Chapter 8

Conclusion and Future Work

This section summarizes *obstacle avoidance framework* functionality (sec. 8.1), provides the*comparison to other methods* (sec. 8.2), outlines *approach reusability* (sec. 8.3), summarizes *lessons learned* (sec. 8.4) and introduces possible *future research heading* (sec. 8.5).

8.1 Summary

The approach presented in (ch. **??**) is covering the challenges (fig. **??**) related to:

1. *Reactive avoidance* - covering static obstacles, moving intruders, geo-fencing, weather effects.

2. *Event-based avoidance* - covers core UTM functionality and cooperative avoidance.

3. *Pre-emptive avoidance* - not covered in this work, the assumption about initial waypoint reachability still holds (ass. **??**).

Reactive Avoidance Test Coverage: Testing avoidance capabilities against *static obstacles*, *non-cooperative intruders*, *moving hard constraints* are covered. Coverage of the *soft constraints*, *map obstacles*, and *detected obstacles* are implicitly covered due to the properties of *safety* and *body* margins (tab. **??**).

1. *Building Avoidance* (sec. **??**) covers *static obstacles* explicitly and *map obstacles, hard constraints, ground avoidance* implicitly.

2. *Slalom* (sec. **??**) covers *open space navigation capabilities*, showing the determinism of the *avoidance loop run*, in addition to *building avoidance*.

3. *Maze* (sec. **??**) covers *closed space navigation capabilities*, showing the higher level navigation properties of primitive *right-side* 2D maze solver. The main point is to show possibility to enrich the *Navigation loop algorithm* (fig. **??**).

4. *Storm* (sec. **??**) covers *hard moving constraints avoidance* explicitly and *hard static constraints*, *soft static constraints*, *soft moving constraints* implicitly.

5. *Emergency converging* scenario (sec. **??**) covers *non-cooperative intruder with the right of the way* avoidance capability.

6. *Emergency head-on* scenario covers (sec. **??**) *non-cooperative intruder without right of the way* avoidance capability

7. *Emergency mixed* scenario (sec. **??**) covers *multiple intruders with/without right of the way* avoidance capability.

Event-Based Avoidance Coverage: Test cases covers *well clear breach prevention*, *situation-based avoidance*, and *rules of the air enforcement*. Coverage of *near miss situations*, *clash incidents* is given implicitly by *safety* and *body* margins (tab. **??**).

1. *Rule-based converging* (sec. **??**) covers *well clear breach* and *the converging rule of the air*, showing determinism and *UTM resolution execution*.

2. *Rule-based head-on* (sec. **??**) covers *well clear breach* and *head on rule of the air*, showing determinism and *UTM resolution execution*.

3. *Rule-based mixed head on with converging* (sec. **??**) covers *well clear breach* and *head on and converging rules of the air*. The main focus is on the *virtual roundabout* concept when multiple collision cases are clustered into one avoidance maneuver.

4. *Rule-based overtake* (sec. **??**) covers *well clear breach* during *overtaking* by faster UAS.

8.1.1 Hierarchical Decision Making

The key feature of the approach is hierarchical decision making. The hierarchy is introduced not only to threat prioritization but also to the time sequence of the decisions. The framework scheme (fig. **??**) shows that situational assessment in the form of *data fusion* is done in (sec. **??**). The trajectory selection process is separated from the final avoidance strategy pick-up process. The decision making separation is done like follows:

→ *Avoidance Run* (sec. **??**) - for one specific data set, for a fixed time, for a fixed goal find the cheapest trajectory which is safe (reachable).

→ *Navigation Run* (sec. **??**) - for multiple data sets (threat hierarchy), for a fixed time, for context depending goal try to find existing solution as a safe trajectory. Select trajectory breaking least restrictions from candidates.

The decision making process on threat assessment level (fig. **??**) is the configuration of threat type application in 7^{th}-10^{th} step. The idea is that UAS:

1. must avoid any *static obstacles* (sec. **??**),
2. can skip *intruders* (sec. **??**) if there are no other options
3. can enter into *hard constraints* (sec. **??**, def. **??**) area if its necessary,
4. can enter into *soft constraints* (sec. **??**, def. **??**) area if its necessary.

Note. The *implementation* of *soft/hard constraints* and their evaluation process is same in *avoidance run*; the only difference is when they are applied by *navigation run*.

8.1.2 Use of Developed Reach Set Approximations

The reach set approximation (sec. **??**) represents a set of *maneuvering* strategies (avoidance/movement) which can be used in a different context. The properties of the trajectories are given based on *constrained expansion* (sec. **??**) and performance parameters priority (sec. **??**).

The conditions and priorities are changing with the operational environment, to successful integration into nonsegregated airspace is necessary to reflect this. For this purpose, the multiple reach set approximation approaches were developed. The performance tests and longer summary can be found in (sec. **??**). The summary of each reach set approximation method is given like follow:

→ *Coverage-maximizing reach set approximation* (sec. **??**) - contains a lot of *curved trajectories* with high space coverage ratio, used for *emergency reactive avoidance* in non-controlled/controlled airspace, *last resort reach set*.

→ *Turn-minimizing reach set approximation* (sec. **??**) - contains a lot of *straight* trajectories with low space coverage ratio, used for navigation in *non-controller airspace*.

→ *Combined reach set approximation* (sec. **??**) - the reach set approximation is represented as a tree, there is a possibility to combine two or more reach sets with the same initial state, the outcome of such merge is a new reach state with different properties.

→ *ACAS-like reach set approximation* (sec. **??**) - contains *mandatory separations* (depending on aircraft type), has low trajectory count and low coverage ratio, it emulates ACAS maneuver table in reach set data structure, used for navigation and avoidance in ATC controlled airspace.

8.2 Comparison to Other Methods

The *testing* (sec. **??**) is strongly *capability* oriented, some performance testing regrading computational load (sec. **??**) is on the place. The advantages of our approach regarding scalability are outlined in (sec. 8.2.2).

This section summarizes the results achieved by one of the master students [1]. Who compares other avoidance methods concerning performance. The *conservative* vector field avoidance [2] and *adaptive* potential field [3] methods are compared to our approach (sec. 8.2.1).

8.2.1 Conservative and Adaptive Method Comparison

Testing scenario: The testing scenario is similar to *building avoidance* (sec. **??**), using the default framework testing configuration (sec. **??**). The *reach set approximation* used is a *combination of TM-RSA and CM-RSA* (fig. **??**) filling the role for *navigation/avoidance* in non controlled airspace. The *mission* is given by waypoints defined in (tab. 8.1), the UAS is starting at $\mathcal{WP}_1$.

Position		Waypoints				
$[x, y, z]$	$[\theta, \varpi, \psi]$	$\mathcal{WP}_1$	$\mathcal{WP}_2$	$\mathcal{WP}_3$	$\mathcal{WP}_4$	$\mathcal{WP}_5$
$[0, 0, 0]^T$	$[0°, 0°, 0°]^T$	$[0, 0, 0]^T$	$[20, 0, 0]^T$	$[20, 20, 0]^T$	$[0, 20, 0]^T$	$[0, 0, 10]^T$

Table 8.1: Mission setup for *Performance test* scenario.

The *static obstacle set* (tab. 8.2) contains three obstacles in the middle of waypoints. The obstacles are balls with uniform radius. The concept of *body* and *safety margin* was not used in this test.

Obstacle			Radius
id	position	type	
1	$[10, 0, 0]^T$	ball	2
2	$[20, 10, 0]^T$	ball	2
3	$[10, 20, 0]^T$	ball	2

Table 8.2: *Obstacle set* for *Building avoidance* scenario.

Note. The z coordinate value of zero does not represent the ground in this test scenario, because some methods do not support ground avoidance like ours.

Scenario example: The *scenario* was run for the following approaches:

1. *Static obstacle avoidance based on reach sets* - implementation [1] of [4], using own framework.
2. *Conservative* - vector field avoidance [2], adapted implementation [1].
3. *Adaptive* - potential field [3], adapted implementation [1].

The example of the testing framework developed by my student is shown in (fig. 8.1). The *obstacles* are big red balls consisting from smaller red balls, representing LiDAR sensor

hit zones. The *UAS* sensing range is represented as an area outlined with a black dashed line. The waypoints are marked as green squares. The flew trajectory is represented as blue line. The *planned trajectories* (red line) for decision frames (magenta circle) which are not executed are also shown.

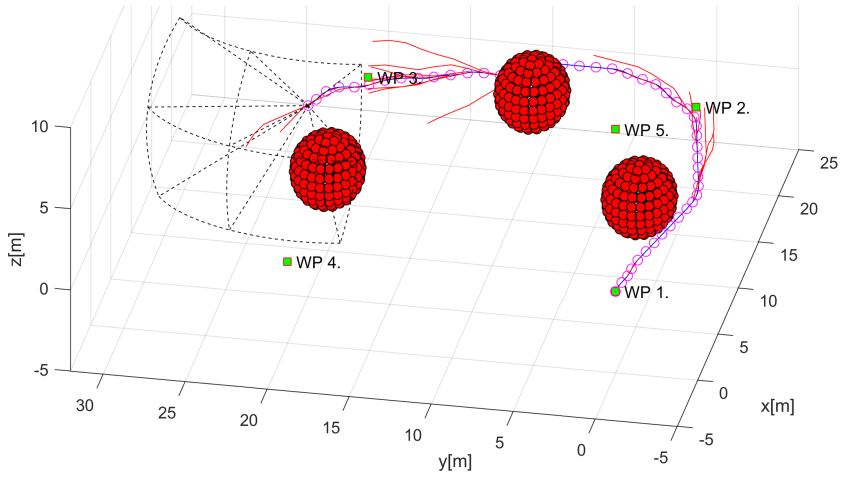

Figure 8.1: A testing scenario for method performance comparison [1].

Conservative Method Performance Margin: The *principle* of the conservative method is like follows: *Every time there must be a place to turn without the context.* The conservative method performance margin was estimated by [1] like follow:

$$conservative_{margin} = 2 \times turningRadius + 3 \times uasRadius$$

Using parameters from our testing configuration (sec. **??**) where the turning radius is 2 and UAS body radius is 0.6 the theoretical $conservative_{margin}$ is 4.6.

Conservative Method Performance Margin: The *principle* of the conservative method is like follows: *Every mass point has a potential equal to its expected mass, those potentials repel each other with proportional force.* The adaptive method performance margin was estimated by [1] like follow:

$$adaptive_{margin} = 3 \times uasRadius$$

Using parameters from our testing configuration (sec. **??**) where the UAS body radius is 0.6 the theoretical $adaptive_{margin}$ is 1.8.

Note. The *real conservative and method performance* was close to *theoretical best performance* [1].

Distance to Body Margin Evolution: The performance of an *obstacle avoidance framework based on reach set* for the mission (tab. 8.1) is shown in (fig. 8.2). The critical margin (crash zone) is 0.6 (UAS body radius) and is denoted as a red dashed line.

The *adaptive margin* id 1.8 and its denoted as a yellow dashed line. The *conservative margin* is 4.6 and its denoted as green dashed line.

The distance to the *body margin* of the UAS center stayed mostly in *adaptive zone* (yellow line) and two times entered into *outperforming zone* (red line). The *outperforming zone* is distance values between:

$$criticalMargin(0.6) \geq outperformingZone < adaptiveMargin(1.8)$$

One can say that reach set method outperforms selected conservative and adaptive methods representatives.

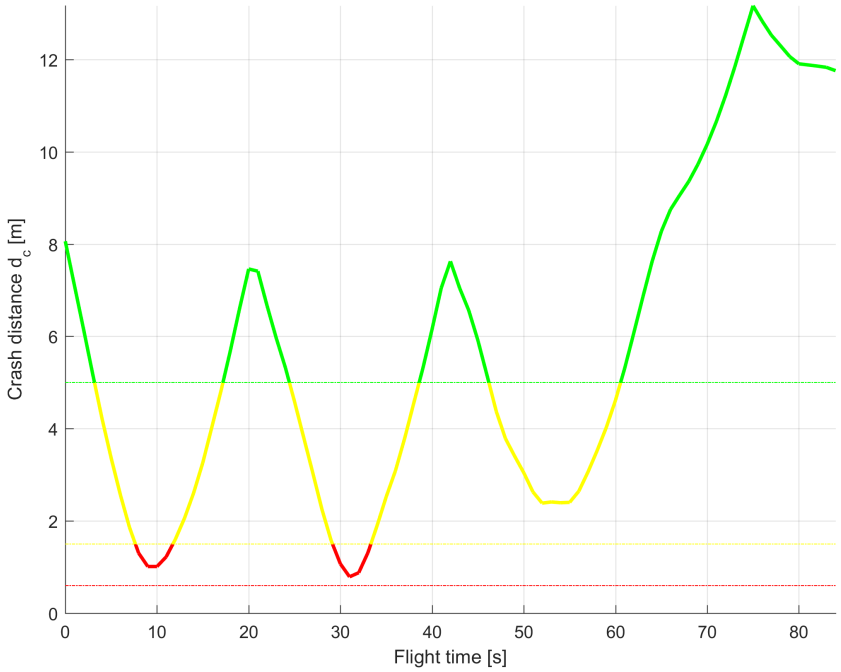

Figure 8.2: Distance to body margin evolution [1].

Disclaimer: Further investigation regarding *cost-effectiveness* is required, the approach needs to be tested in real environment and on the multiple scenarios. The testing concerning approach feasibility have been proven (tab. **??**).

8.2.2 Scalability

The *many methods* do not offer *scalability*. The scalability means that method applies to multiple problems with different scale or there is a sufficient amount of tune-able parameter to make it work in different environments. Many avoidance concepts have hardwired margins.

The *Detect and Avoid* system must offer scalability to some extent, because of changing regulations and performance criteria on the UAS systems.

Range scalability: The physical constraints need to be sensed by sensor array in avoidance grid, and the UAS cannot get to close to them.

Static obstacles: The body margin or the minimal distance of UAS center to obstacle body is constrained like follow:

$$uasRadius < bodyMargin \leq gridRange$$

The *tuning parameter* for static obstacles is *safety margin* is then constrained like follow:

$$bodyMargin < safetyMargin \leq \infty$$

The constraints for safety margin calculation and some guidelines were outlined in (app. **??**).

Intruders: The *intruders* (sec. **??**) are limited by *detection range* and intersection models parameters ranges.

The *body volume intersection* (app. **??**) has a following constraint:

$$intruderRadius < bodyRadius \leq minimalDetectionRange - 2 \times ourUASradius$$

Note. The *body radius* does not need to fit exactly on the intruder real body radius; usually, some padding space is added.

The *maneuverability uncertainty intersection* (app. **??**) is tuned through the uncertainty spread on the horizontal and vertical plane. This spread cannot be infinite, and the only limitation is given by implementation:

$$\begin{aligned} 0^\circ \leq \ & horizontalSpread < \ 90^\circ \\ 0^\circ \leq \ & verticalSpread < \ 90^\circ \end{aligned}$$

Constraints: The constraints are considered as *airspace restrictions*, *weather zones*, *geo-fencing zones* in this work. The constraints have two implementations *Static constraints* (sec. **??**) and *Moving constraints* (def. **??**). The only real constraint is the detection range; the constraint must be detected before making any impact on *grid*:

$$impact(uasPosition, constraintCenter) < detectionRange - gridRange$$

8.3 Approach Reusability

The *avoidance framework* can be used as a whole, or some of the concepts can be transferred as modules to other approaches. This section picks such reusable concepts,

UTM Services: The constrained *UTM functionality* is outlined in (sec. **??**) including:

1. *Future UTM Communication Architecture* (fig. **??**) as the authority over *airspace segment* (fig. **??**)[5].
2. *Cooperative Conflict Resolution Under UTM Supervision* (fig. **??**) designed as mild/feasible directives (commands) with *constant supervision*.
3. *Rules of the Air Enforcement* (sec. **??**, **??**, **??**) including designs of *Position Notification* (sec. **??**) and *Collision Case Structure/Calculation* (sec. **??**).
4. *Divergence/Convergence Waypoints* concept is showcased in *Overtake Rule* (rule **??**).
5. *Weather Avoidance* (app. **??**) is using a similar concept to *Collision Case*: *Weather Case*. The information is provided by *Local Airspace Authority*.

Emergency Avoidance Functionality: The standard framework implementation (fig. **??**) can handle the situations given in non-cooperative test cases (sec. **??**). The list of threats is given by (tab. **??**).

Event-Based Avoidance Functionality: The standard framework implementation (fig. **??**) with active $C2$ link and rules setup (fig. **??**) can handle the situations given in cooperative test cases (**??**). The list of threats is given by (tab. **??**). The *Avoidance Mode Concept* enables to switch between *Event-Based Avoidance* (Navigation) and *Emergency Avoidance*.

Note. The emergency Avoidance Functionality is included in *Event-Based Avoidance* (Navigation) mode. The prioritization of *threats* may differ (tab. **??**).

Reusability for More Complex Systems: The framework (fig. **??**) with implemented rule engine (fig. **??**) can be used on *any system*, with appropriate *Movement automaton* (sec. **??**) enabling *wave-front* propagation (alg. **??**) for reach set estimation. Following artifacts needs to be delivered for concept reuse:

1. The *Movement Automaton* is used to generate *thick series of waypoints* which guarantees desired degree of safety.
2. The *complex UAS system* is following the *reference trajectory* (sec. **??**).
3. The *Sensor Fusion* (sec. **??**) implementation including classification to *Free*, *Occupied*, *Restricted* space type.
4. The *sensor field* is supporting detection of threats. There should be at least one sensor with the capability of feeding *Avoidance Grid*. Our implementation was based on LiDAR/ADS-B feeds.

5. The *Information Sources* are supporting the online/offline threat processing. This one is completely optional.

Note. On UTM integration: The future UTM system will not be giving the extreme commands, the directives are more like constraints; therefore our system can provide the guidance and constraint evaluation

Note. On Safety Margin: The disparity between real flown trajectory (nonlinear dynamics) and planned trajectory (Movement Automaton) needs to be accounted into *Safety Margin*.

Reach Set Approximations: The *wave-front* approach (alg. **??**) can be used with *Constrained expansion function* (sec. **??**) to create own *Reach set Approximation Method*. Existing reach set approximation methods are always following a different goal; they can be reused for other tasks (perf. **??**):

1. *Coverage-maximizing* (sec. **??**) - high space coverage, ideal for unpredictable and complex avoidance maneuvers.

2. *Turn-minimizing* (sec. **??**) - smooth trajectories, medium space coverage, ideal for navigation maneuvers.

3. *Combined* (sec. **??**) - a combination of the *CM-RSA* and *TM-RSA*, the cost function defines preferred trajectories. The procedure is reusable for any reach set approximation types (2^+) combination.

4. *ACAS-X Like* (def. **??**) - following *TCAS/ACAS separation modes*, can be used as an alternative for *controlled avoidance* and *navigation*.

8.4 Lessons Learned

During the approach development, some mistakes were made. This section summarizes the most notable design choices with the reasoning behind them.

Euclidean vs. Planar Space Partitioning: The *operational space discretization* was planned from the beginning. The initial approach was to use a *uniform Euclidean grid* where the smallest space portion was represented as cubic cell.

Euclidean grid shortcomings: The problem was with *LiDAR* reading assessment because for each point it was necessary to do *transformation* defined as a series of the functions (eq. **??**, **??**, **??**, **??**). The continuous updating of the *space assessment* usually timed-out.

The other issue was with wavefront algorithm (alg. **??**). The intersection (def. **??**) function was not problematic, due the unlimited estimation time of *reach set approximation*. The *reach set approximation* is shaped usually like the cone; therefore each propagation caused hitting more cells in Euclidean grid. This caused the following challenges:

1. *An exponential growth of trajectory count* - the *trajectory footprint* (def. **??**) is almost always unique in Euclidean Grid. The problem is that two very similar trajectories concerning avoidance were considered unique.

2. *Low avoidance strategies variability* - the generated trajectories by *TM-RSA* (sec. **??**) or *CM-RSA* (sec. **??**) reach set approximation did not follow the desired properties of coverage.

3. *Threat impact reflection* - the space which is closer to UAS needs to be surveillance with greater detail than space which is further away from UAS. The Euclidean grid does not reflect this.

Euclidean Grid Benefits: The intersection model performance was good because of most of the numeric/analytically intersection solutions for:

1. *Line* - used in *linear intruder intersection model* (app. **??**).

2. *Tunnel* - used in *body-volume intersection model* (app. **??**).

3. *Polygon* - used in constraints (sec. **??**, def. **??**)

Planar Grid Deployment: The shortcomings and benefits of *Euclidean grid* are obvious, the main reason why the *planar gird* was employed is the *performance of the wave-front algorithm* (alg. **??**) for UAS system dynamic. The main focus of this work lies on *Reach Set* properties, the planar grid (fig. **??**) properties:

$$\uparrow distance \implies \uparrow cellSize \implies \downarrow importance$$

are ideal for *avoidance problems*. The cell size is increasing proportionally with increasing distance. The space partitioning at the lower distance is dense, reflecting the importance of proximity threats. The space partitioning at the greater distance is getting looser, reflecting the unimportant of threats at greater distance.
The *planar grid* employment has following benefits:

1. Fast *LiDAR* (and other ranging sensors) clustering.

2. Reflection of space importance.

3. A significant increase of *wave-front* algorithm.

4. A significant decrease in reach set approximation node count.

The *planar grid* employment has the following shortcomings:

1. Implementation complexity of intruder intersection models.

2. Implementation complexity of constraints intersection models.

Note. The count of intruder and constraints is not so great concerning *aerial transpiration*; the situation is opposite in case of *ground transportation* especially road vehicles.

Intruder Intersection Models: The *implemented intruder intersection models* (app. **??**) are basic and sufficient. There are few ideas which do not make it into the final concept:

1. *Intruder as line/curve intersection* - the idea is to use a parametric curve instead of the *linear intersection model* (app. **??**). The problem is those curve parameters are hard to obtain; they are not mandatory part of any position notification proposal so far.

2. *Intruder as movement automaton simulated trajectory* - the idea is to have access to movement automaton of an intruder in *predictor mode* (sec. **??**) to generate future expected trajectory. The problem is in some situations the trajectory of intruder changes a lot and *maneuverability uncertainty intersection* (app. **??**) outperforms this approach significantly. The other problem is that most of the *general aviation* does not want to share this kind of information due to security reasons. This concept makes it into the final solution, but not in *avoidance framework*, but as *UTM collision case intersection calculation* (sec. **??**).

3. *Intruder well clear implementation as a rule* - the idea is to implement *potential fields principle* as core mechanism of avoidance. The problem is determinism of this approach, and the idea was scrapped.

Approach to data fusion: To make our approach universal, the decision to use input/output interfaces were made (fig. **??**). The choice of the *output control interface* is clear, the movement automaton (sec. **??**) was performing well from the beginning [6, 7].

The *input interface* represented as *data fusion* (theory sec. **??**, implementation sec **??**), was more problematic.

Deterministic approach: The article [4] and technical reports [6, 7] are using a deterministic approach where the property of the avoidance grid cell is given as a Boolean value. This approach is too restrictive because all the values needed to be determined on the spot. The overall data fusion was nonexistent because the approaches were used only static obstacles and *LiDAR* sensor as the base.

Probabilistic approach: The technical report [8] introduced the *intruders* and *map obstacles* [9]. This requires*probabilistic implementation*. The probabilistic density functions with all that theoretical apparatus are established in order to cover the processes of the data fusion and decision making. The problem was the formalization level; much *effort* is wasted to formalize very simple calculations and thresholds. The tuning parameters were nonexistent, preventing the customization.

Final Approach as a Synthesis: The benefits of both approaches for data fusion are used in final implementation (sec. **??**), the useful aspects of probabilistic approach like ratings and addition rules persisted. The useful aspects of deterministic, like threshold, is kept through predicates (tab. **??**). The concept is the following:

1. *Ratings reflects probabilities* - the property of visibility (eq. **??**), obstacle encounter in cell (eq. **??**), map obstacle in cell (eq. **??**), intruder intersection with cell (eq. **??**), constraint intersection of the cell (eq. **??**) are formalized as threat property (def. **??**).

2. *Man-made rules incorporation* - man-made rules can be incorporated into any part of the data fusion process.

3. *Situation assessment* - the known world classification through data fusion (eq. **??**) is implemented as cell set classification:

   a. *Uncertain* (eq. **??**) - the state of this space cannot be determined by sensors or prior knowledge.

   b. *Occupied* (eq. **??**) - these cells are occupied by a physical object, entering into this space will cause real damage to UAS.

   c. *Constrained* (eq. **??**) - intruder or other kinds of threat may be present, this space is better to avoid.

   d. *Free* (eq. **??**) - these cells are free of any threat, this fact is verified by sensors and prior knowledge.

   e. *Reachable* (eq. **??**) - these cells are free and moreover there exists at least one *safe trajectory* leading there, the safe trajectory is a trajectory from *reach set approximation* which goes through free space.

Note. The formal definition for data fusion (sec. **??**), known world (sec. **??**) and their role in problem solution (sec. **??**) are impacting the final data fusion implementation.

8.5 Future Work

The future work needs to address issues of *adversarial avoidance* (sec. **??**) and *practical implementation* (sec. **??**).

Adversarial Avoidance: The counterexample (sec. **??**) showed that the approach is vulnerable to *adversarial behavior*. The adversarial UAS just simply avoided our field of the vision to proceed with side-hit.

This opens the possibility to solve the problem as *differential game* [10, 11]. Our UAS should pose as the *defender*; adversary should pose as the *attacker*. The *reach set* of the system given as:

$$differentialGame = model(positions, defender, attacker)$$

This reach set will give us the options in our decision-making process to avoid the pursuer. This adversarial will bring additional scientific challenges which can yield interesting results.

Real System Implementation: The testing proved the *capability of approach* for wide range of applications (tab. **??**). To proceed further with comparative testing, beyond theoretical implementations (sec. 8.2.1), it is necessary to deploy it in real environment.

The *ideal candidate* is *LSTS-tooolchain* [12]. The LSTS offers all necessary base for *approach software architecture* (fig. **??**). The parts of our approach can be distributed over LSTS-toolchain in the following manner:

1. *LSTS Dune* (UAS on-board control) - the implementation of main *navigation loop* (sec. **??**), including sensor integration with data fusion (sec. **??**). The *rule engine* (sec. **??**) can be deployed after UTM services implementation, to support *cooperative maneuvers*.

2. *LSTS Neptus* (UTM equivalent) - the implementation of *UTM* services and calculations (sec. **??**).

3. *LSTS IMC* (Messaging implementation) - the messaging support for cooperative (fig. **??**) and non-cooperative (fig. **??**) communication schemes.

4. *LSTS Ripples* (Long term data storage) - the flight-log storage.

The real system implementation will enable to:

1. *Compare approach performance* - the theoretical performance of approach is good, only the real challenges can show the flaws and strengths of approach.

2. *Develop fault-tolerant and recovery procedures* - the process of *event-based* and *reactive* avoidance have been developed and tested in theoretical environment. The real implementation can improve the process weak points by *"learning by doing"* method

3. *Advertise the approach benefits* - the successful implementation on a real system will increase the outreach and visibility of the approach significantly.

Bibliography

[1] Martin Hrdlik. Advanced obstacle detection and navigation methods of unmanned vehicles. Master thesis, Institute of Automotive Mechatronics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Ilkovicova 3, Bratislava. Slovak Republic, jun 2018.

[2] Johann Borenstein and Yoram Koren. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, 1991.

[3] Yoram Koren and Johann Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 1398–1404. IEEE, 1991.

[4] Alojz Gomola, João Borges de Sousa, Fernando Lobo Pereira, and Pavel Klang. Obstacle avoidance framework based on reach sets. In *Iberian Robotics conference*, pages 768–779. Springer, 2017.

[5] Ingrid Gerdes, Annette Temme, and Michael Schultz. Dynamic airspace sectorization using controller task load. *Sixth SESAR Innovation Days*, 2016.

[6] Alojz Gomola. Optimal control of unmanned air vehicles with obstacle avoidance capabilities in partially known environment. Technical report, FEUP, 2017.

[7] Alojz Gomola. Model predictive control of unmanned air vehicles with obstacle avoidance capabilities. Technical report, FEUP, 2017.

[8] Alojz Gomola, Pavel Klang, and Jan Ludvik. Probabilistic approach in data fusion for obstacle avoidance framework based on reach sets. In *Internal publication collection*, pages 1–93. Honeywell, 2017.

[9] Maria Cerna. Usage of maps obtained by lidar in uav navigation. Master thesis, Institute of Automotive Mechatronics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Ilkovicova 3, Bratislava. Slovak Republic, jun 2018.

[10] Nikolai Nikolaevich Krasovskij, Andrei Izmailovich Subbotin, and Samuel Kotz. *Game-theoretical control problems*. Springer-Verlag New York, Inc., 1987.

[11] NN Krasovskii and AI Subbotin. Game-theoretical control problems. translated from the russian by samuel kotz, 1988.

[12] José Pinto, Paulo S Dias, Ricardo Martins, Joao Fortuna, Eduardo Marques, and Joao Sousa. The lsts toolchain for networked vehicle systems. In *OCEANS-Bergen, 2013 MTS/IEEE*, pages 1–9. IEEE, 2013.