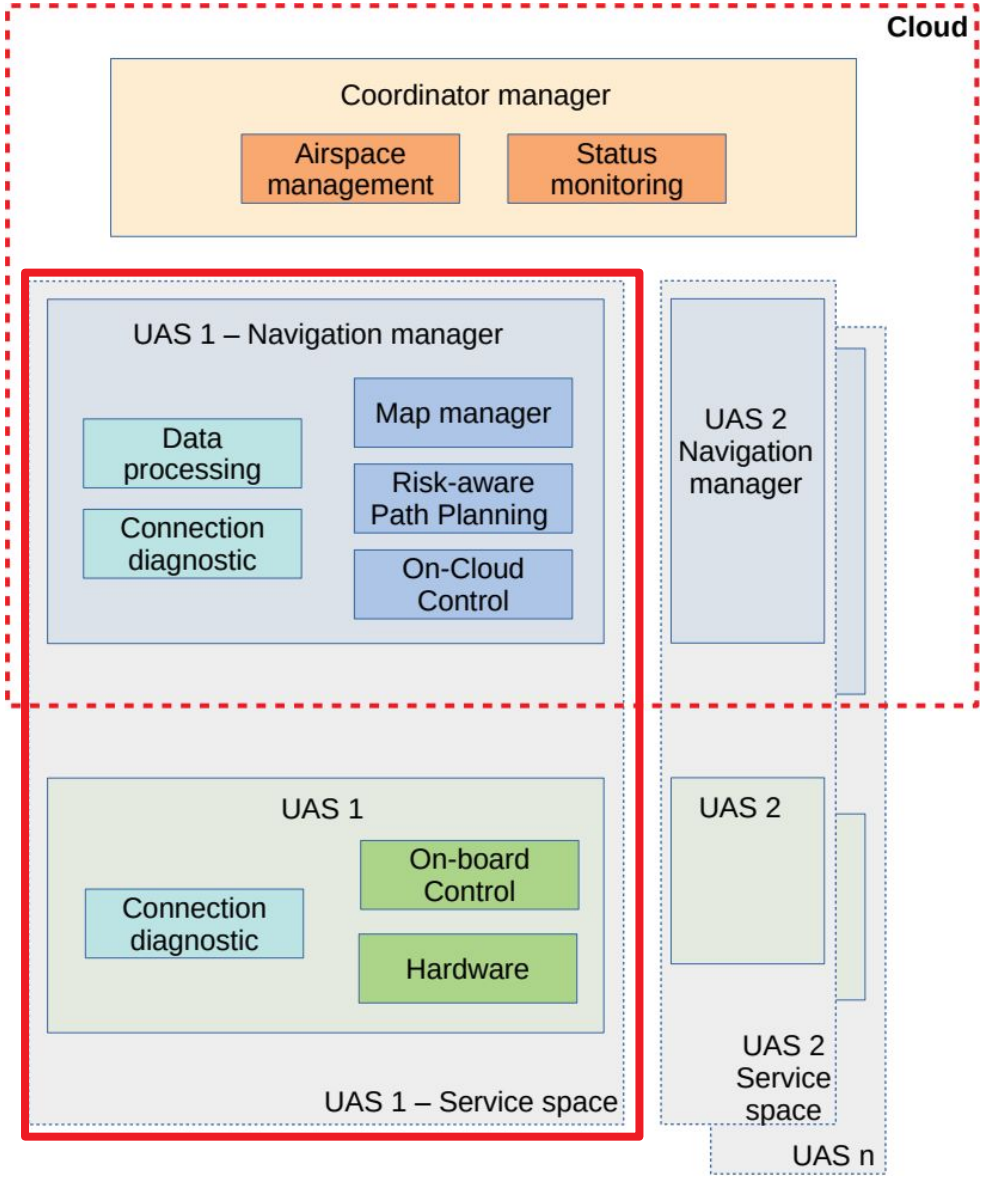
Main architecture (1)

- The **Coordinator manager** is the core of the architecture
- It manages the low altitude airspace (priority assignment, identification of the *operational area*)
- It monitors each flight mission



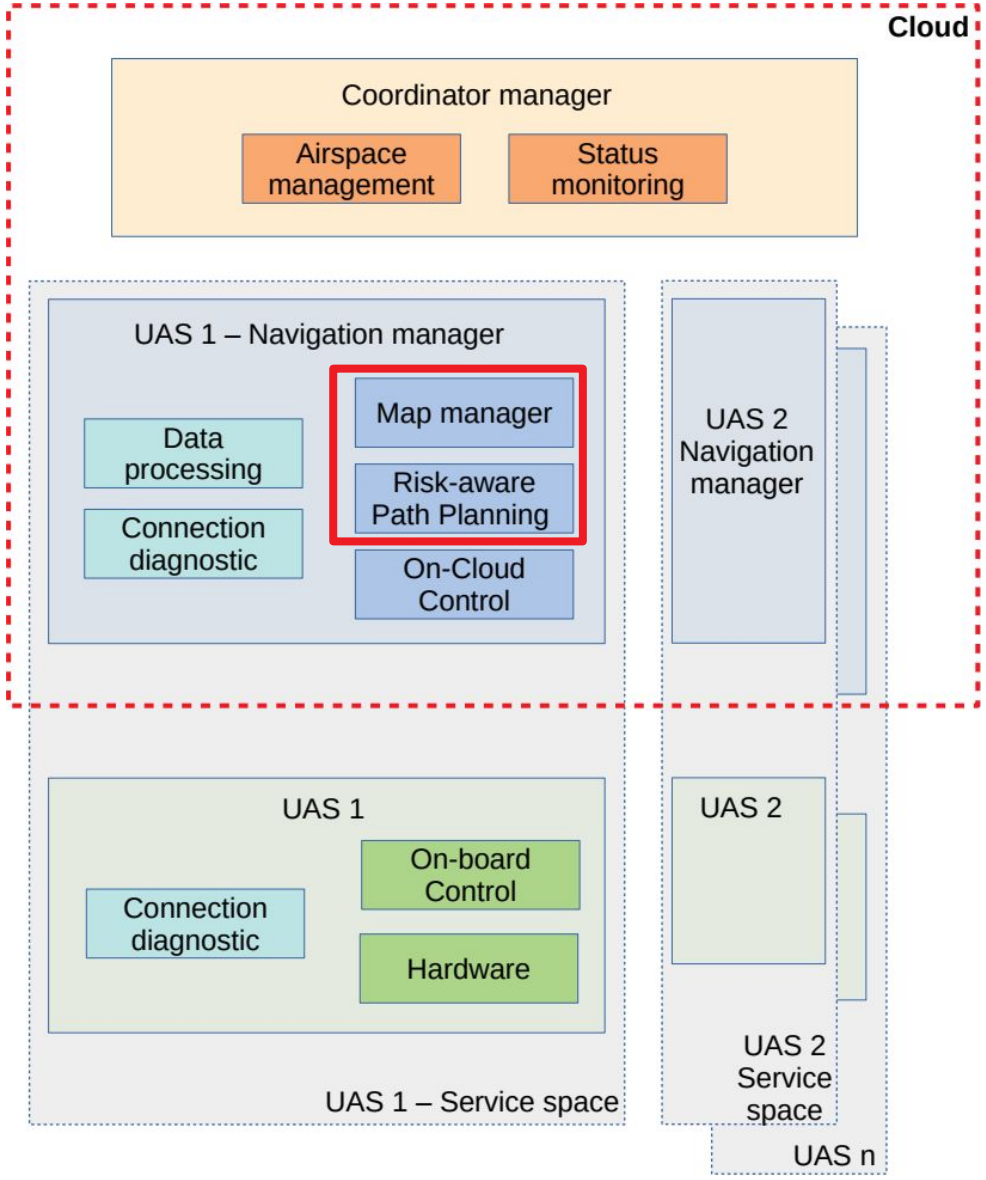
Main architecture (2)

- An **UAS Service Space** is determined for each UAS managed by the framework
- The **Navigation Manager** is allocated on-Cloud and plans and executes a safe flight mission
- The **UAS** module refers to the aircraft, including both on-board software and hardware
- The **Connection Diagnostic** monitors the communication between the Cloud and the UAS
  - It evaluates the network latency and detects disconnections
  - Safety must be maintained in case of disconnection or bad quality of communication



Main architecture (3)

- In the Ph.D. thesis, two elements are studied and developed:
  - Map Manager
  - Risk-aware Path Planning

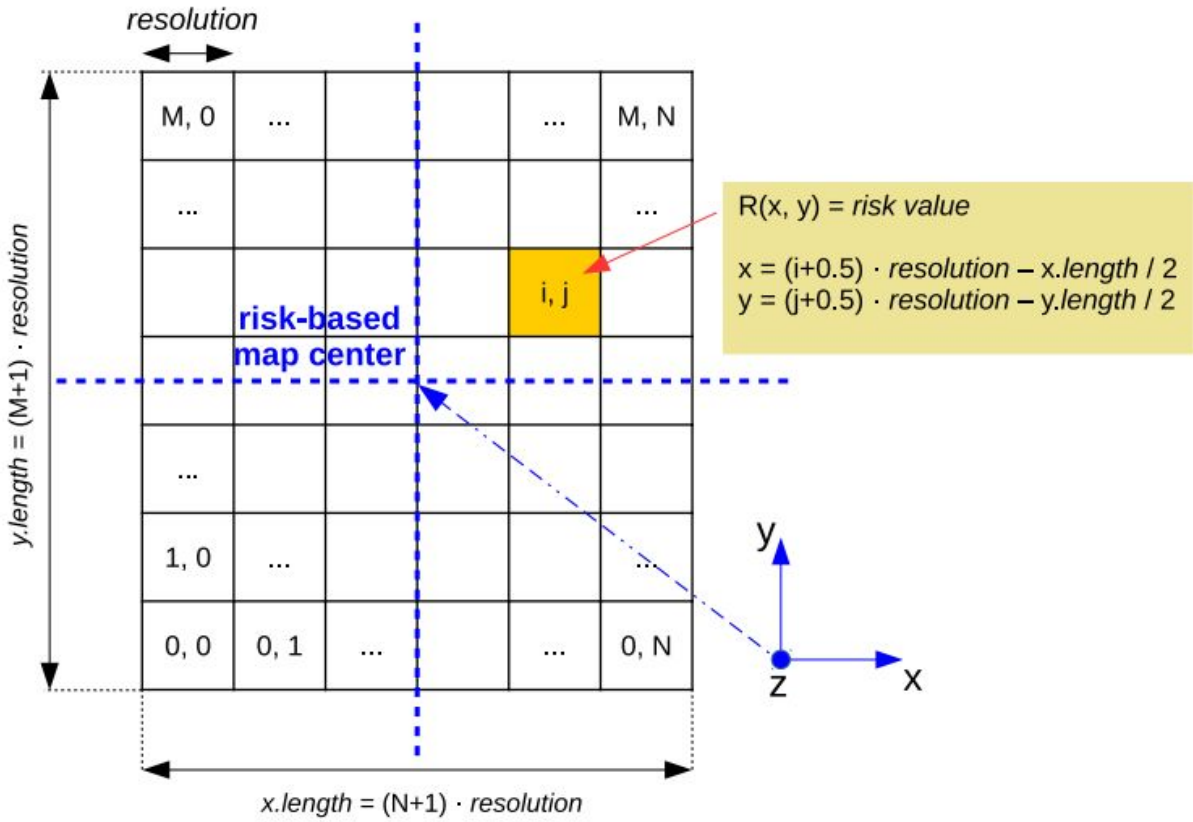


Outline

- Introduction
- **Safe Navigation for UAS in urban areas**
  - Cloud-based framework for UASs
  - **Risk-based map**
  - Risk-aware path planning
  - Simulation
- Autonomous navigation for ground robots
  - Cloud-based architecture for Service Robotics Applications
  - Dynamic path planning
  - Motion control with PF-MPEPC
  - Service Robotics Applications
- Conclusions

Risk-based map: definition

- The **risk-based map** is a tool for risk-informed decision-making [3]
- It quantifies the risk to the population on the ground in urban areas
- The risk-based map is a two-dimensional location-based map, denoted as $\boldsymbol{R}$
- Each cell $\boldsymbol{R}(i, j)$ represents a georeferenced location centered in (x, y) and has a risk value
- For simplicity, a cell is denoted as $\boldsymbol{R}(x, y)$



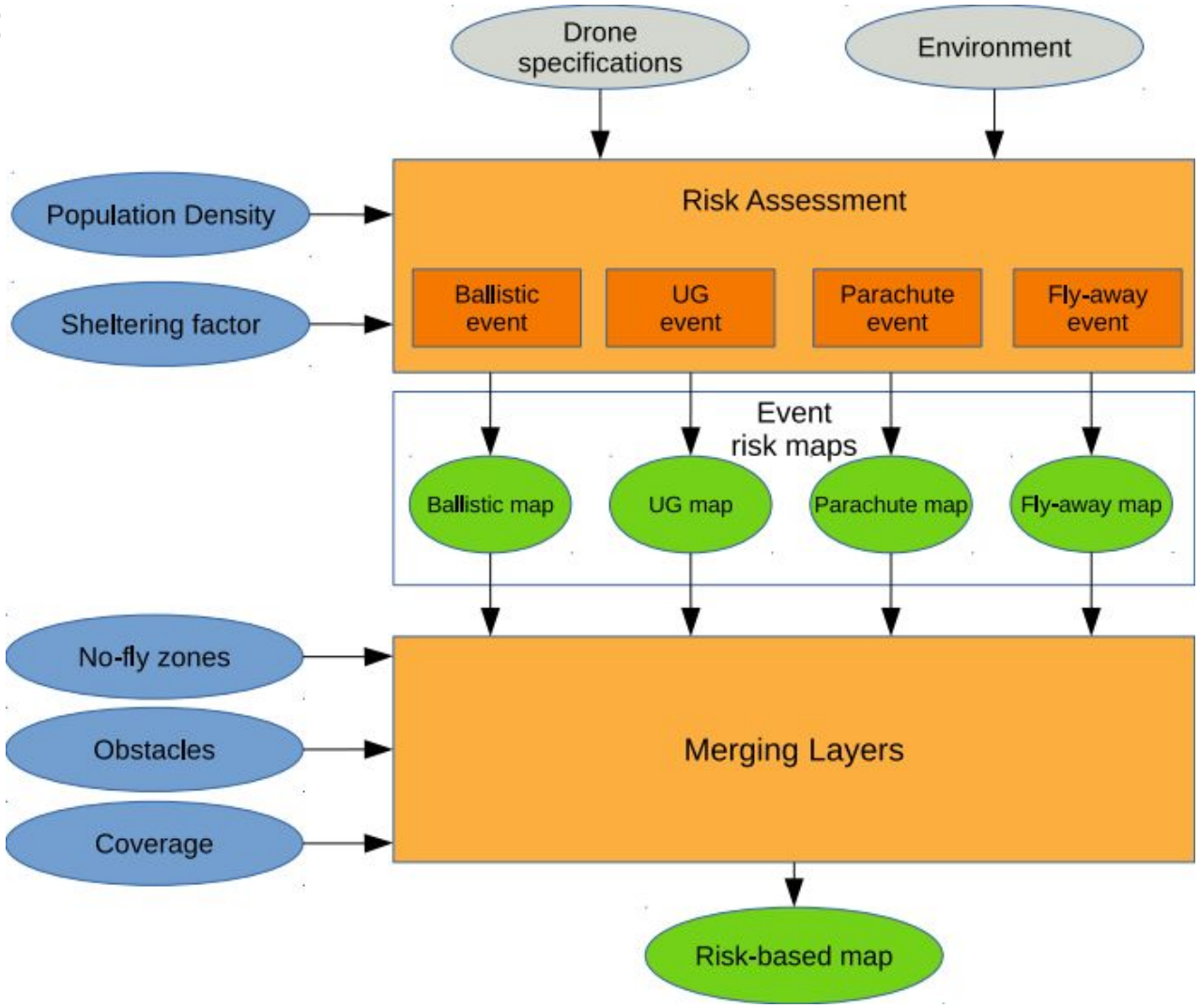
[3] S Primatesta, et al. "Ground risk map for Unmanned Aircraft in Urban Environments". In: Journal of Intelligent & Robotic Systems, 2019.

Risk-based map: generation

- The risk map is computed considering several layers:
  - *Population density layer*
  - *Sheltering factor layer*
  - *No-fly zones layer*
  - *Obstacles layer*
  - *Coverage layer*

- Each layer has the same characteristics of the risk-based map

- Four different descent events:
  - *Ballistic descent*
  - *Uncontrolled glide descent*
  - *Parachute descent*
  - *Fly-away event*



Risk-based map: risk assessment

- The risk is defined as the risk to the population on the ground, when the UAS flies over a specific area
- It is expressed in casualties *per hour* (h^{-1})
- The risk is computed as the probability to have a casualty $P_{casualty}$:

$$P_{casualty}(x, y) = P_{event} \cdot P_{impact}(x, y) \cdot P_{fatality}(x, y)$$

  - P_{event} is the probability that the UAS loses the control with the uncontrolled descent and impact on the ground
  - P_{impact} is the probability to impact with a person
  - $P_{fatality}$ is the probability that a hit person suffers fatal injuries
- It is a probabilistic risk assessment approach commonly used in the literature [4], [5]

[4] K Dalamagkidis et al. On integrating unmanned aircraft systems into the national airspace system: issues, challenges, operational restrictions, certification, and recommendations. Springer, 2012.
[5] R A Clothier et al. "A casualty risk analysis for unmanned aerial system (UAS) operations over inhabited areas". In: Proc. 12th Australian International Aerospace Congress. 2007.

Risk-based map: risk assessment (2)

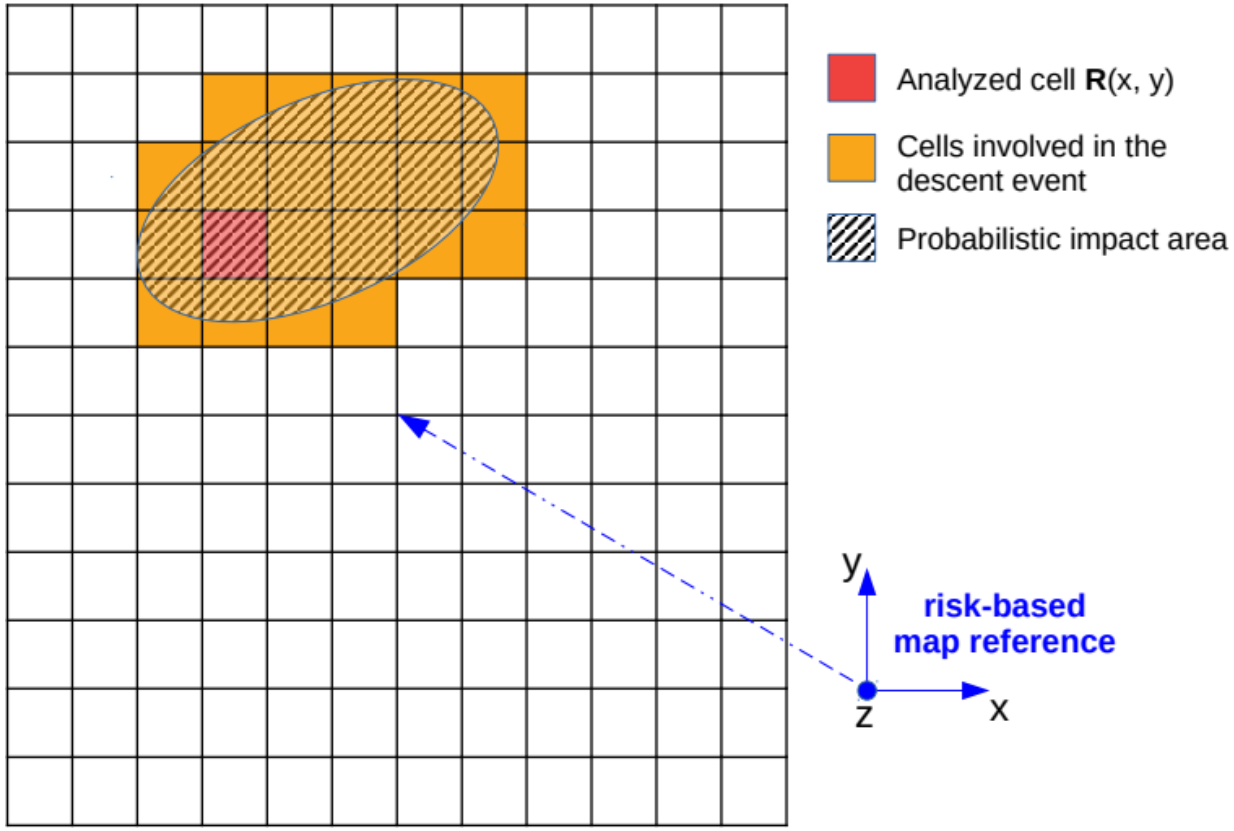
- The risk is computed assuming 4 different descent event (ballistic, uncontrolled glide, parachute and fly-away)
- Each descent type implies a different behavior (impact area, impact angle and velocity)
- Despite the descent type, a common procedure is applied to compute the risk:

  1) The probabilistic impact area is computed, considering the drone specifications and initial conditions (cruise velocity, flight altitude, etc.). It is defined as a ***two-dimensional probability density function (PDF)***

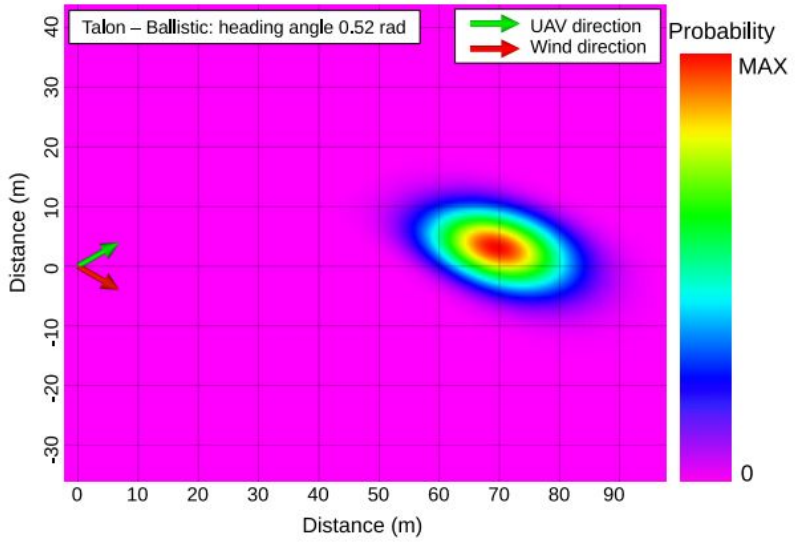
  2) The probabilistic impact area is modified according to the probabilistic wind

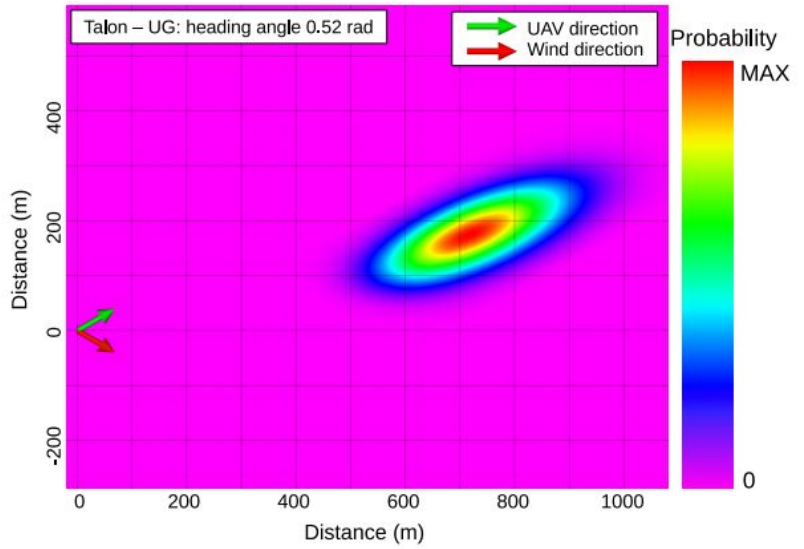
  3) The two-dimensional PDF is used to compute the probabilities P_{impact} and $P_{fatality}$

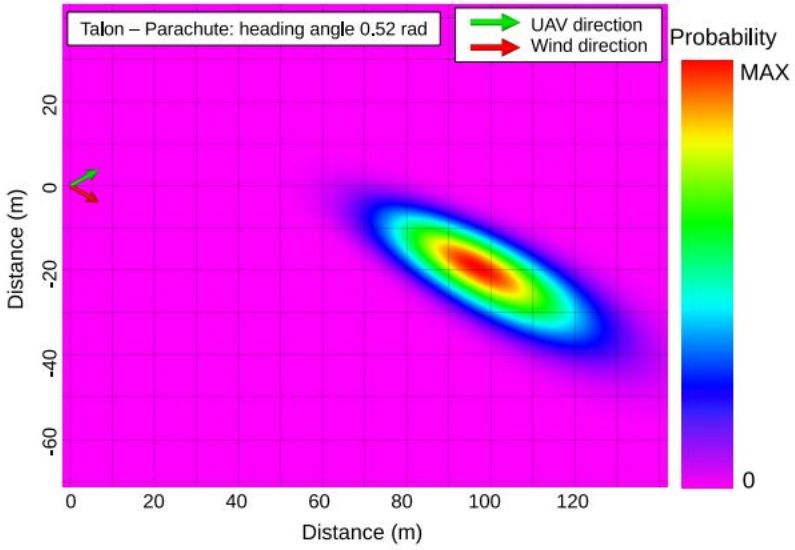
  4) The probability $P_{casualty}$ is computed for each element of the map

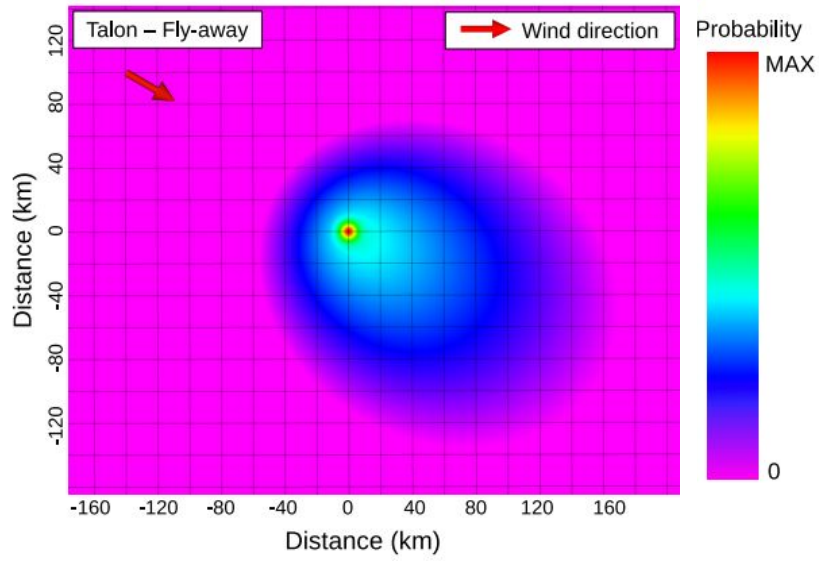


Risk-based map: descent event types









Heliscope Talon

Fixed wing aircraft

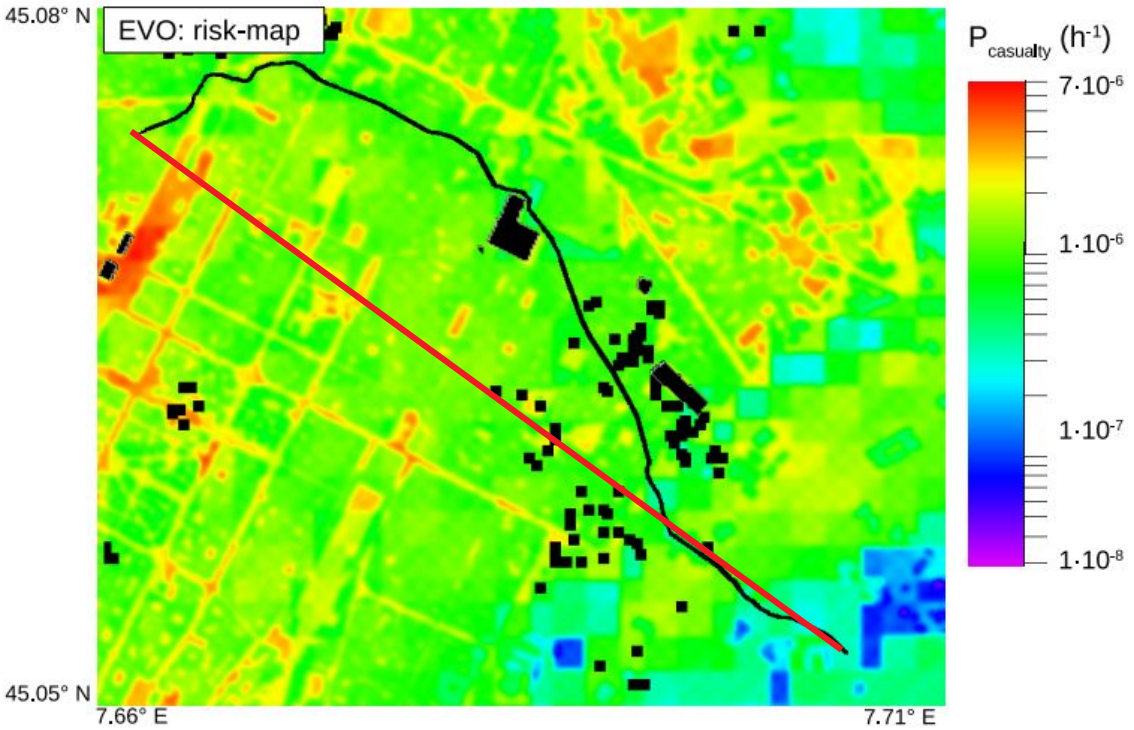
Mass: 3.75 kg

Speed: 18 m/s

Altitude: 50 m

Wind speed: 10 m/s

Risk-based map: Vehicles

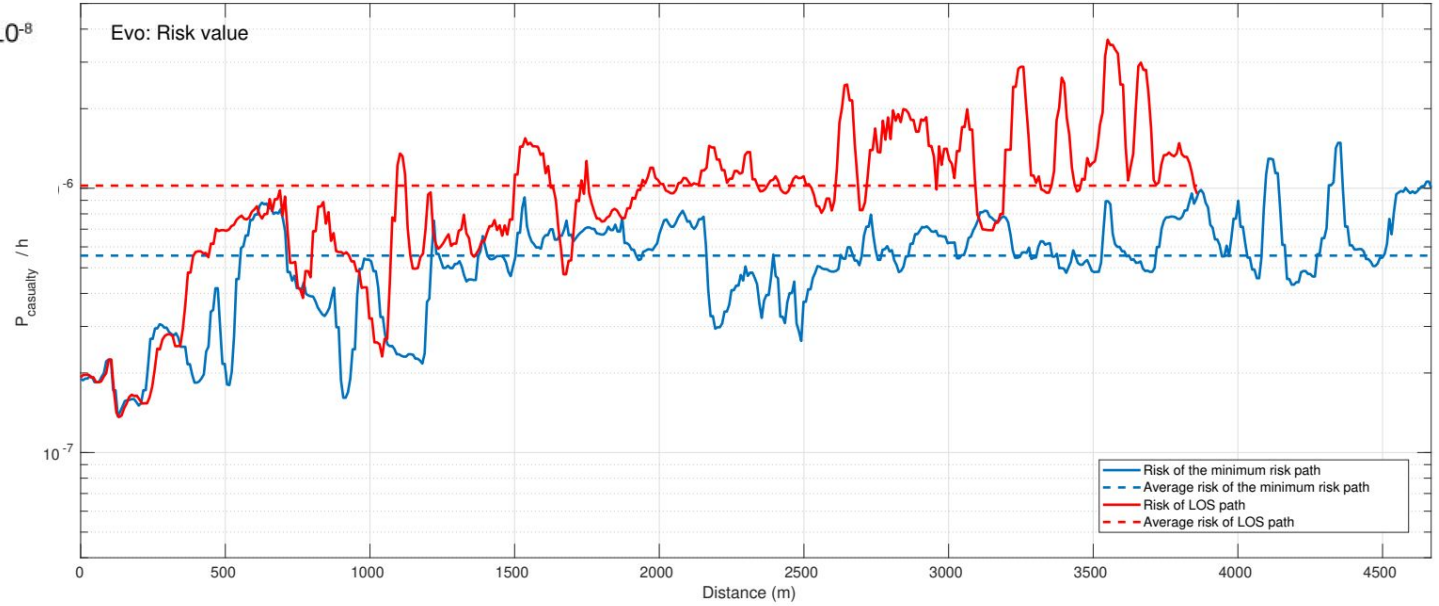


ADPM Evo
Quadrotor
Mass: 0.3 kg
Speed: 9.7 m/s

Altitude: 50 m
Wind speed: 5 m/s

Equivalent Level Of Safety (ELOS) = $1 \cdot 10^{-6}\ h^{-1}$

Risk of the minimum risk path = $5 \cdot 10^{-7}\ h^{-1} <$ **ELOS**



Risk-based map: Vehicles

Specific	Talon	DJI Phantom 4	DJI Inspire 2	Parrot Disco	DJI Mavic	Parrot Bebop	ADPM EVO
Type	Fixed wing	Quadrotor	Quadrotor	Fixed wing	Quadrotor	Quadrotor	Quadrotor
Mass (kg)	3.75	1.4	4.25	0.75	0.7	0.5	0.3

Vehicle		Ballistic	Uncontrolled glide	Parachute	Fly-away	Final risk
Talon	min	$7.820 \cdot 10^{-7}$	$2.392 \cdot 10^{-6}$	$2.894 \cdot 10^{-7}$	$1.302 \cdot 10^{-6}$	$8.497 \cdot 10^{-6}$
	max	$2.042 \cdot 10^{-4}$	$1.638 \cdot 10^{-5}$	$6.346 \cdot 10^{-5}$	$3.593 \cdot 10^{-6}$	$2.784 \cdot 10^{-4}$
	av	$2.161 \cdot 10^{-5}$	$1.056 \cdot 10^{-5}$	$9.164 \cdot 10^{-6}$	$1.962 \cdot 10^{-6}$	$4.329 \cdot 10^{-5}$
Phantom	min	$9.439 \cdot 10^{-8}$	$6.518 \cdot 10^{-9}$	0	$7.105 \cdot 10^{-9}$	$1.106 \cdot 10^{-7}$
	max	$2.631 \cdot 10^{-5}$	$2.565 \cdot 10^{-7}$	0	$1.535 \cdot 10^{-8}$	$2.653 \cdot 10^{-5}$
	av	$2.977 \cdot 10^{-6}$	$8.705 \cdot 10^{-8}$	0	$9.961 \cdot 10^{-9}$	$3.074 \cdot 10^{-6}$
Inspire	min	$4.479 \cdot 10^{-7}$	$2.823 \cdot 10^{-7}$	$1.369 \cdot 10^{-7}$	$3.518 \cdot 10^{-7}$	$1.369 \cdot 10^{-6}$
	max	$1.107 \cdot 10^{-4}$	$1.030 \cdot 10^{-5}$	$3.430 \cdot 10^{-5}$	$5.330 \cdot 10^{-7}$	$1.498 \cdot 10^{-4}$
	av	$1.201 \cdot 10^{-5}$	$3.045 \cdot 10^{-6}$	$4.660 \cdot 10^{-6}$	$4.159 \cdot 10^{-7}$	$2.013 \cdot 10^{-5}$
Disco	min	$2.942 \cdot 10^{-8}$	$1.011 \cdot 10^{-7}$	0	$5.409 \cdot 10^{-8}$	$3.494 \cdot 10^{-7}$
	max	$8.715 \cdot 10^{-6}$	$9.977 \cdot 10^{-7}$	0	$1.103 \cdot 10^{-7}$	$9.679 \cdot 10^{-6}$
	av	$1.326 \cdot 10^{-6}$	$6.703 \cdot 10^{-7}$	0	$7.051 \cdot 10^{-8}$	$2.067 \cdot 10^{-6}$
Mavic	min	$4.758 \cdot 10^{-8}$	0	0	0	$4.758 \cdot 10^{-8}$
	max	$1.093 \cdot 10^{-5}$	0	0	0	$1.093 \cdot 10^{-5}$
	av	$1.413 \cdot 10^{-6}$	0	0	0	$1.413 \cdot 10^{-6}$
Bebop	min	$2.599 \cdot 10^{-8}$	0	0	0	$2.599 \cdot 10^{-8}$
	max	$6.853 \cdot 10^{-6}$	0	0	0	$6.853 \cdot 10^{-6}$
	av	$9.653 \cdot 10^{-7}$	0	0	0	$9.653 \cdot 10^{-7}$
EVO	min	$1.957 \cdot 10^{-8}$	0	-	0	$1.957 \cdot 10^{-8}$
	max	$6.482 \cdot 10^{-6}$	0	-	0	$6.482 \cdot 10^{-6}$
	av	$9.771 \cdot 10^{-7}$	0	-	0	$9.771 \cdot 10^{-7}$

Equivalent Level Of Safety (ELOS) = $1 \cdot 10^{-6}$ h^{-1}

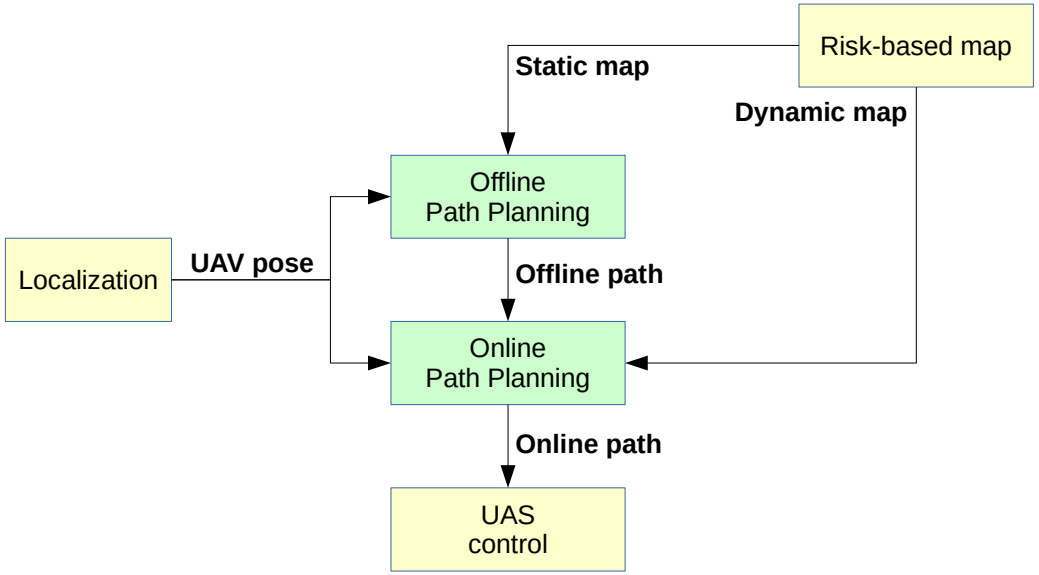
Only small and light UASs are suitable to fly over the analyzed urban area!

Outline

- Introduction
- **Safe Navigation for UAS in urban areas**
  - Cloud-based framework for UASs
  - Risk-based map
  - **Risk-aware path planning**
  - Simulation
- Autonomous navigation for ground robots
  - Cloud-based architecture for Service Robotics Applications
  - Dynamic path planning
  - Motion control with PF-MPEPC
  - Service Robotics Applications
- Conclusions

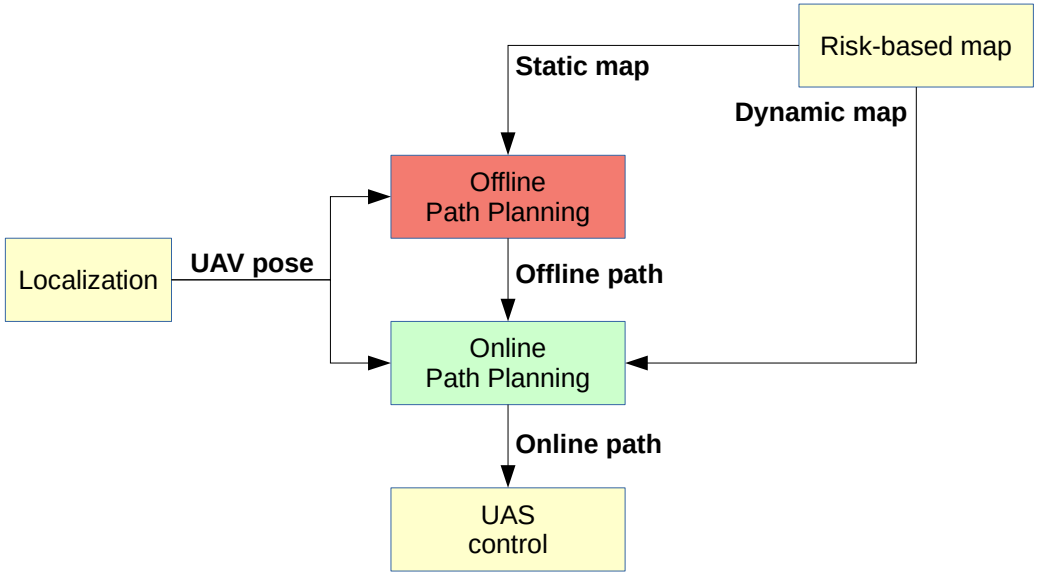
Risk-aware path planning

- The ***risk-aware path planning*** computes a safe path, considering both static and dynamic risk factors for the population on the ground
- It minimizes risk values of the risk-based map
- It consists of two phases:
    - Offline path planning
    - Online path planning
- The path is handed over the UAS control system



Risk-aware path planning: offline path planning

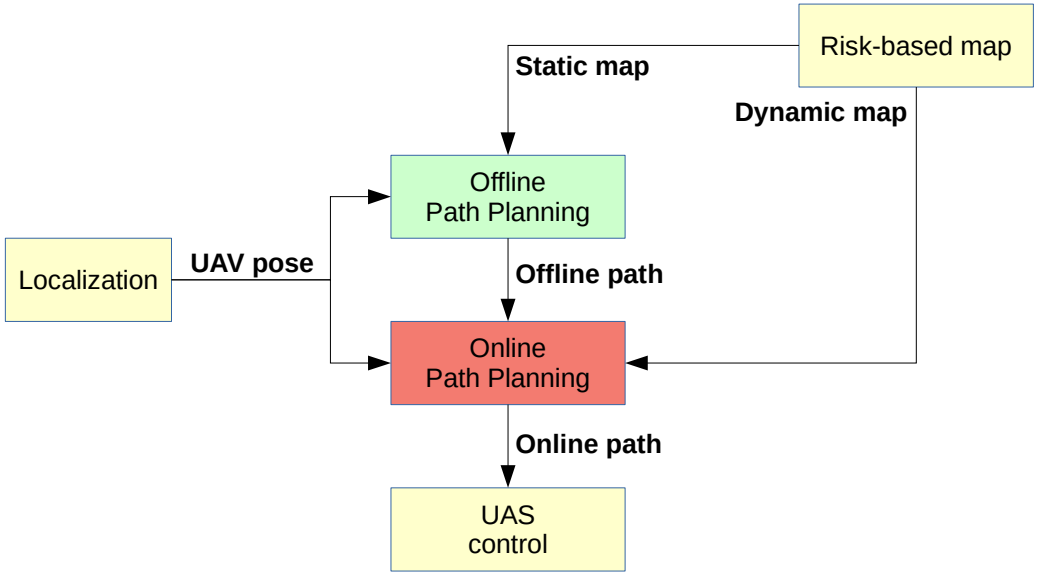
- The ***offline path planning*** searches for the globally optimal path from the start to the target position
- It minimizes the risk to the population on the ground defined by the risk-based map
- No tight time constraints
- Algorithms: riskA* [6], riskRRT^{X} (offline)



[6] Stefano Primatesta et al. "A Risk-Aware Path Planning Strategy for UAVs in Urban Environments". In: Journal of Intelligent & Robotic Systems, 2018.

Risk-aware path planning: online path planning

- The ***online path planning*** adapts the path to changes in the dynamic risk-based map
- Tight time constraints
- The optimal solution is not guaranteed
- Algorithms: Borderland [6], riskRRT^{X} (online)



[6] Stefano Primatesta et al. "A Risk-Aware Path Planning Strategy for UAVs in Urban Environments". In: Journal of Intelligent & Robotic Systems, 2018.

Risk-aware path planning strategies

- In the thesis, two different risk-aware path planning strategies are proposed:
  - RiskA* and Borderland algorithms
    - ***RiskA**** searches for the global optimal path
    - ***Borderland*** rapidly adapts the offline path
  - RiskRRT^{X} algorithm
    - ***RiskRRT**^{X}* is a path planning and re-planning algorithm

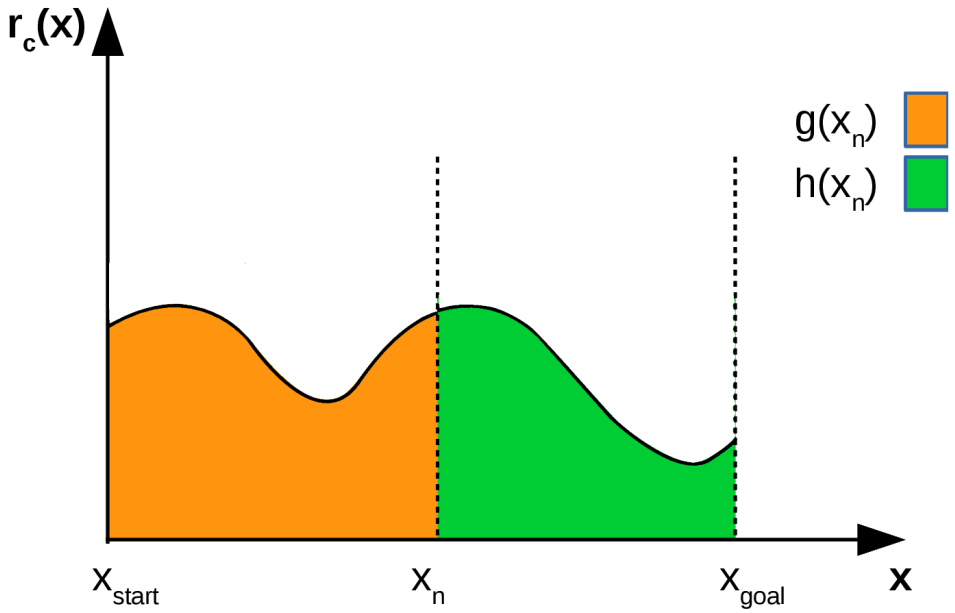
Risk-aware path planning: riskA*

- ***RiskA**** is an offline path planning algorithm based on the well-know A* [6]
- Unlike A*, the cost function $f(x)$ takes into account risk values of the risk-based map

$$f(x_n) = g(x_n) + k \cdot h(x_n)$$

$$g(x_n) = \int_{x_{start}}^{x_n} r_c(x)\,dx$$

$$h(x_n) = \int_{x_n}^{x_{goal}} r_c(x)\,dx$$



- $r_c(x)$ is the risk cost function based on the risk-based map

[6] Stefano Primatesta et al. "A Risk-Aware Path Planning Strategy for UAVs in Urban Environments". In: Journal of Intelligent & Robotic Systems, 2018.

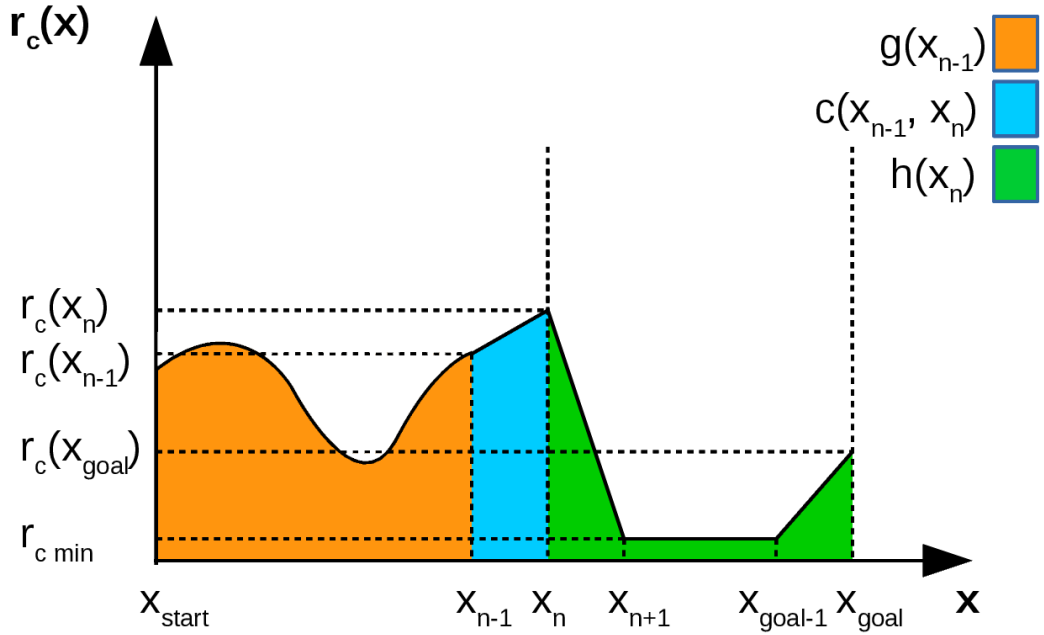
Risk-aware path planning: riskA* (2)

- Because of the discrete grid map, functions $g(x)$ and $h(x)$ are computed with a discrete and incremental approach

$$g(x_n) = \int_{x_{start}}^{x_{n-1}} r_c(x)\,dx + \int_{x_{n-1}}^{x_n} r_c(x)\,dx = g(x_{n-1}) + c(x_{n-1}, x_n)$$

$$\text{with} \quad c(x_{n-1}, x_n) = \frac{r_c(x_{n-1}) + r_c(x_n)}{2}\, dist(x_{n-1}, x_n)$$

$$h(x_n) = \frac{r_c(x_n) + r_c(x_{goal})}{2}\, dist_{min} + (dist(x_n, x_{goal}) - dist_{min})\, r_{c\,min}$$

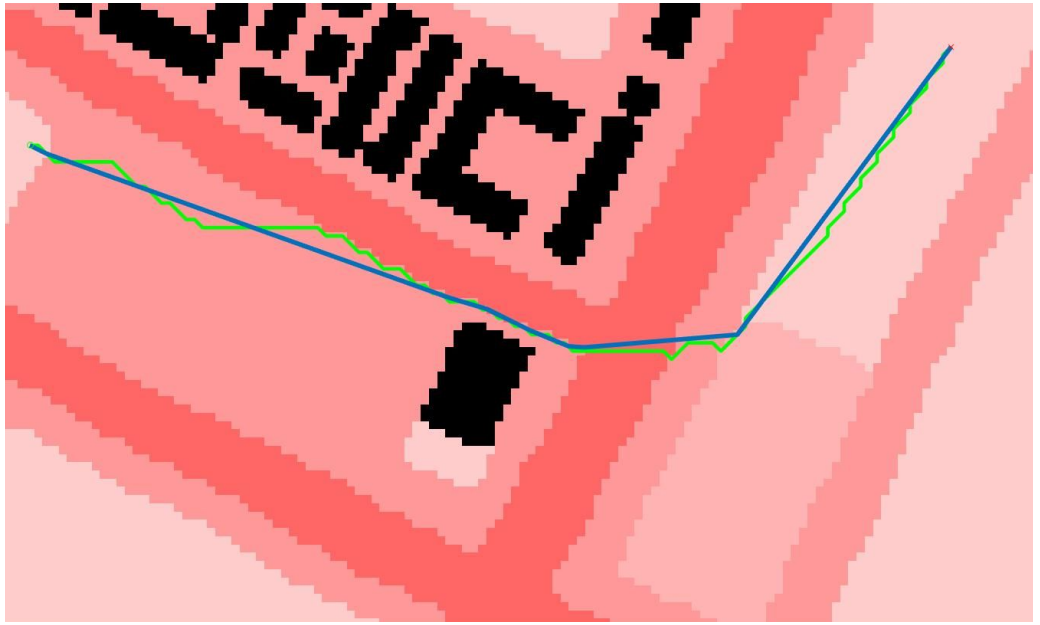


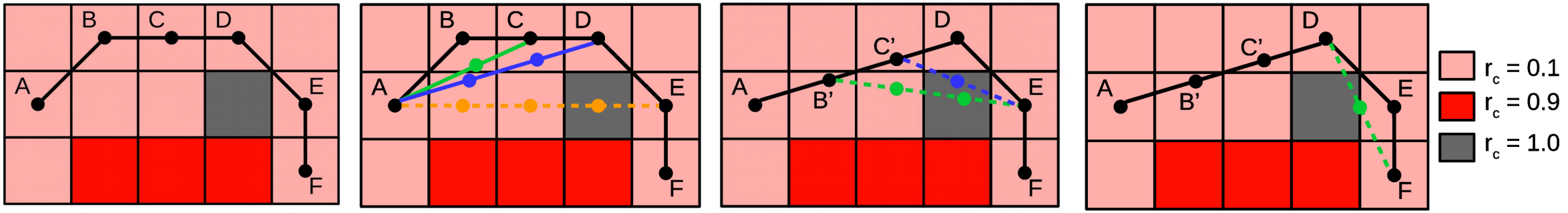
$dist_{min}$ is the minimum distance assumed between two adjacent nodes

$r_{c\,min} > 0$ is the minimum risk assumed

Risk-aware path planning: riskA* (3)

- Generally, A*-based algorithm searches for the optimal path in the graph (i.e. the grid map)
- The path is not optimal in the continuous space
- A *post-optimization* procedure is performed





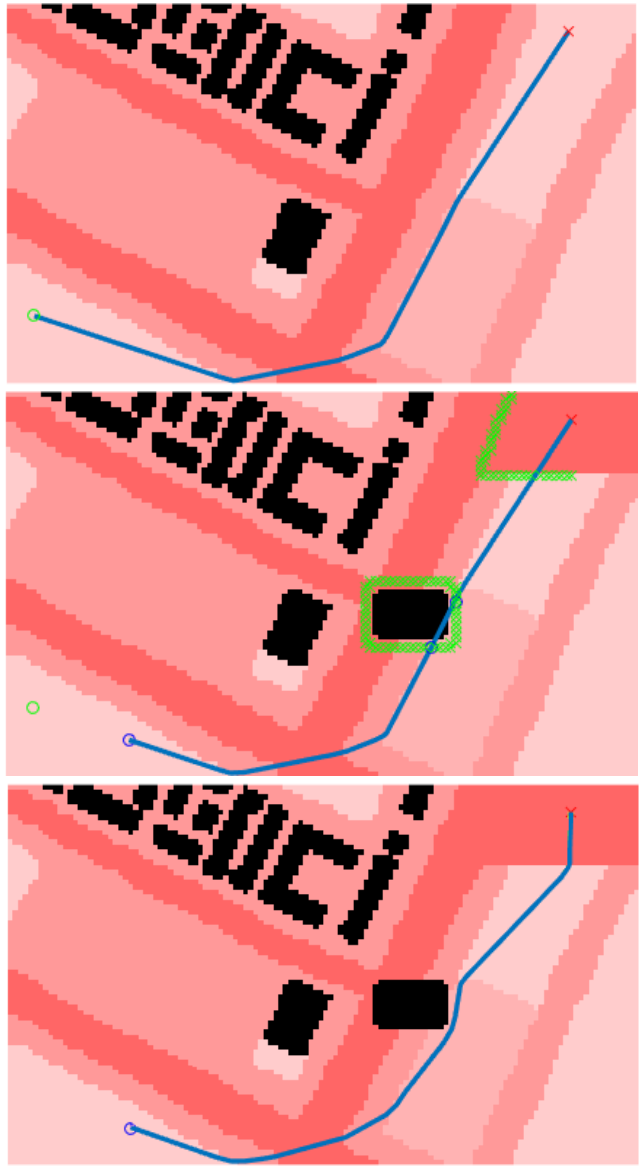
Risk-aware path planning: Borderland

- ***Borderland*** is an online path planning algorithm [6]
- It is an extension of the Bug algorithm, applied to grid graph
- It circumnavigates the areas involved by changes in the risk-based map, searching for an alternative path with less risk
- It uses a *check and repair* approach

    1) It verifies what portions of path are involved in the dynamic map

    2) It seeks for an alternative path and replaces the old portion of path with the new one

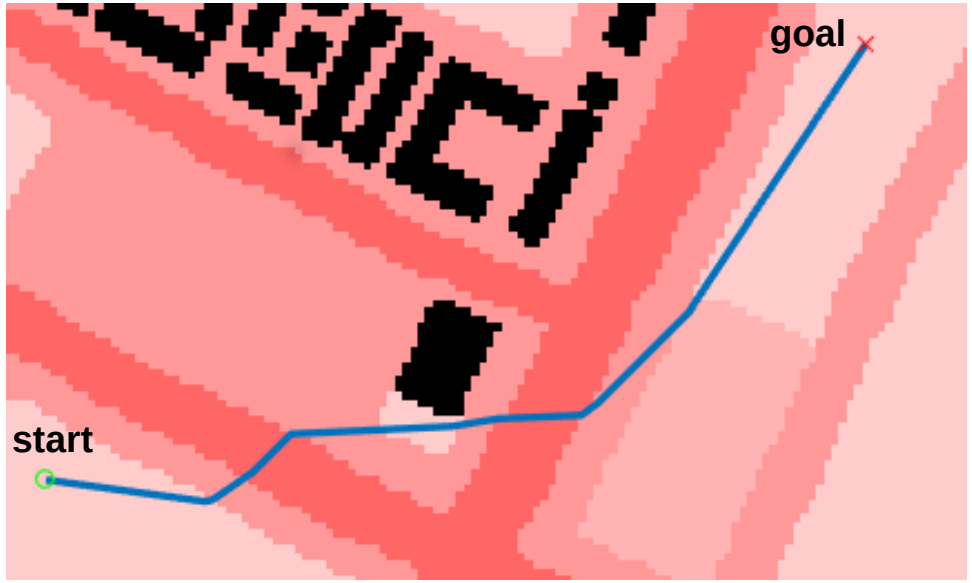
    3) A post-optimization is executed



[6] Stefano Primatesta et al. “A Risk-Aware Path Planning Strategy for UAVs in Urban Environments”. In: Journal of Intelligent & Robotic Systems, 2018.

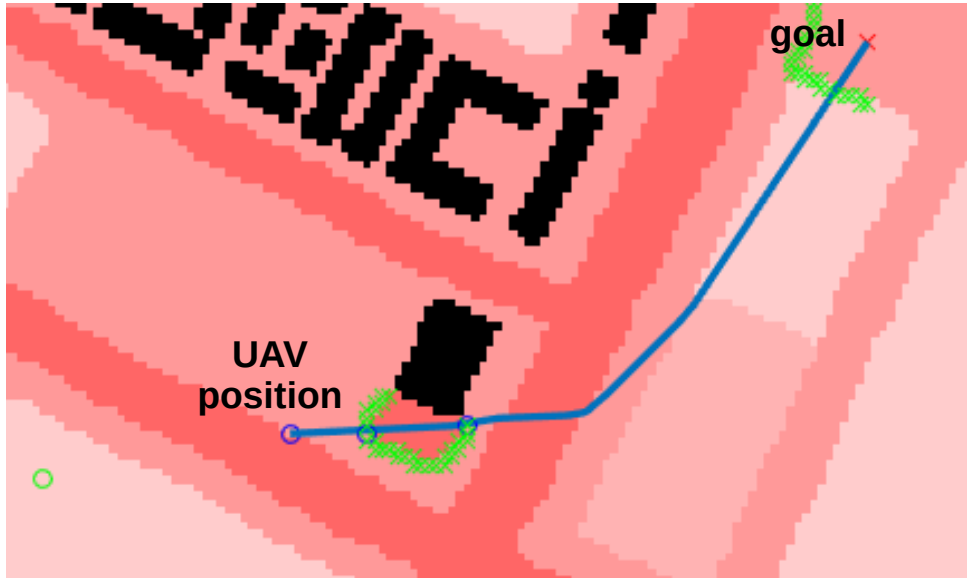
Risk-aware path planning: riskA* and Borderland results

Map		Time [s]	Path length [m]	g_{risk} cost	Average risk-cost
1	PO riskA*	0.863	674.479	189.948	0.278



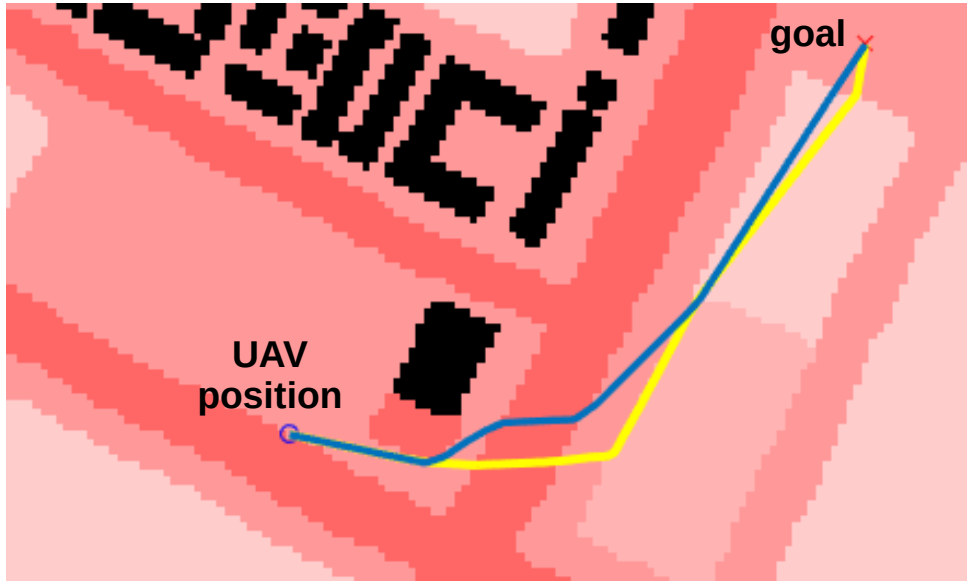
Risk-aware path planning: riskA* and Borderland results (2)

Map		Time [s]	Path length [m]	g_{risk} cost	Average risk-cost
1	PO riskA*	0.863	674.479	189.948	0.278
2	previous path	-	497.397	165.702	0.334



Risk-aware path planning: riskA* and Borderland results (3)

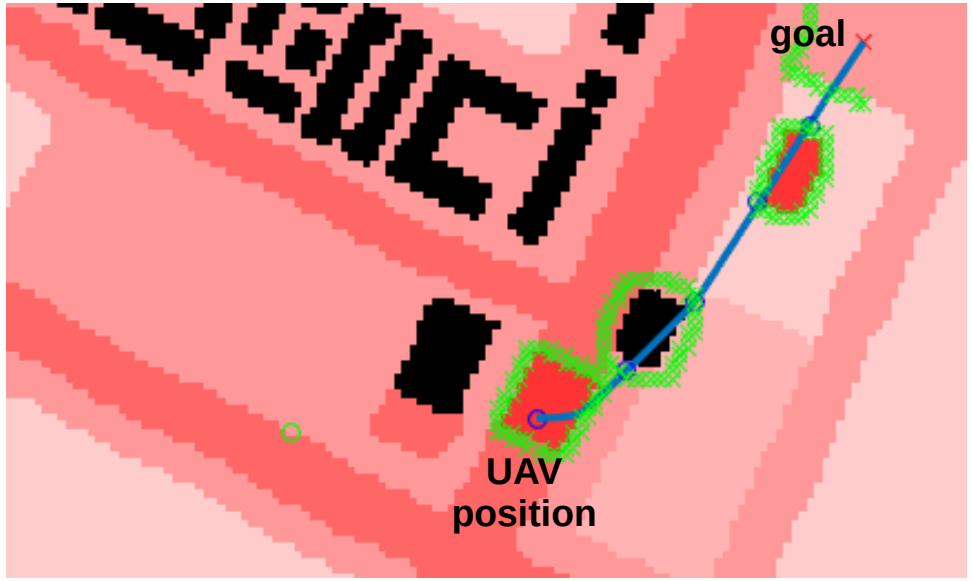
Map		Time [s]	Path length [m]	g_{risk} cost	Average risk-cost
1	PO riskA*	0.863	674.479	189.948	0.278
2	previous path	-	497.397	165.702	0.334
	PO riskA*	0.790	530.051	157.974	0.299
	Borderland	0.193 (-75.55%)	506.566 (-4.43%)	159.368 (+0.88%)	0.316 (+5.79%)



Updated path
Optimal path (offline)

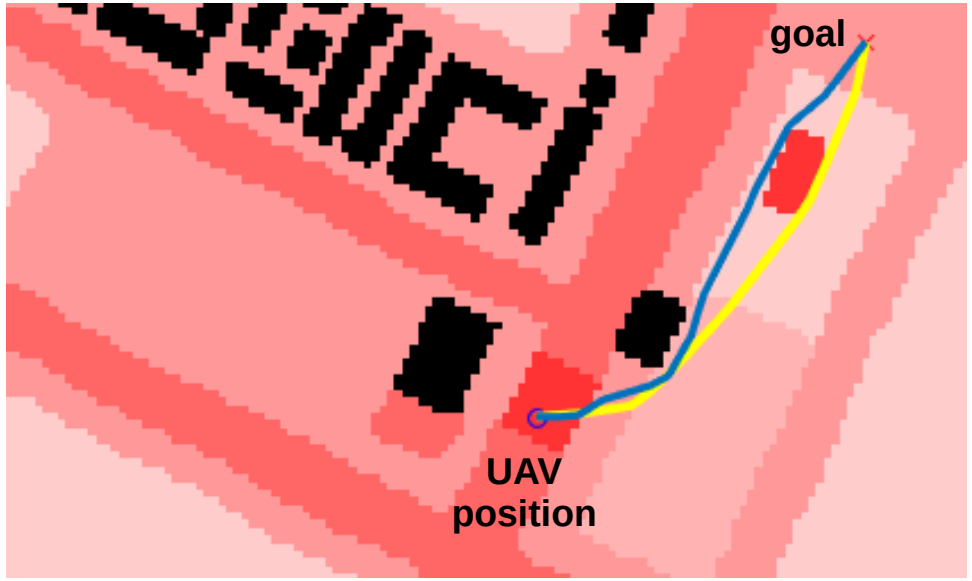
Risk-aware path planning: riskA* and Borderland results (4)

Map		Time [s]	Path length [m]	g_{risk} cost	Average risk-cost
1	PO riskA*	0.863	674.479	189.948	0.278
2	Previous path	-	497.397	165.702	0.334
	PO riskA*	0.790	530.051	157.974	0.299
	Borderland	0.193 (-75.55%)	506.566 (-4.43%)	159.368 (+0.88%)	0.316 (+5.79%)
3	Previous path	-	336.566	Invalid	Invalid



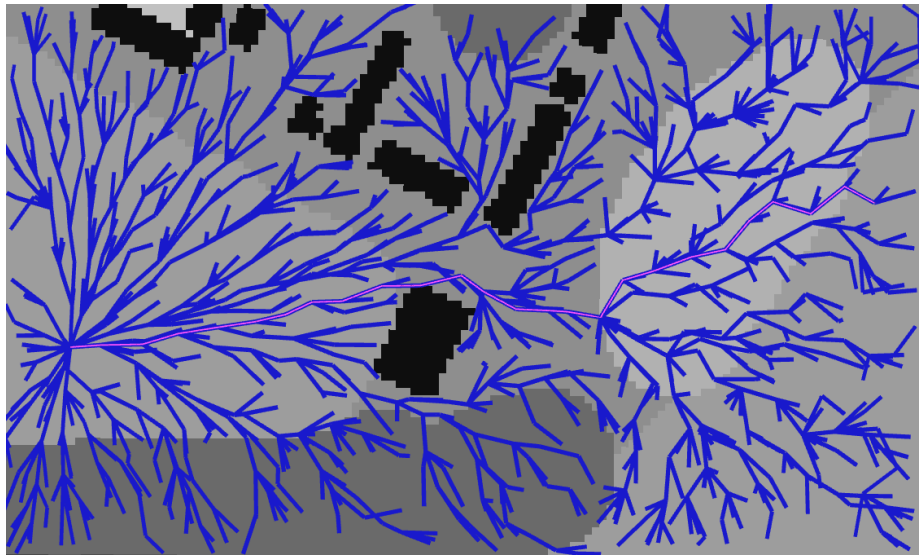
Risk-aware path planning: riskA* and Borderland results (5)

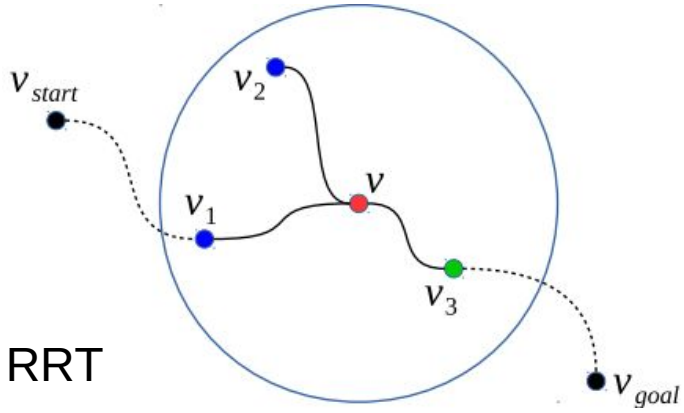
Map		Time [s]	Path length [m]	g_{risk} cost	Average risk-cost
1	PO riskA*	0.863	674.479	189.948	0.278
2	Previous path	-	497.397	165.702	0.334
	PO riskA*	0.790	530.051	157.974	0.299
	Borderland	0.193 (-75.55%)	506.566 (-4.43%)	159.368 (+0.88%)	0.316 (+5.79%)
3	Previous path	-	336.566	Invalid	Invalid
	PO riskA*	0.299	351.598	98.205	0.284
	Borderland	0.121 (-59.43%)	348.972 (-0.74%)	98.421 (+0.22)	0.286 (+0.60%)



Risk-aware path planning: riskRRT^X

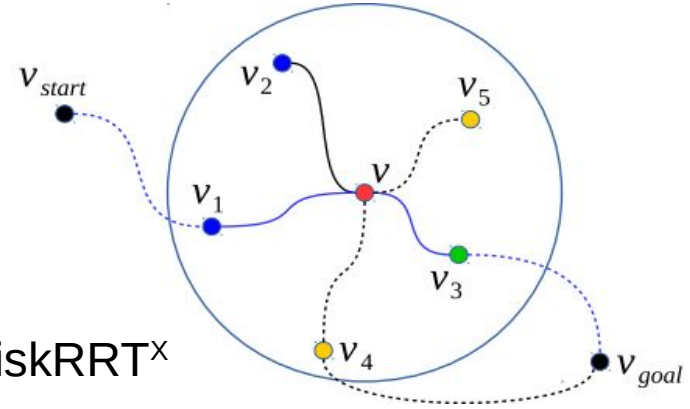
- ***RiskRRT^X*** is a path planning and re-planning algorithm
- It minimizes the risk to population on the ground
- It performs both the offline and online path planning
- It is a sample based algorithm based on RRT^X [7]
- RRT-based algorithms explore the search space with an incremental tree





RRT

$$p(v_1)=p(v_2)=v$$
$$C(v_3)\ni v$$



riskRRT^X

$$p(v_1)=p(v_2)=v$$
$$C(v_3)\ni v$$
$$N(v)=\{v_4,v_5\}$$

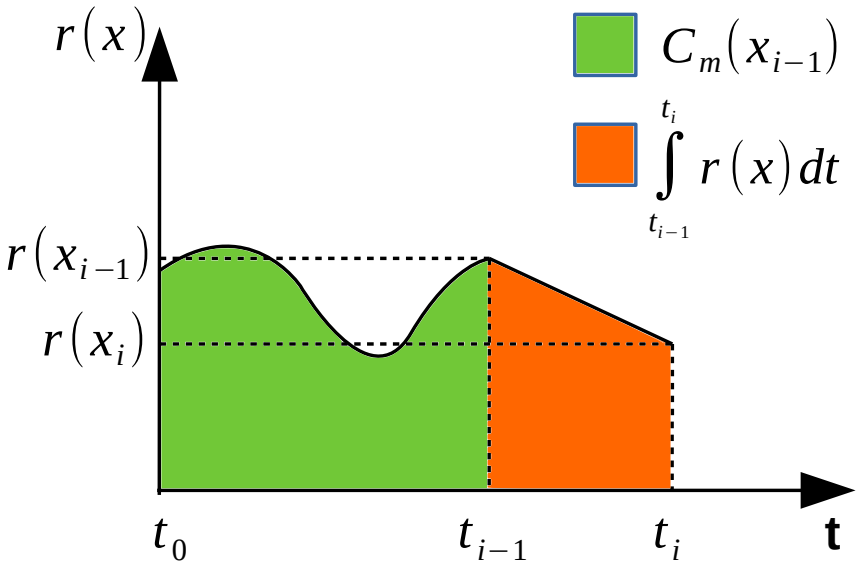
[7] M Otte and E Frazzoli. "RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning". In: The International Journal of Robotics Research, 2016.

Risk-aware path planning: riskRRT^X

- RiskRRT^X minimizes the risk to the population on the ground
- We use the concept of *time reliance*, i.e. the probability to cause a casualty is proportional to how long the person is exposed to the risk
- RiskRRT^X considers the risk in respect of the flight time
- The algorithm explores the search space minimizing the motion cost $C_m(x)$ in the graph

$$C_m(x_i) = C_m(x_{i-1}) + \int_{t_{i-1}}^{t_i} r(x)\,dt$$

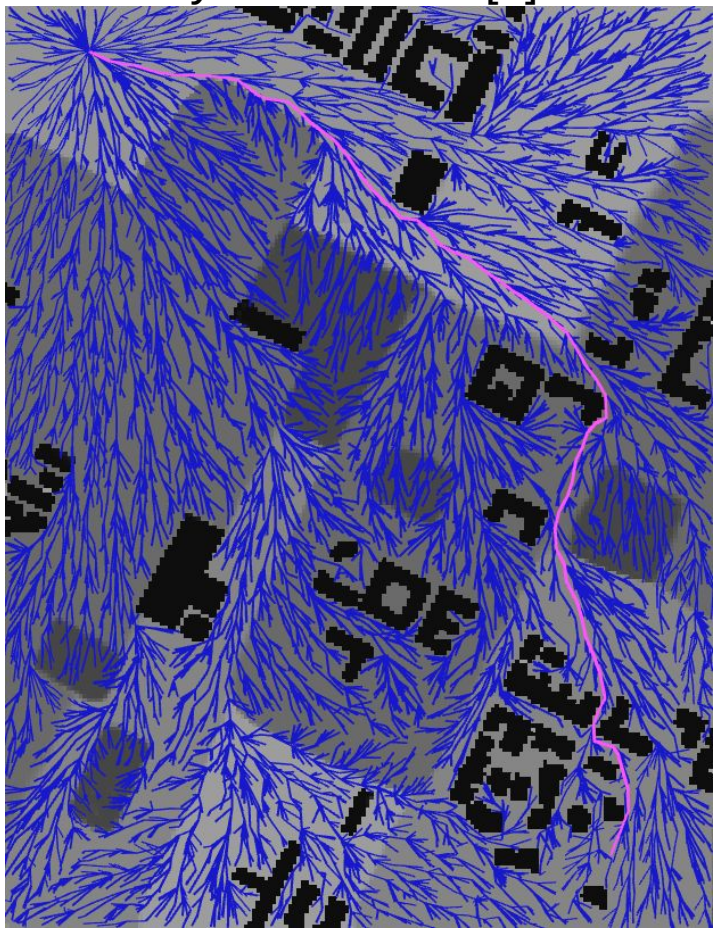
$$C_m(x_i) = C_m(x_{i-1}) + \frac{r(x_{i-1}) + r(x_i)}{2}\, t(x_{i-1}, x_i)$$



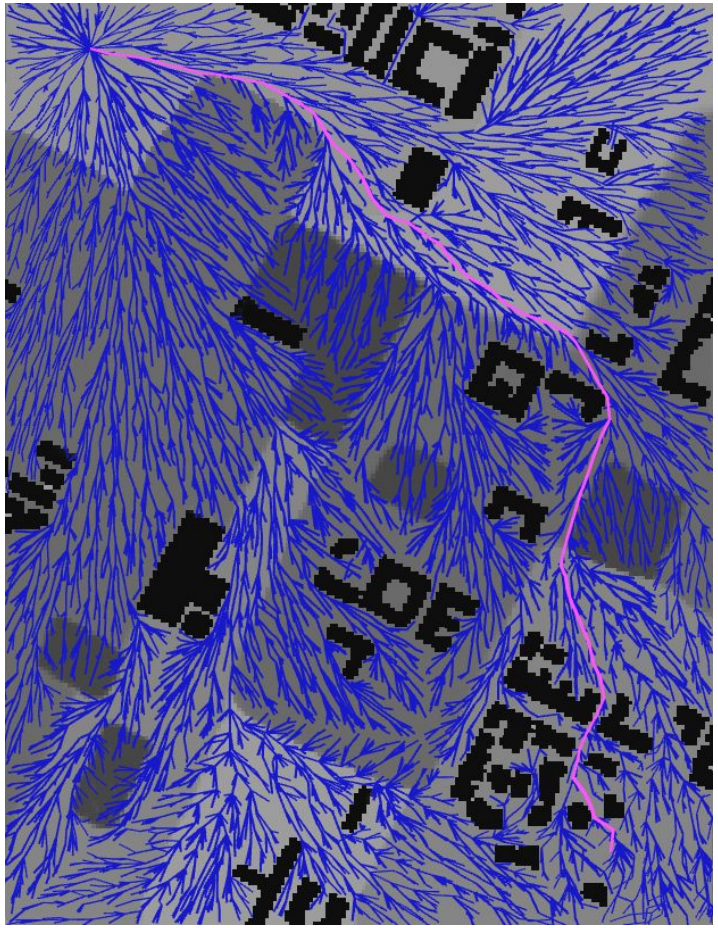
Risk-aware path planning: riskRRT^X results

Offline phase

Dynamic RRT* [8]



riskRRT^X



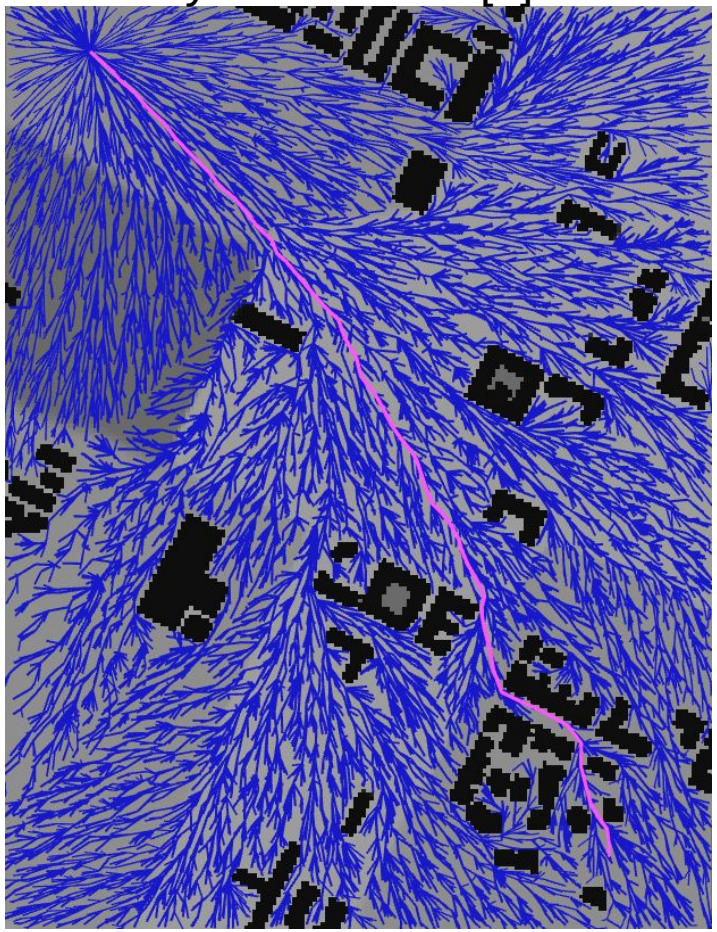
Algorithm	Nodes	Computational time (s)	Solution motion cost (h-1)
RRT*	10000	0.741	$1.233 \cdot 10^{-8}$
Dynamic RRT*	10000	0.731	$1.245 \cdot 10^{-8}$
riskRRT^X	10000	4.863	$1.237 \cdot 10^{-8}$

[8] D. Ferguson, et al. "Replanning with RRTs". In: Proceedings 2006 IEEE ICRA, 2006.

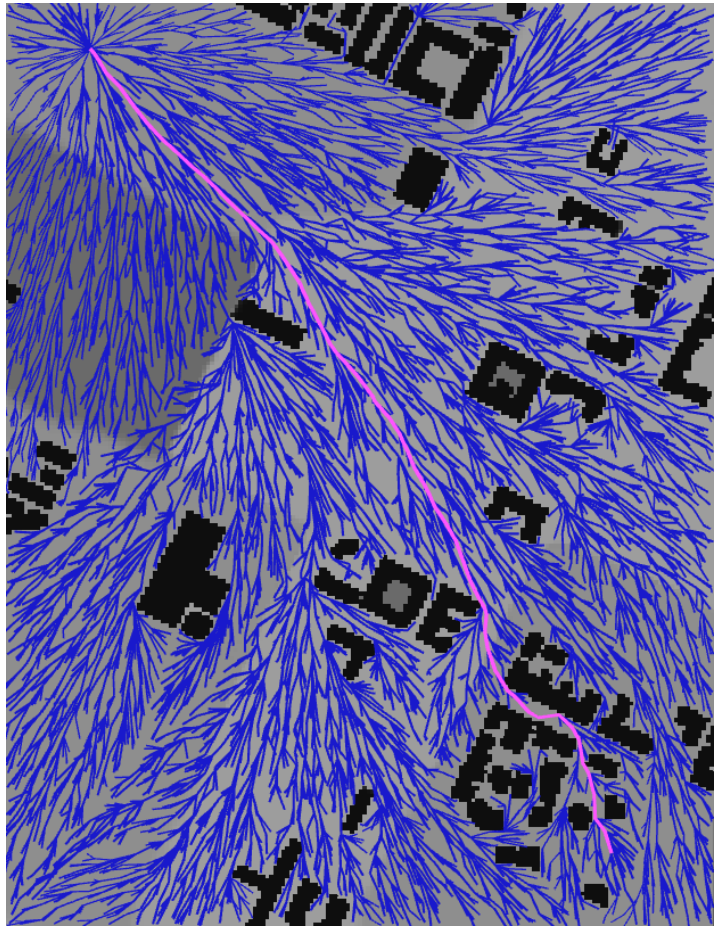
Risk-aware path planning: riskRRT^X results

Online phase: Scenario 1

Dynamic RRT* [8]



riskRRT^X



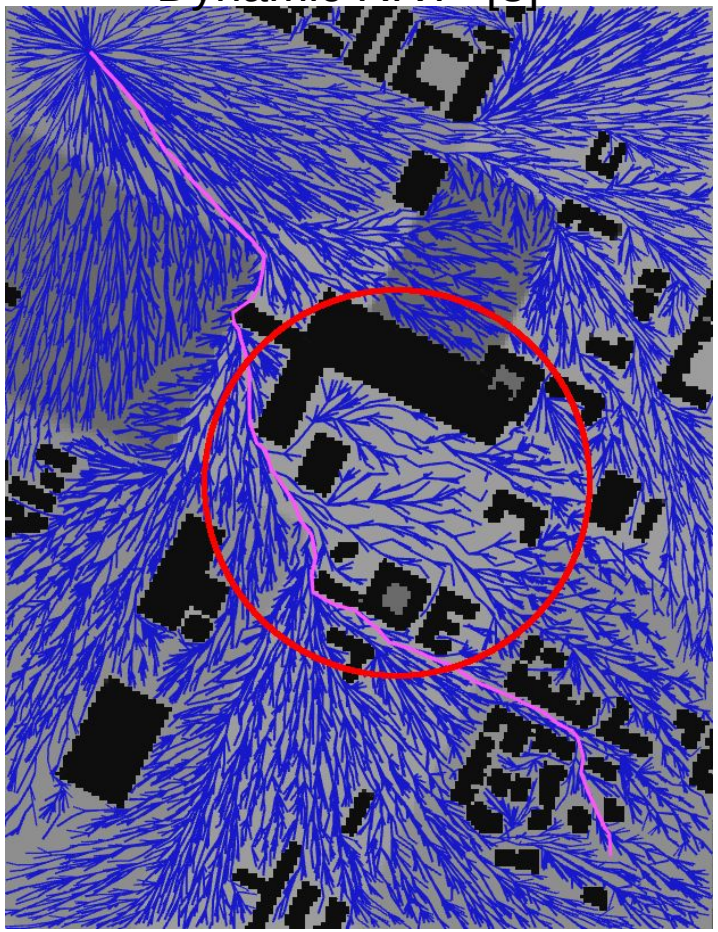
Algorithm	Nodes	Computational time (s)	Solution motion cost (h-1)
RRT*	10000	0.738	$0.841 \cdot 10^{-8}$
Dynamic RRT*	10000	0.649	$0.840 \cdot 10^{-8}$
riskRRT^X	10000	0.482	$0.844 \cdot 10^{-8}$

[8] D. Ferguson, et al. "Replanning with RRTs". In: Proceedings 2006 IEEE ICRA, 2006.

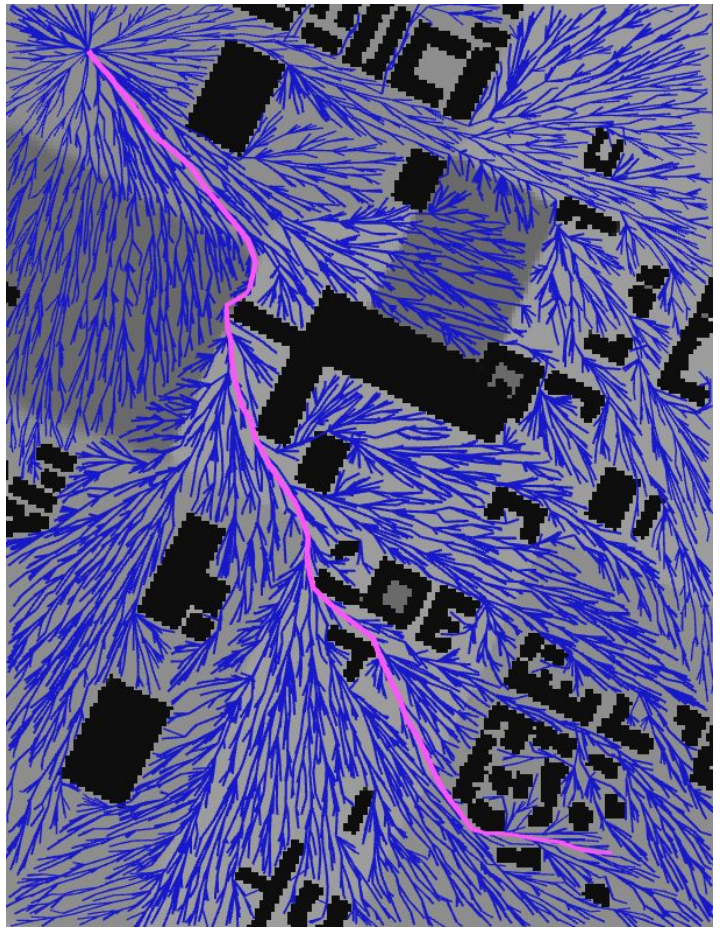
Risk-aware path planning: riskRRT^X results

Online phase: Scenario 2

Dynamic RRT* [8]



riskRRT^X



Algorithm	Nodes	Computational time (s)	Solution risk cost (h-1)
RRT*	10000	0.810	$0.926 \cdot 10^{-8}$
Dynamic RRT*	13352	0.701	$0.925 \cdot 10^{-8}$
riskRRT^X	8327	0.449	$0.932 \cdot 10^{-8}$

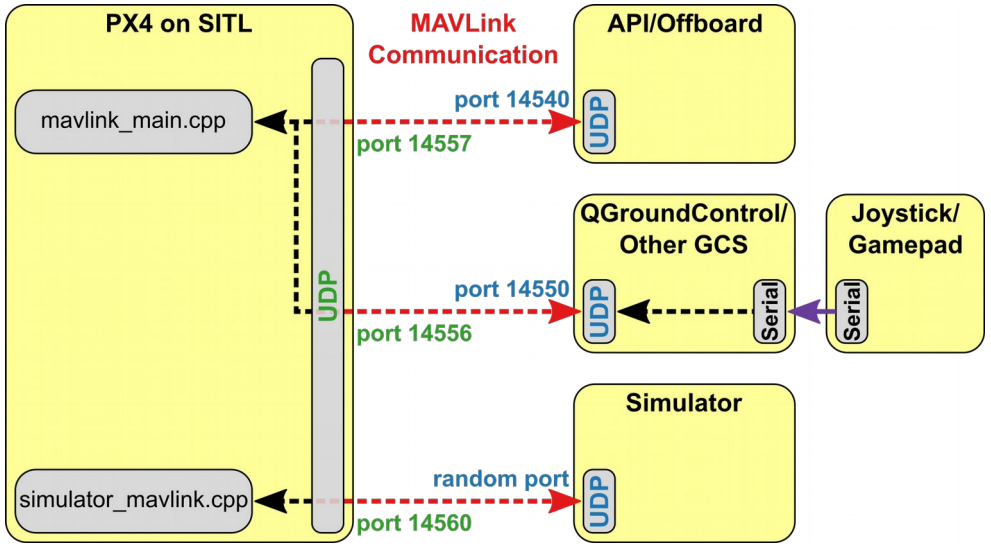
[8] D. Ferguson, et al. "Replanning with RRTs". In: Proceedings 2006 IEEE ICRA, 2006.

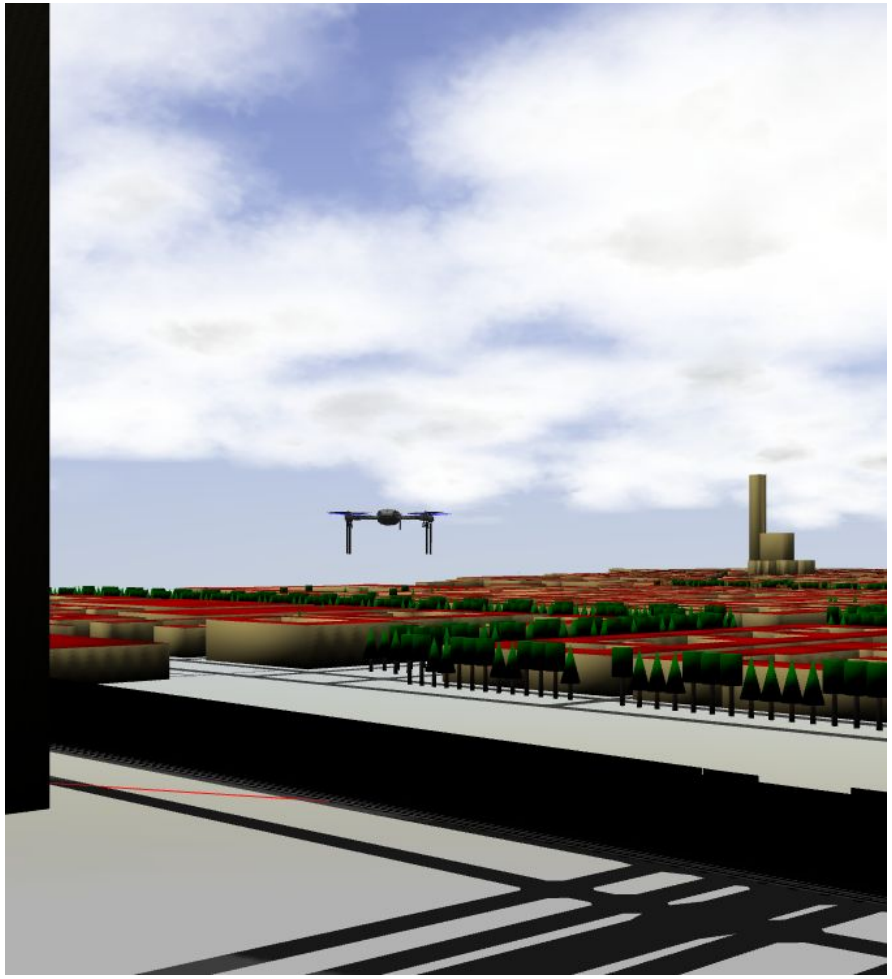
Outline

- Introduction
- **Safe Navigation for UAS in urban areas**
  - Cloud-based framework for UASs
  - Risk-based map
  - Risk-aware path planning
  - **Simulation**
- Autonomous navigation for ground robots
  - Cloud-based architecture for Service Robotics Applications
  - Dynamic path planning
  - Motion control with PF-MPEPC
  - Service Robotics Applications
- Conclusions

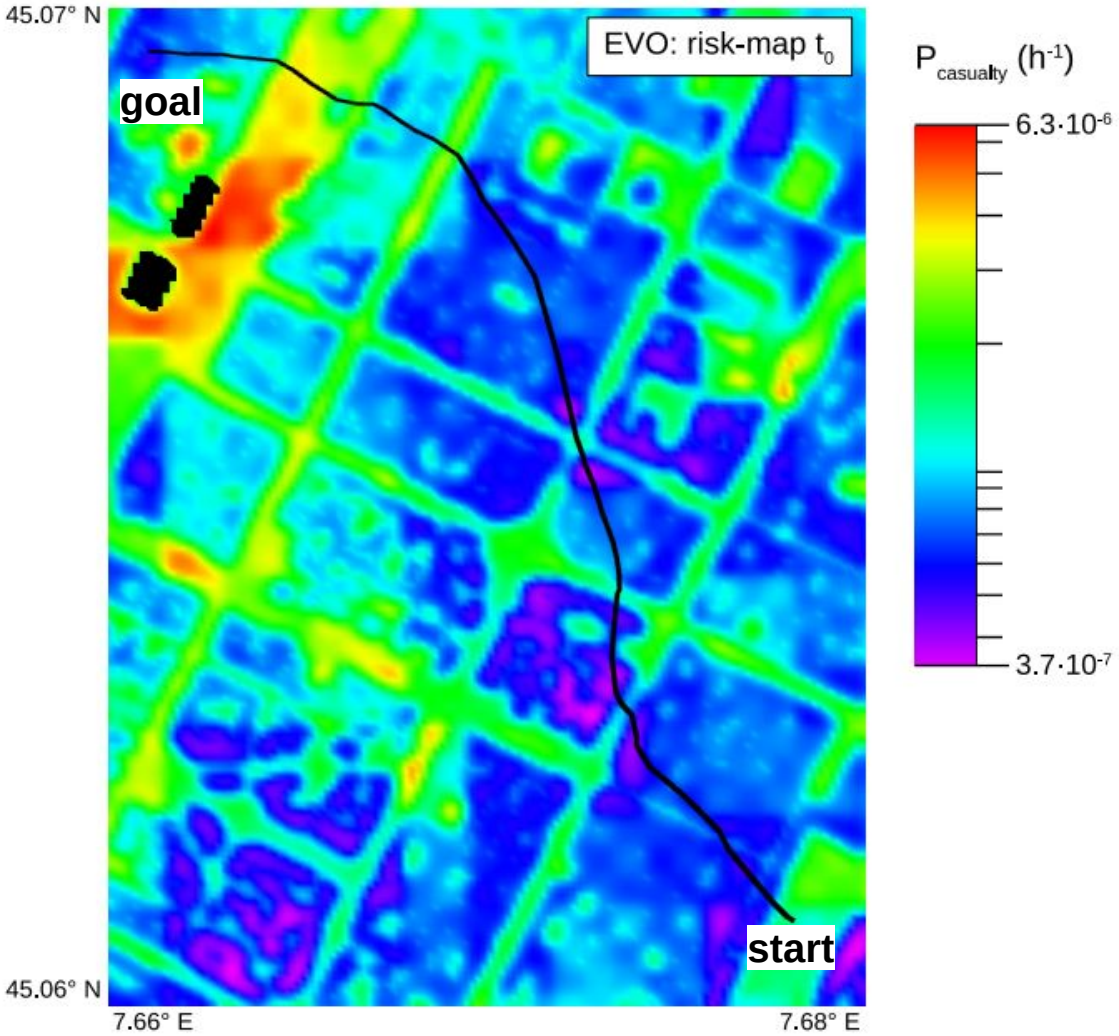
Safe navigation for UAS: simulation

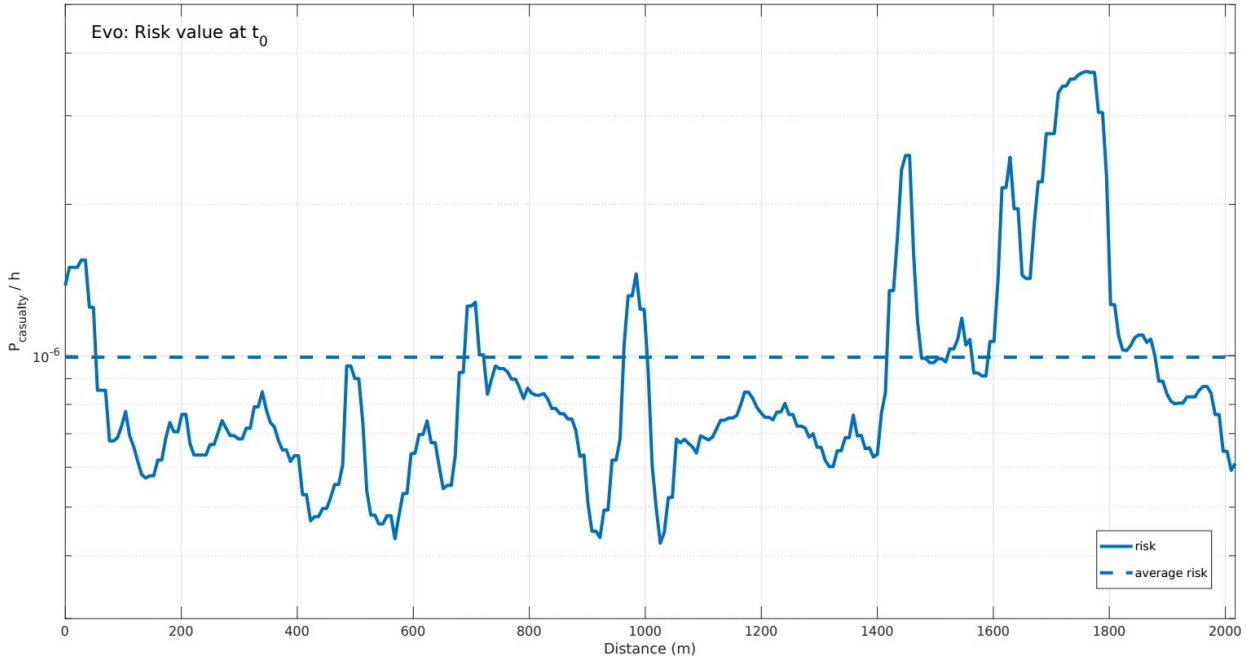
- A simulation of a flight mission in an urban area is performed
- The simulation is performed using the Robot Operating System (ROS) and Gazebo
- The simulated quadcopter uses the PX4 flight control, simulated using the SITL (Software In The Loop) framework





Safe navigation for UAS: simulation (2)

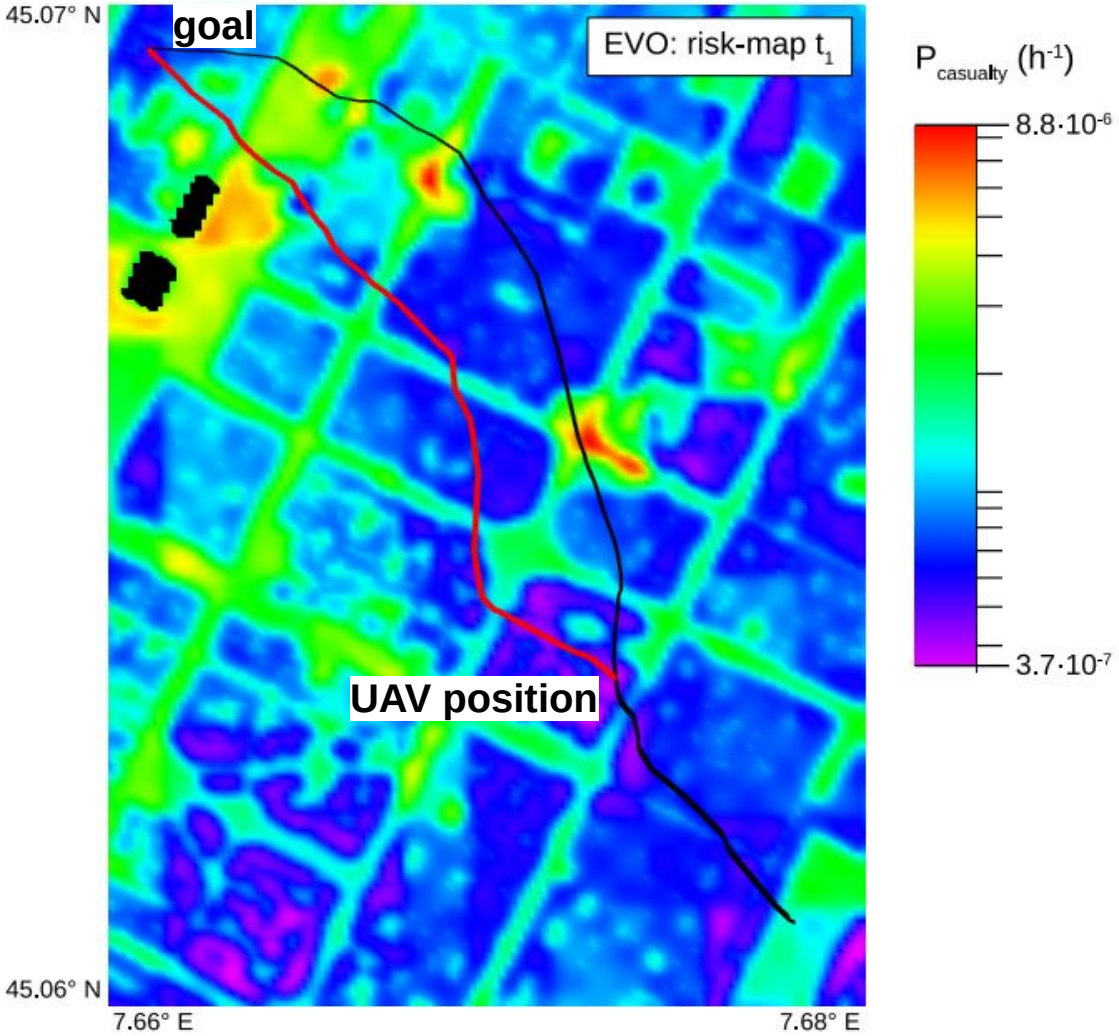


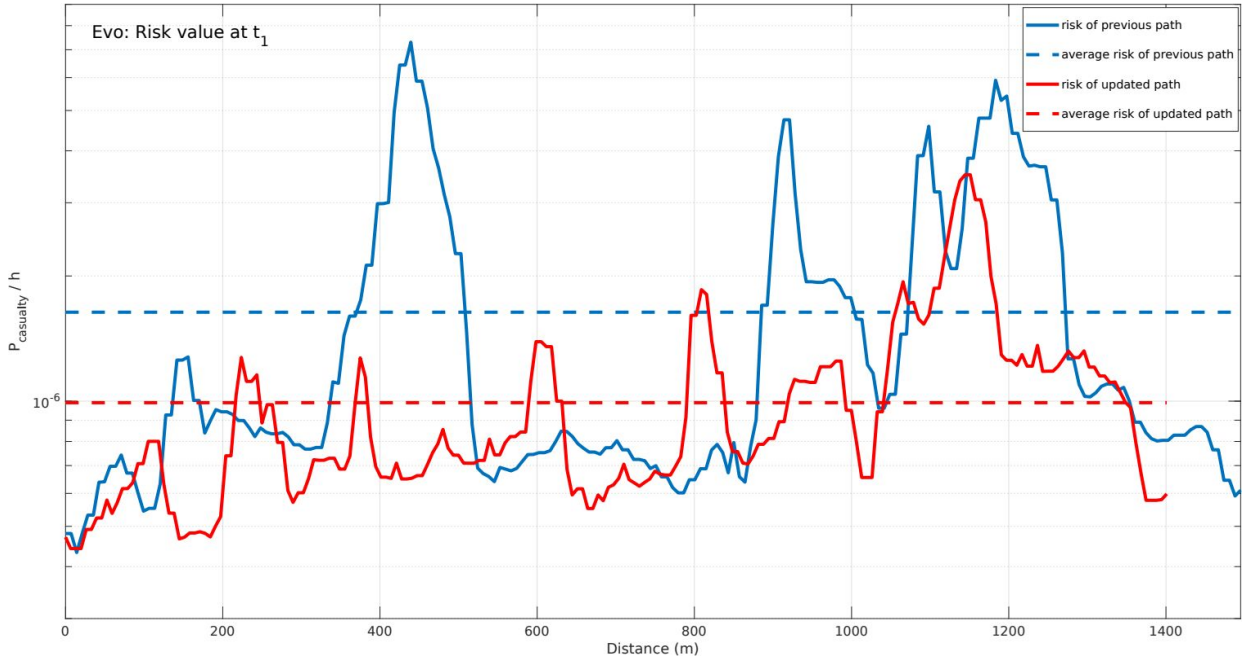


Offline phase with riskRRT^X

Average risk of $9.93 \cdot 10^{-7}$ h^{-1} < **ELOS** ($1 \cdot 10^{-6}$ h^{-1})

Safe navigation for UAS: simulation (3)





Online phase with riskRRT^X (0.614 s)

Old path: Average risk of $1.64 \cdot 10^{-6}$ h^{-1} > **ELOS** ($1 \cdot 10^{-6}$ h^{-1})

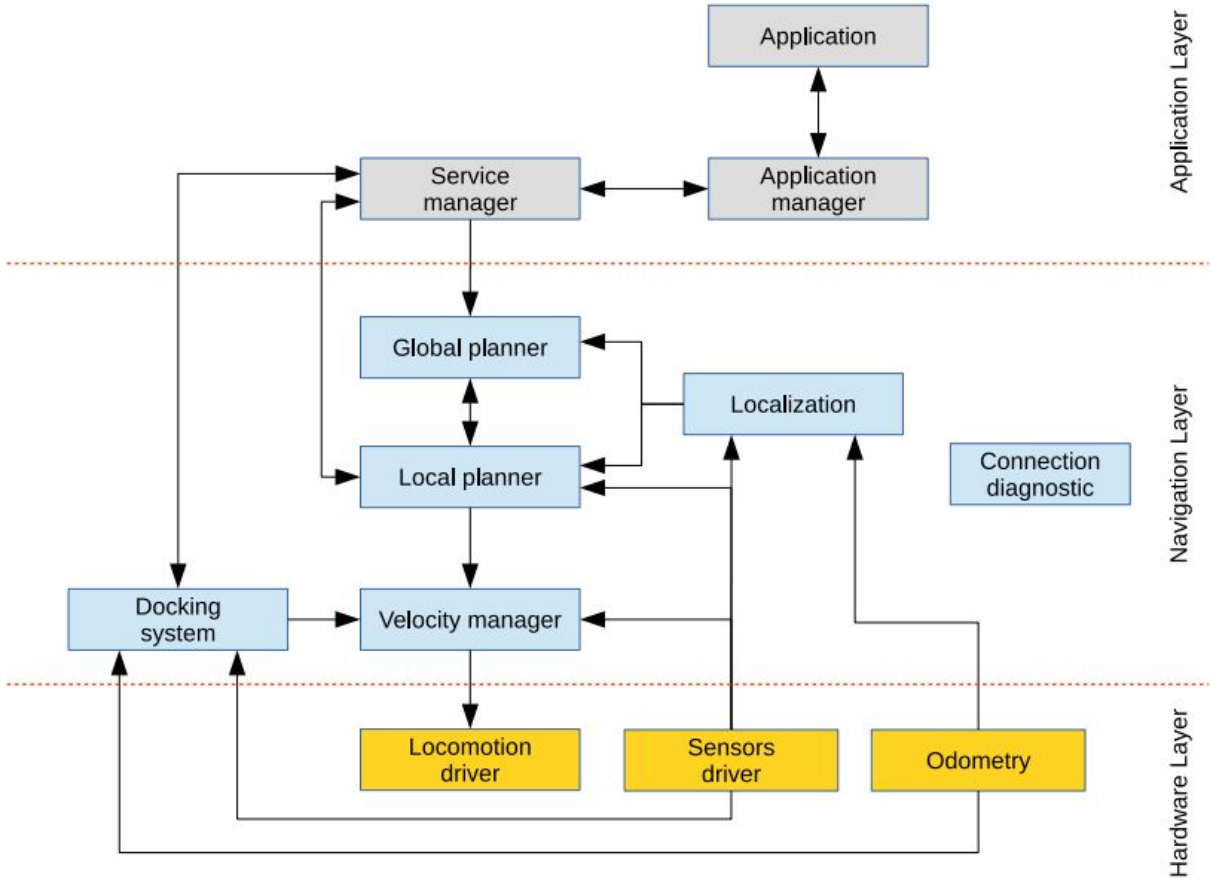
New path: Average risk of $9.91 \cdot 10^{-7}$ h^{-1} < **ELOS** ($1 \cdot 10^{-6}$ h^{-1})

Outline

- Introduction
- Safe Navigation for UAS in urban areas
  - Cloud-based framework for UASs
  - Risk-based map
  - Risk-aware path planning
  - Simulation
- **Autonomous navigation for ground robots**
  - **Cloud-based architecture for Service Robotics Applications**
  - Dynamic path planning
  - Motion control with PF-MPEPC
  - Service Robotics Applications
- Conclusions

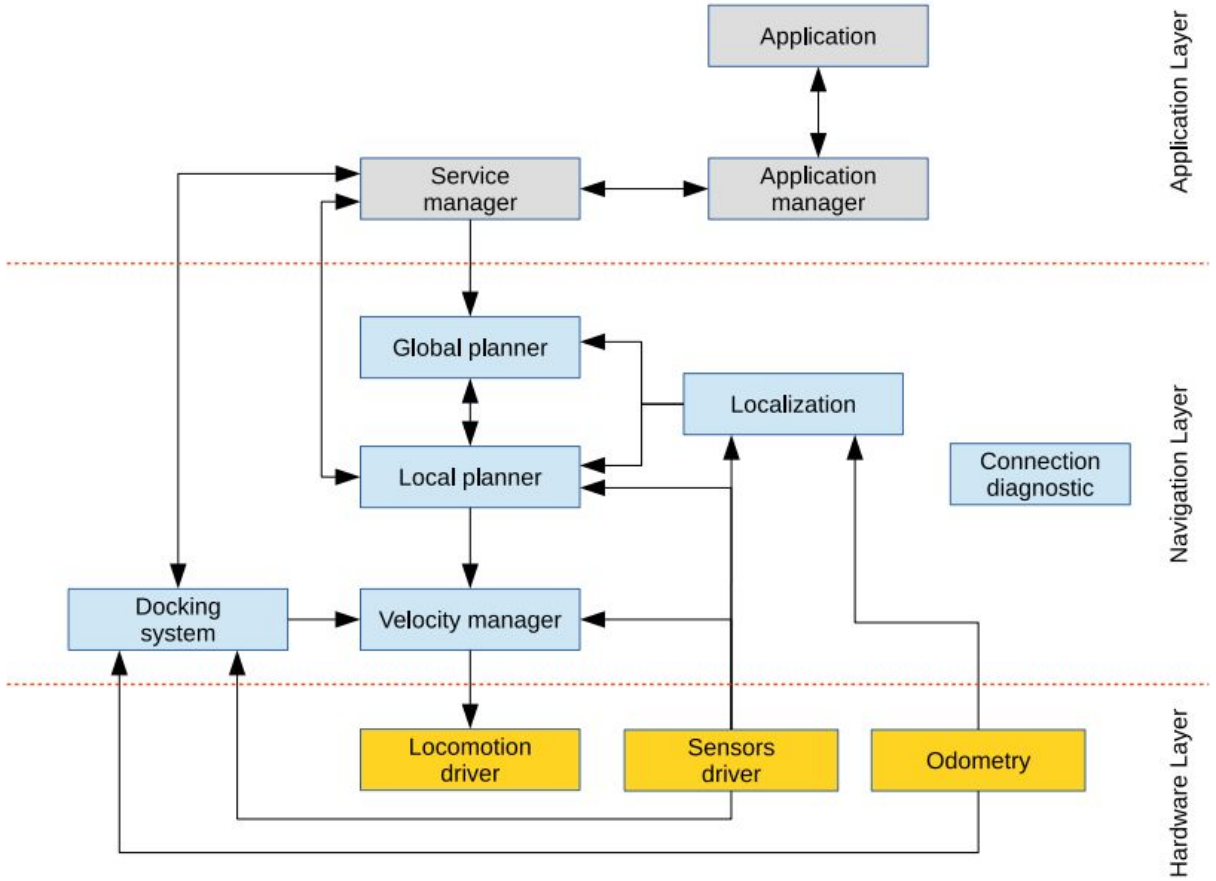
Cloud-based architecture for Service Robotics Applications

- The aim is to propose a reference framework to offer service robotics applications
- Intensive use of Cloud technologies
- The robot is connected with the Cloud using mobile networks (4G, 5G)
- ROS-compatible
- Three layers:
  - Application layer
  - Navigation layer
  - Hardware layer



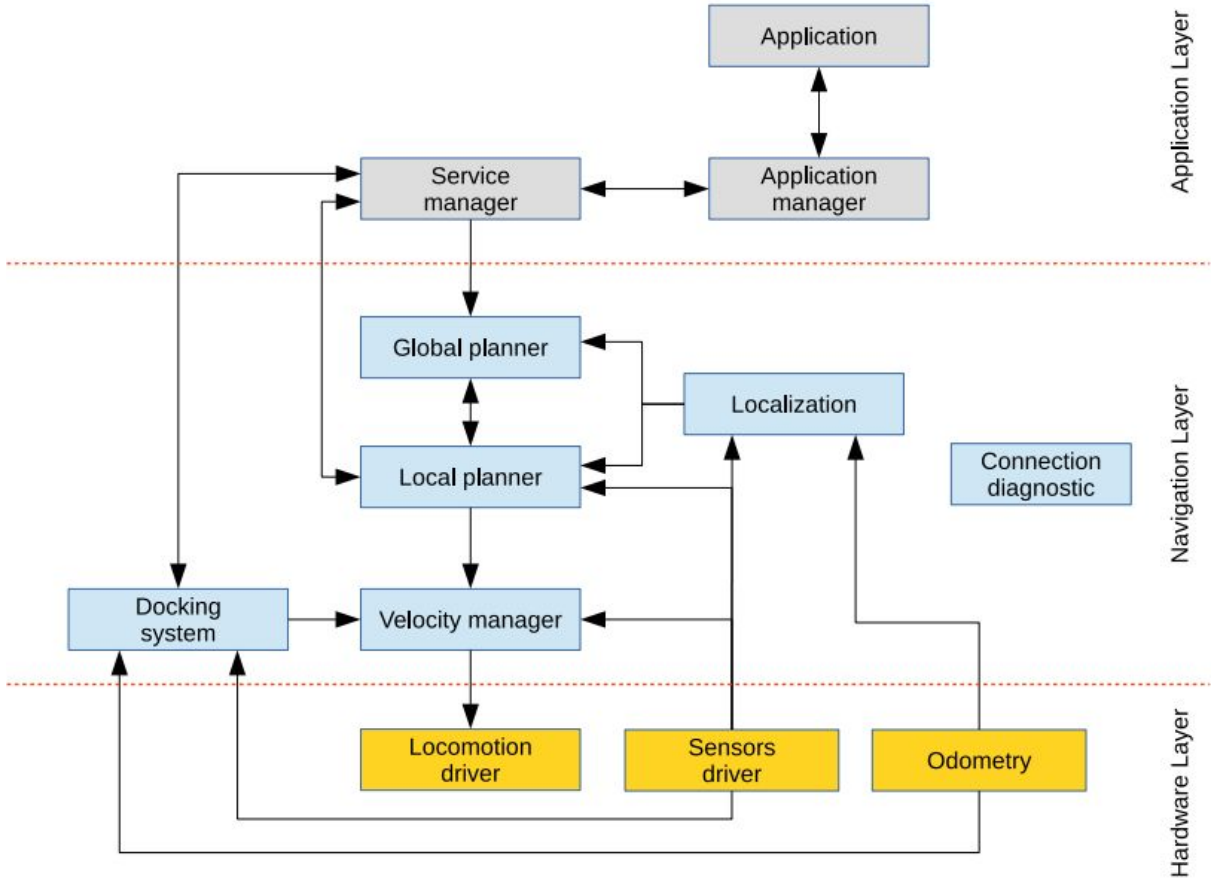
Cloud-based architecture for Service Robotics Applications

- The ***Application layer*** manages the service robotics application
  - It is allocated on-Cloud
- The ***Navigation layer*** provides the autonomous navigation, in order to provide the Service
  - It is distributed between the Cloud and the robot
- The ***Hardware layer*** represents the mobile robot platform including both hardware and software
  - It resides on-board the robot



Cloud-based architecture for Service Robotics Applications

- In this thesis, two methods for the autonomous navigation in crowded environments are proposed:
  - A dynamic path planning based on Informed-RRT* [9]
  - A Motion control with Particle Filter Model Predictive Equilibrium Point Control (PF-MPEPC) approach [10]



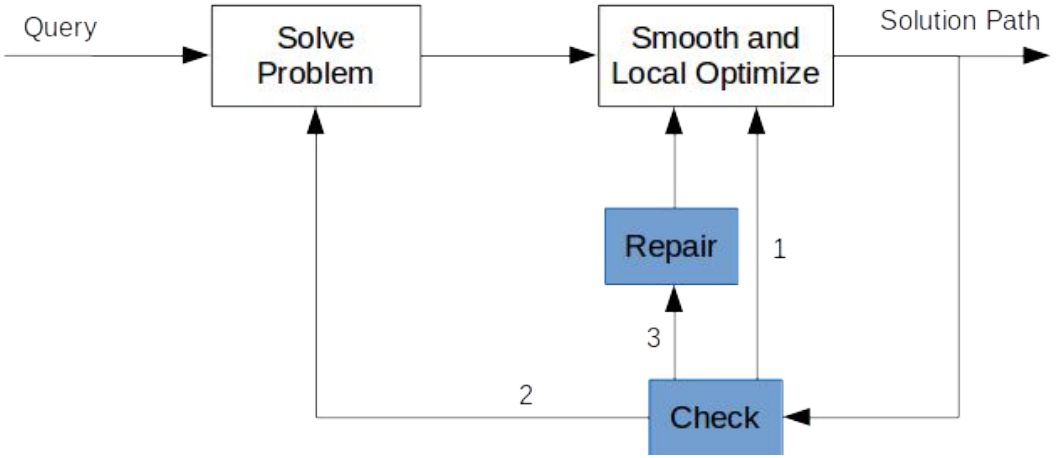
[9] S Primatesta et al. "Dynamic trajectory planning for mobile robot navigation in crowded environments". In: 2016 IEEE ETFA, 2016.
[10] S Primatesta and B Bona. "Motion control of mobile robots with particle filter model predictive equilibrium point control". In: 2017 IEEE ICARSC. 2017.

Outline

- Introduction
- Safe Navigation for UAS in urban areas
  - Cloud-based framework for UASs
  - Risk-based map
  - Risk-aware path planning
  - Simulation
- **Autonomous navigation for ground robots**
  - Cloud-based architecture for Service Robotics Applications
  - **Dynamic path planning**
  - Motion control with PF-MPEPC
  - Service Robotics Applications
- Conclusions

Dynamic Path Planning

- A new framework for dynamic path planning in crowded environments is proposed
- People are assumed as static obstacles
- The search space is continuously updated
- It is based on a *check and repair* method.
  - Initially, the global path is computed to reach the desired pose
  - While the robot is executing the path, the algorithm continuously verifies and updates (if necessary) the path
- A safe and valid path is always computed
- The proposed solution uses the Informed-RRT* algorithm [11]



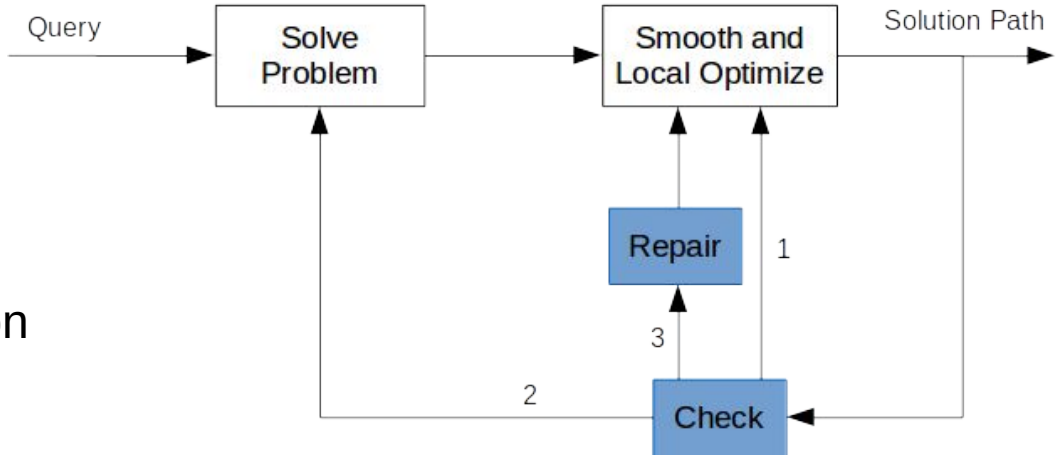
[11] J D Gammell at al. "Informed RRT: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic". In: 2014 IEEE/RSJ IROS, 2014.*

Dynamic Path Planning: check

- At each iteration, the algorithm computes a score χ(σ)

$$\chi(\sigma)=\frac{|\{V_{obs}, E_{obs} \in ([s_k, 1])\}|}{|\{V, E \in \sigma\}|}$$

  - σ is the current path
  - V, E are the vertices and edges of the path σ
  - V_{obs}, E_{obs} are the invalid vertices and edges of the path not already executed
  - s_k is element of the path corresponding to the actual position of the robot



- Given a **threshold** χ_{max}, three cases are defined:

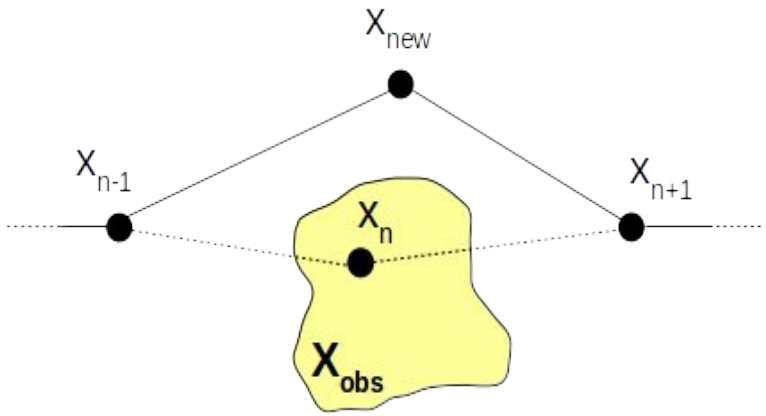
1) Valid path if $\chi(\sigma) = 0$

2) Invalid path if $\chi(\sigma) \geq \chi_{max}$

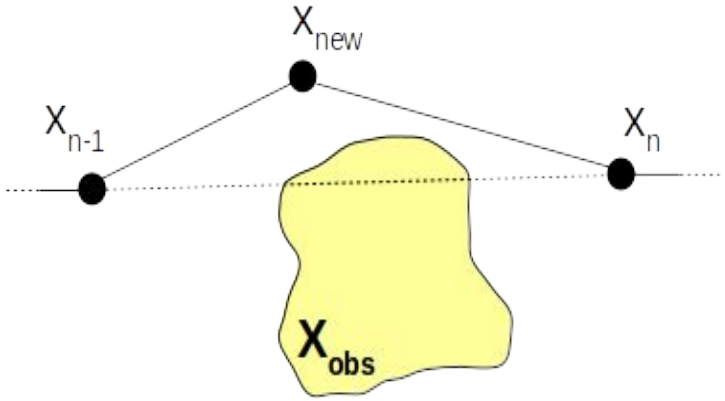
3) Invalid path, but repairable if $0 < \chi(\sigma) < \chi_{max}$

Dynamic Path Planning: repair

- If the solution path is invalid but repairable, we try to repair invalid edges (E_{obs}) and vertexes ($\mathrm{V}_{\mathrm{obs}}$)
- In the simplest case:

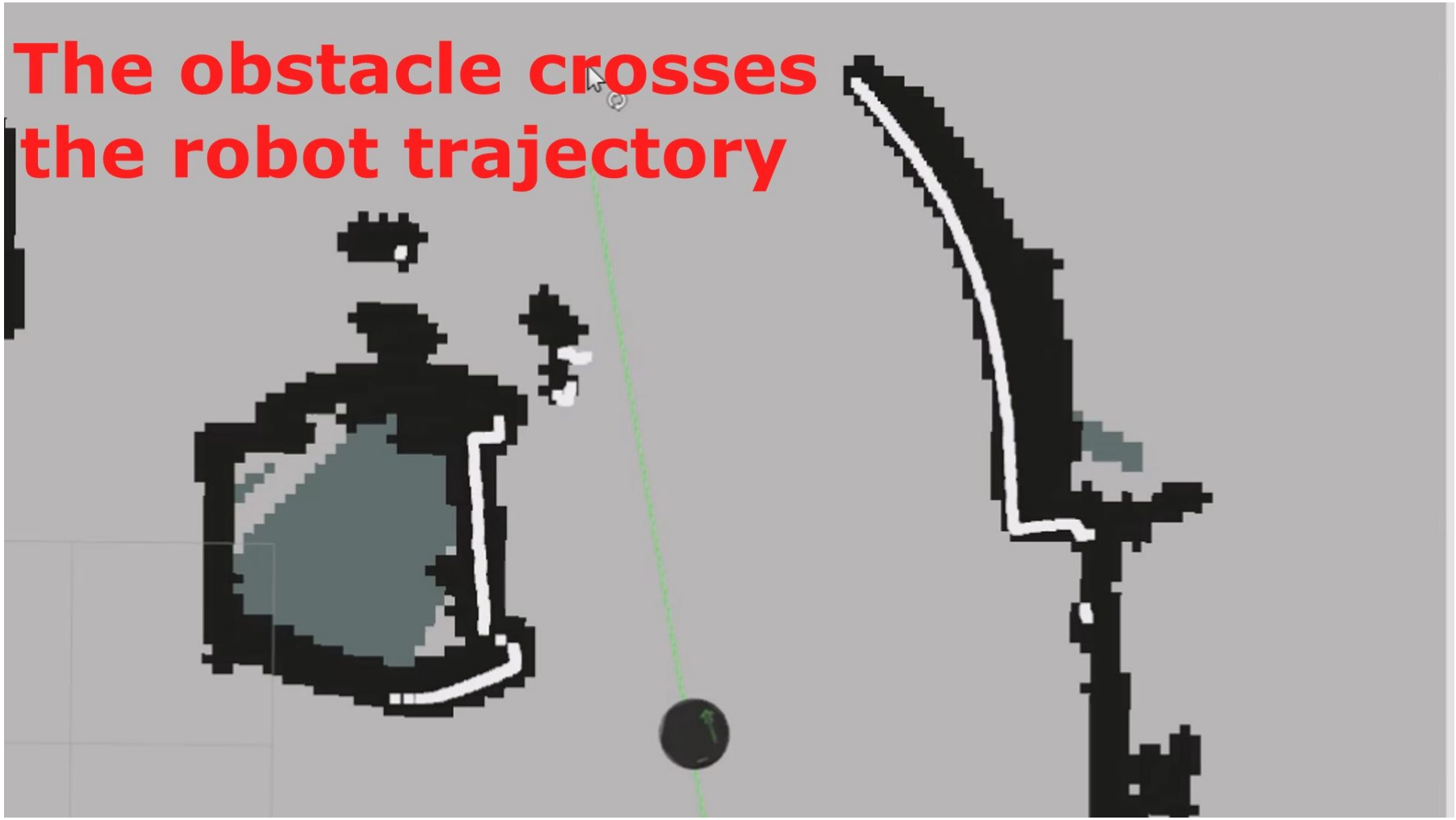


Invalid state



Invalid edge

Dynamic Path Planning: Results

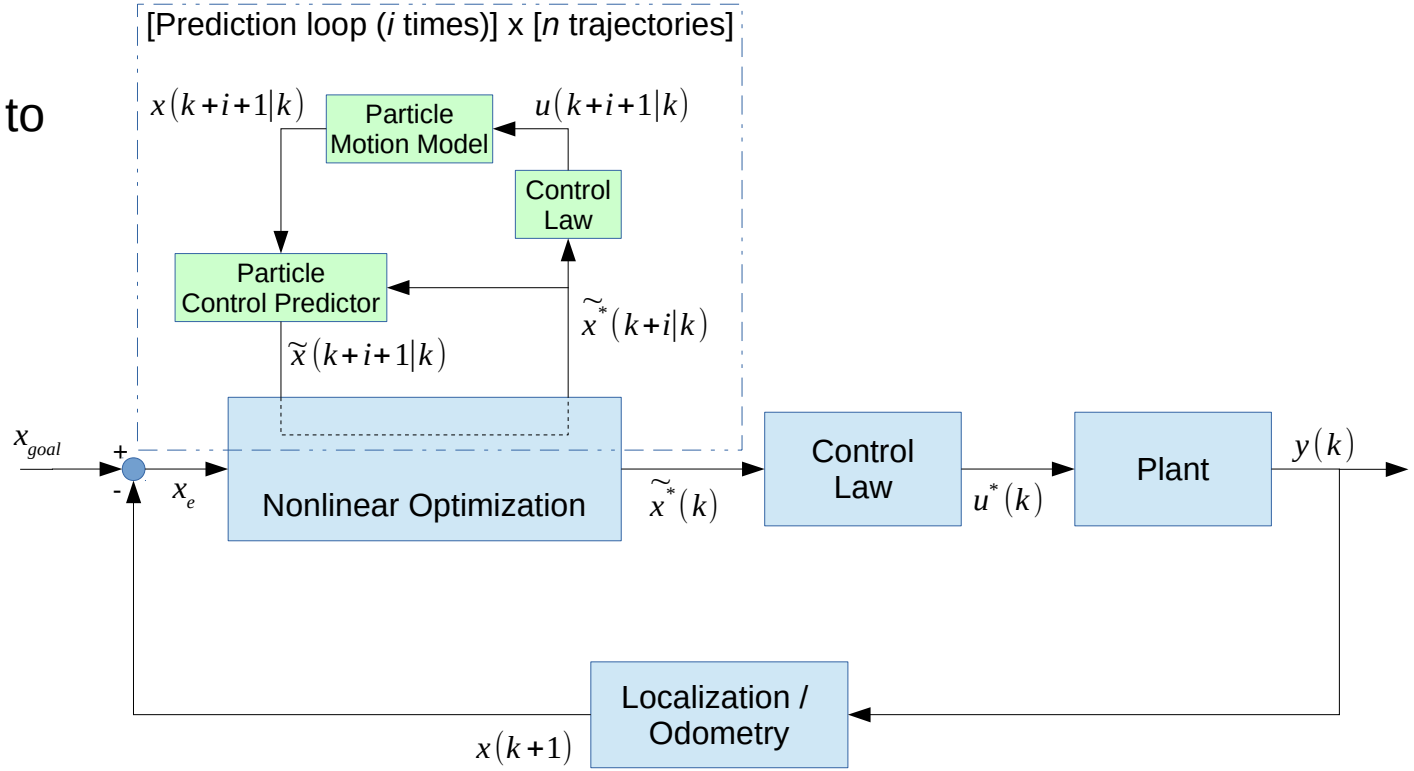


Outline

- Introduction
- Safe Navigation for UAS in urban areas
  - Cloud-based framework for UASs
  - Risk-based map
  - Risk-aware path planning
  - Simulation
- **Autonomous navigation for ground robots**
  - Cloud-based architecture for Service Robotics Applications
  - Dynamic path planning
  - **Motion control with PF-MPEPC**
  - Service Robotics Applications
- Conclusions

Motion Control with PF-MPEPC

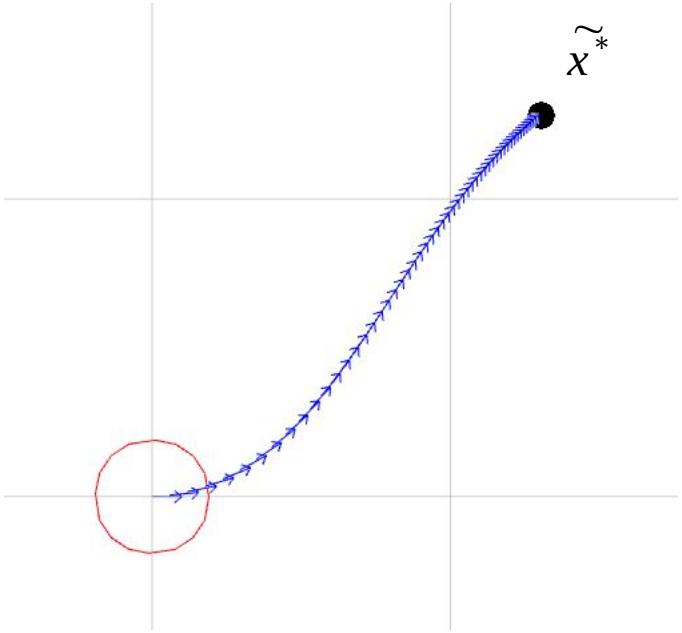
- An optimal motion controller for mobile robots, called ***Particle Filter Model Predictive Equilibrium Point Control (PF-MPEPC)*** is proposed
- Model Predictive Control approach
- The Equilibrium Point Control Method is used to define an optimal trajectory
- Particle filters evaluates the uncertainty due to disturbances, improving safety



Motion Control with PF-MPEPC: MPEPC

- In this approach, the Equilibrium Point Control method is applied to the Model Predictive Control

$$\tilde{x}^*(k) = \min_{U(k)} \quad J(x(k|k), \tilde{x})$$
$$\text{subject} \quad x(k+1) = f(x(k), \pi(x(k), \tilde{x}))$$
$$x(k+i|k) \in \boldsymbol{X}$$

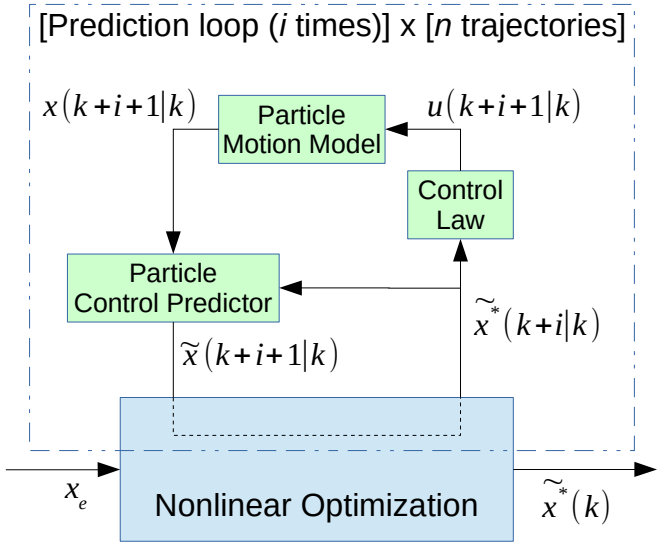


- The Control Law defines the resulting trajectory:

$$u(k) = \pi(x(k), \tilde{x})$$

Motion Control with PF-MPEPC

- Particle filters are used in the prediction loop to evaluate measurement noise and disturbances
- The ***Particle Motion Model*** is a state estimator. Given a control input, it estimates the robot pose at time k+1
- The ***Particle Control Predictor*** is a predictor for the control input of the model. It updates the equilibrium point, i.e. the control input of the proposed approach.



Motion Control with PF-MPEPC: Cost function

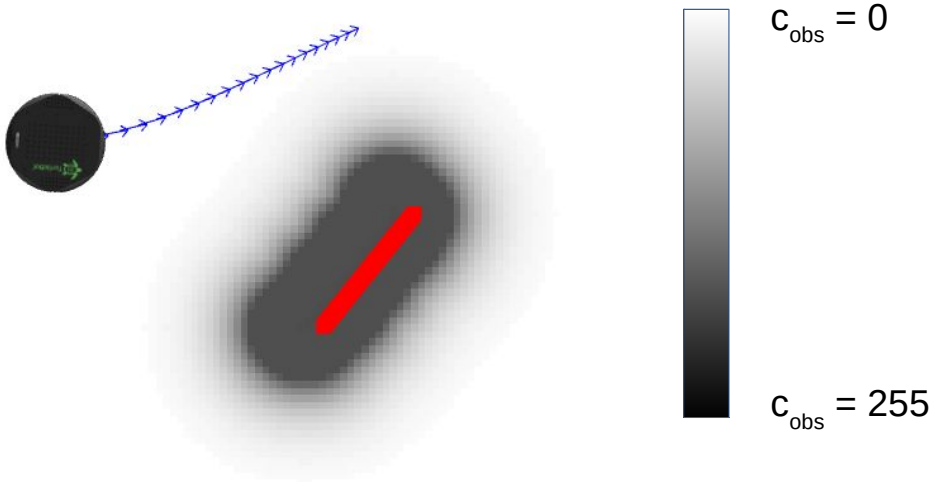
$$J(U(k), x(k|k)) = \sum_{i=0}^{H_p-1} \left[\lambda_1 x_e(k+1|k)^2 + \lambda_2 \omega(k+i-1|k)^2 + \lambda_3 \Delta U(k+i-1|k)^2 + \lambda_4 c_{obs}(\hat{x}(k+i|k)^2) \right]$$

Position error x_e

Angular velocity ω

Accelerations ΔU

Obstacle cost c_{obs}



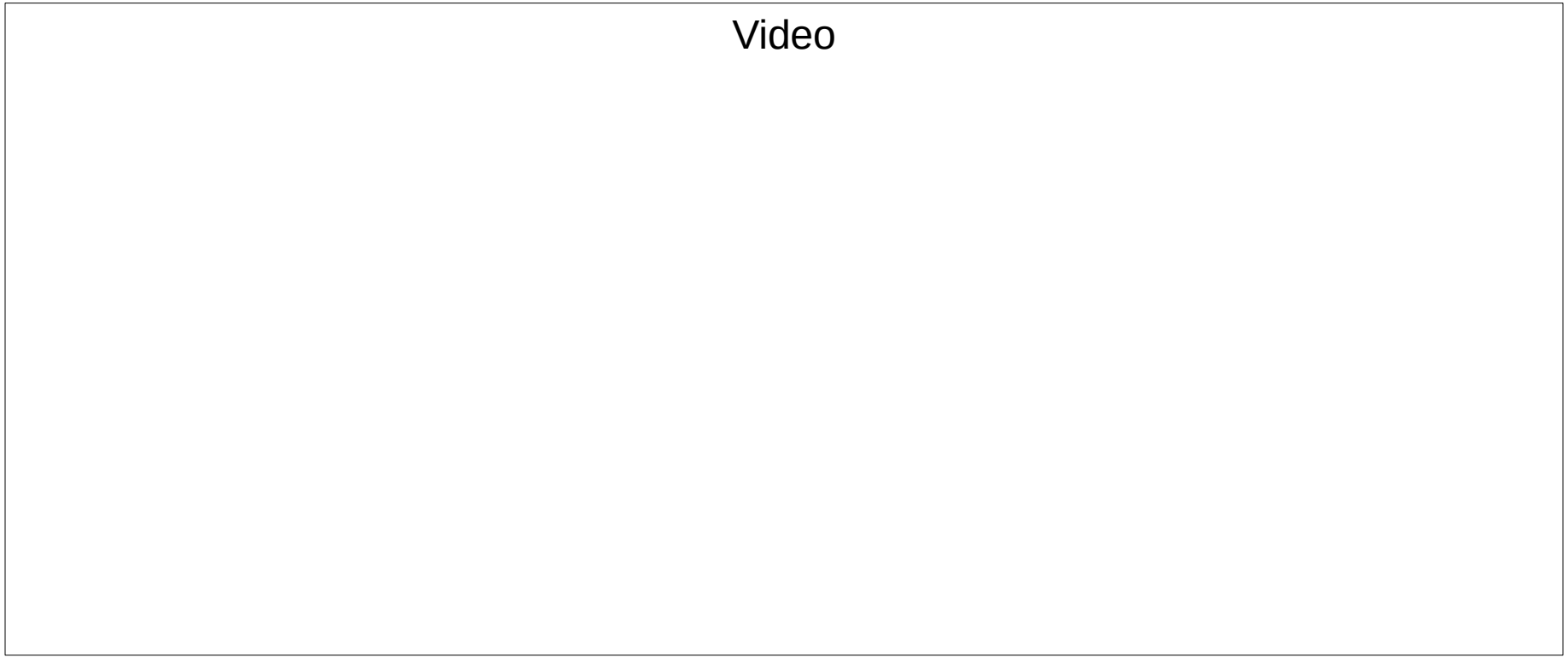
c_{obs} = 0

c_{obs} = 255

Control constraints:

$$U_{min} \leq U(k) \leq U_{max}$$
$$\Delta U_{min} \leq \Delta U(k) \leq \Delta U_{max}$$

Motion Control with PF-MPEPC: Results



Outline

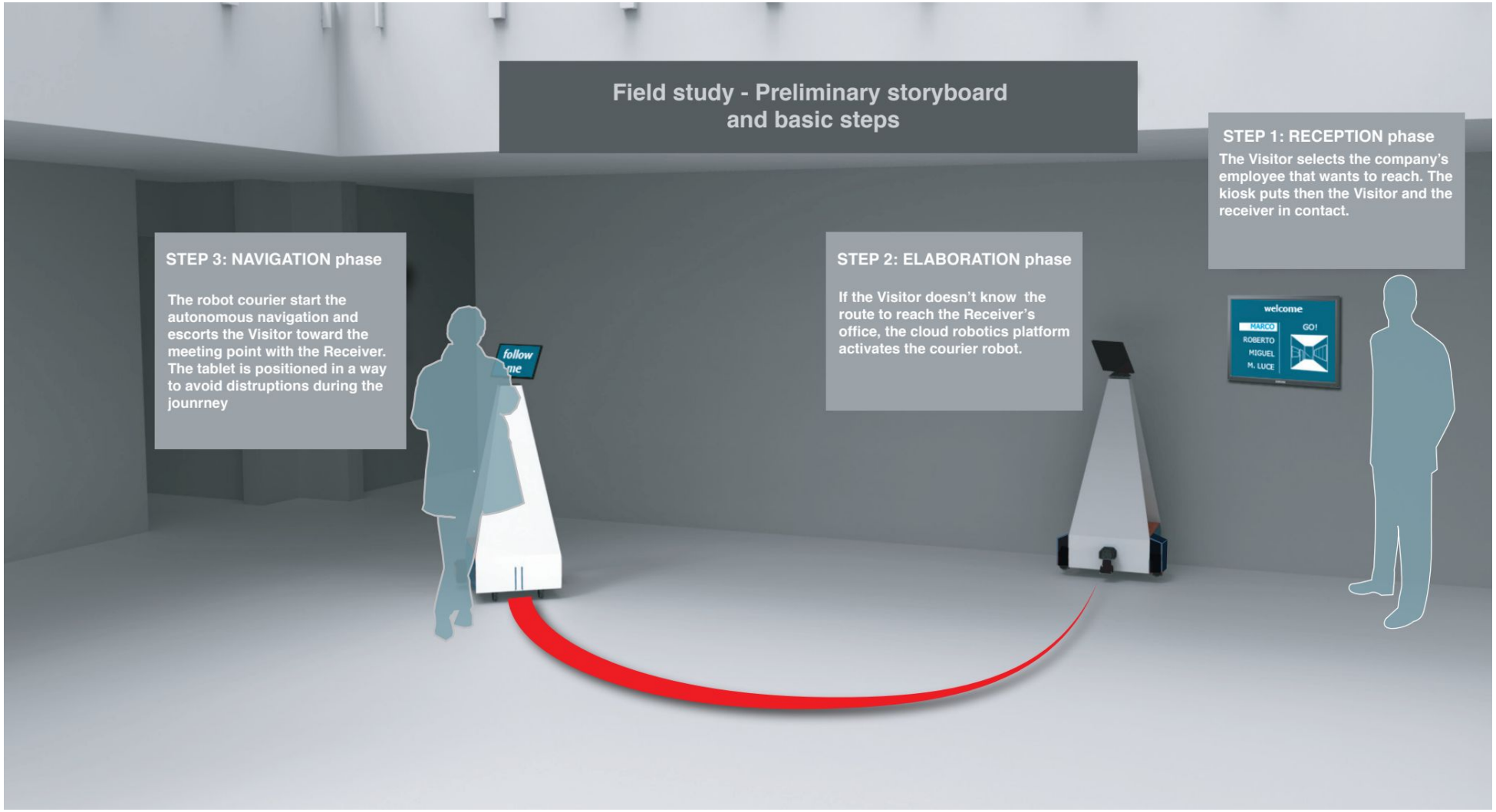
- Introduction
- Safe Navigation for UAS in urban areas
  - Cloud-based framework for UASs
  - Risk-based map
  - Risk-aware path planning
  - Simulation
- **Autonomous navigation for ground robots**
  - Cloud-based architecture for Service Robotics Applications
  - Dynamic path planning
  - Motion control with PF-MPEPC
  - **Service Robotics Applications**
- Conclusions

Service Robotics Applications

- Two *Service Robotics Applications* have been developed:
    - ***Robot Courier***, a service robot in a workspace
    - ***Virgil***, a robot in a museum
- Both services rely on the Cloud-based architecture for Service Robotics Applications
- They are implement in the Robot Operating System (ROS) framework
- Both Service use the Cloud Robotics Platform (CRP) developed by TIM S.p.A.

Service Robotics Applications: Robot Courier

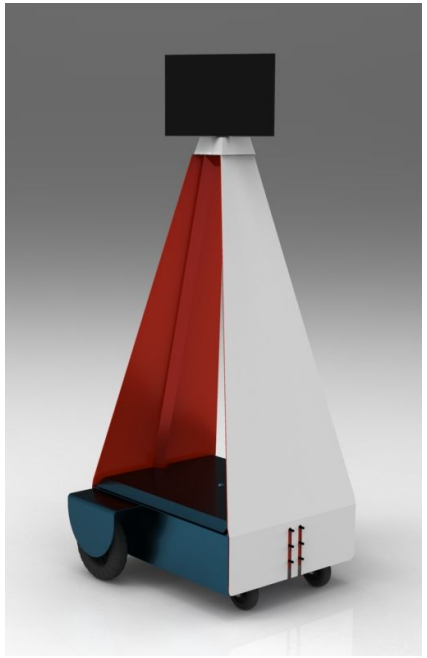
- The ***robot Courier*** aims to welcome and provide assistance to visitors in a workspace

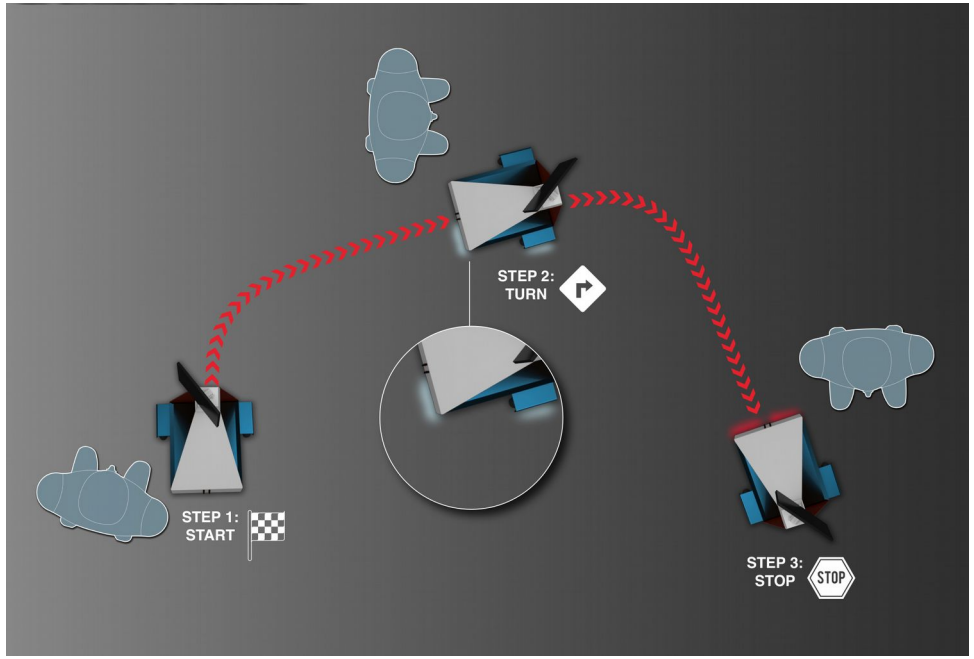


Service Robotics Applications: Robot Courier

- Fully autonomous navigation
- Obstacle avoidance with the PF-MPEPC approach
- Use of the Cloud Robotics Platform (CRP)
- 4G connection
- Auto Docking system

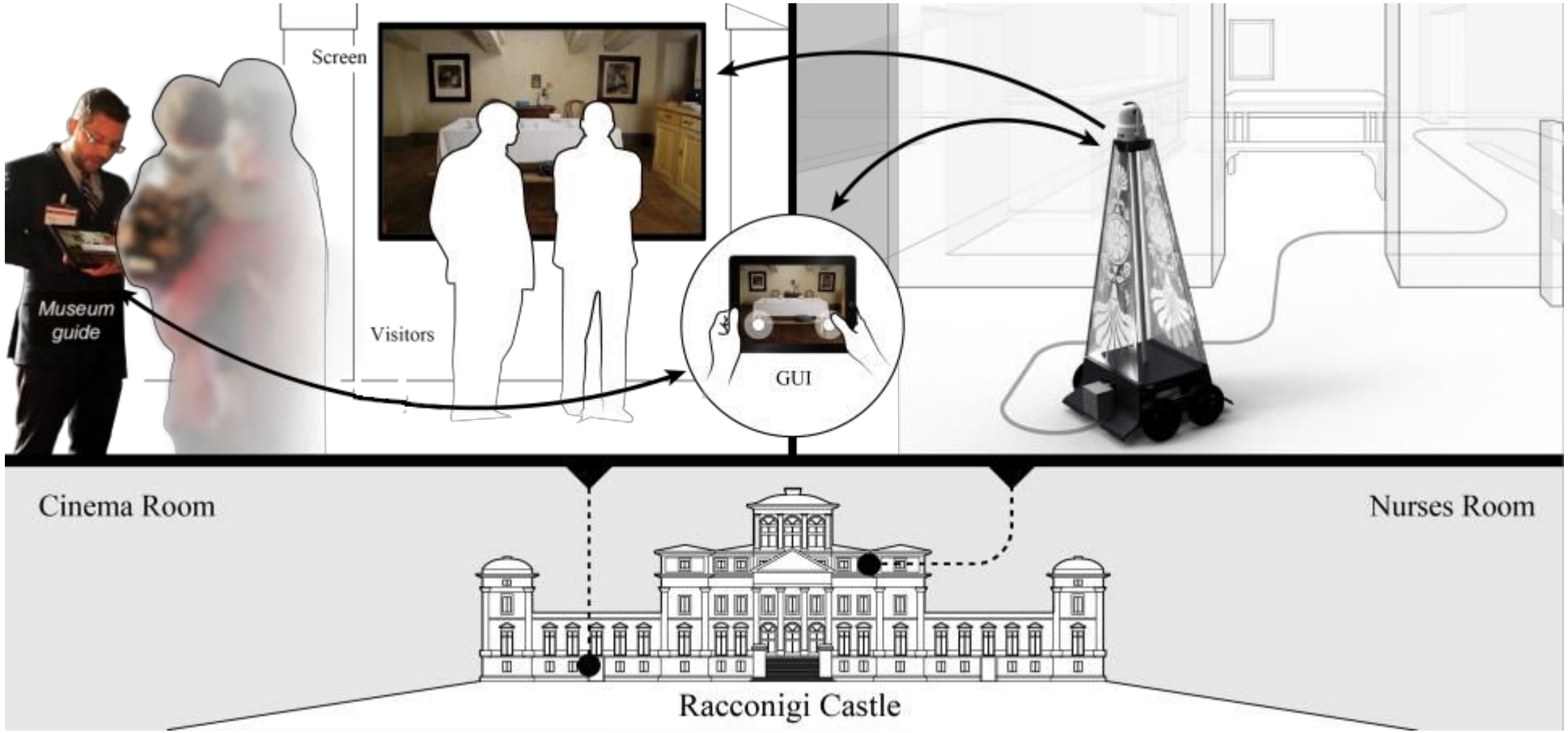
- User interaction with 2 tablets and with light signals



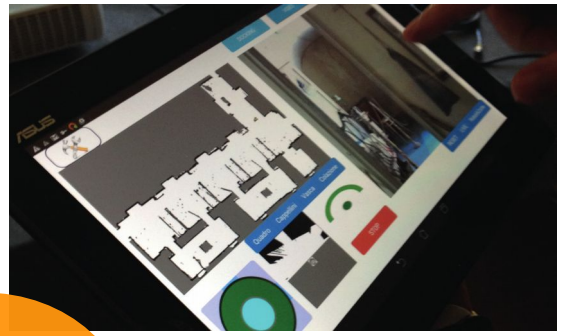


Service Robotics Applications: Virgil

- ***Virgil*** is a telepresence robot designed for improving the museum experience
- Virgil is used by the museum guide to extend the museum tour through a real-time virtual tour



Service Robotics Applications: Virgil



Experimentation at Castle of Racconigi

- Fully autonomous navigation
- Dynamic path planning
- Obstacle Avoidance
- Pan/Tilt Camera
- Use of the Cloud Robotics Platform (CRP)
- 4G connection

- Controllable with a Web User Interface (WUI)

Outline

- Introduction
- Safe Navigation for UAS in urban areas
  - Cloud-based framework for UASs
  - Risk-based map
  - Risk-aware path planning
  - Simulation
- Autonomous navigation for ground robots
  - Cloud-based architecture for Service Robotics Applications
  - Dynamic path planning
  - Motion control with PF-MPEPC
  - Service Robotics Applications
- **Conclusions**

Conclusions

- In this Ph.D dissertation, two different scenarios of autonomous navigation of mobile robots in crowded environments are studied
    - Safe navigation of Unmanned Aerial Systems in urban areas
    - Autonomous navigation of mobile robots for Service Robotics Applications

Conclusions (2)

- We proposed a solution to design and execute safe flight operations in urban environments
- A ground risk-based map assesses the risk to the population on the ground when the UAS flies over inhabited areas
- The risk-based map is a tool for risk-informed decision-making
- The risk-based map is used to plan a safe flight mission with a risk-aware path planning strategy
- The proposed risk-aware path planning consists of two phases: offline and online path planning
- Two different solutions are proposed to perform the risk-aware path planning:
    - Using riskA* and Borderland algorithms
    - Using the riskRRT^{X} algorithm
- Both solutions compute and update a safe flight mission
- Simulation results corroborate the proposed approach, which is able to plan a safe flight mission in a urban area

Conclusions (3)

- A Cloud-based framework for Service Robotics Applications is presented
- A dynamic path planner is proposed, able to compute and maintain a valid path, even in high dynamic environments
- A novel motion controller called Particle Filter Model Predictive Equilibrium Point Control (PF-MPEPC) is proposed.
  This method searches for the optimal equilibrium point that implies an optimal motion.
  Particle Filters consider disturbances and uncertainties in the prediction phase
- The proposed Cloud-based framework is used to implement two Service Robotics Applications
- The robot Courier welcomes visitors in a workplace and escorts them to a desired office
- Virgil provides a real-time virtual tour of inaccessible area of a museum
- Both Service Robotics Applications are tested in real environments


THANKS!