6.8 UAS Traffic Management

The *Traffic Management* for UAS is based on existing Air Traffic Management System for manned aviation [1]. The controlled airspace segments are *static* and have one *authority for one zone* principle. The dynamic zones have been proposed in [2]. However, it will be omitted for *simplification purpose*. The necessity for *UAS integration* into *National Airspace* has been outlined in [3].

The latest *Airbus blueprint* [4] outlines some functionality. The main purpose of this section is to show *Reach Set based Approach* capability to follow *Usual Air Traffic Management* commands.

The *section* is organized to introduce:

1. *UTM Architecture* (sec. 6.8.1) - centralized ATM-like authority over airspace cluster.

2. *Cooperative Conflict Resolution* (sec. 6.8.2) - the model used for conflict resolution in *controlled airspace*.

3. *Non-Cooperative Conflict Resolution* (sec. 6.8.3) - the model used for conflict resolution in *non-controlled* airspace and *emergency avoidance*.

4. *Handling Standard Collision Situations* - head-on approach (sec. 6.8.4), converging situation (sec. 6.8.5), overtake (sec. 6.8.6).

5. *Position Notification* (sec. 6.8.7) - position notification design.

6. *Collision Case* (sec. 6.8.8) - calculation and handling of *collision situations*.

7. *Weather Case* (sec. 6.8.9) - definition and handling of *weather hazards*.

6.8.1 Architecture

UTM Concept is based on *asynchronous event-based control* [5]. *Event* in *controlled airspace* is handled in the form of *cases* [6]. There are following *event sources*:

1. *Weather Information Service* (from [7]) - used to create *weather case* (tab. 6.4).

2. *Position Notification from UAS systems* (tab. 6.1) - used to create *collision cases* (new functionality) (tab. 6.3).

Decision Frame (eq. 6.1). The *UTM* is operating in discrete decision frames which are starting on current *decision time* and ending at next *decision time*:

$$decisionFrame_i = [decisionTime_i, decisionTime_{i+1}[, \quad i \in 1, \ldots, k, k \in \mathbb{N}^+ \tag{6.1}$$

Event-based Airspace Control is collecting events in previous $decisionFrame_{i-1}$ and issuing commands in current $decisionFrame_i$.There are following phases during the *UTM frame* cycle:

1. *Planning* - the detection phase, when the hazardous situations are assessed.

2. *Fulfillment* - the monitoring phase, controlled UAS systems fulfill the state of affairs for directives and mandates.

3. *Acknowledgment* - the closing phase, when UTM assess and acknowledges the performance of controlled UAS systems.

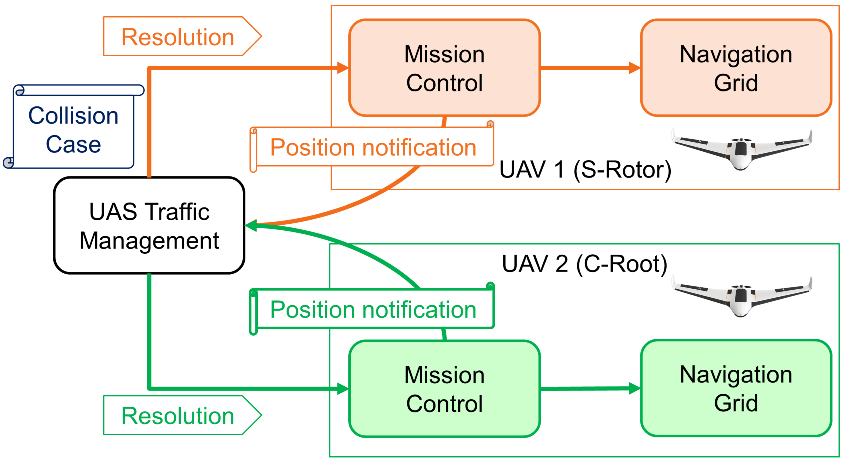


Figure 6.1: UAS Traffic Management (UTM) architecture overview.

Architecture (fig. 6.1). There are multiple UAS systems equipped with standard *Mission Control* and *Navigation* procedures.

Depending on the *airspace cluster* decision time frame they are sending *periodical position notifications* (tab. 6.1).

The *UAS Traffic Management* (UTM) collects the event data from *Weather Information Service* and *Position Notifications* calculating respective *cases*.

If there is an *active collision/weather case*, the *UTM* will send *resolutions* to respective airspace attendants.

6.8.2 Cooperative Conflict Resolution

Idea: There is a *final decision maker* (absolute authority) in conflict resolution. This authority is *UTM* or *air traffic attendant* with higher priority. The future *UTM system* is such authority. The approach to mixed conflict resolution is mentioned in [8], based on navigation [9]. This is similar to our approach.

Note. Open Issue: Decentralized model with UTM as an approver of directives is possible, but that is a topic for own research.

Goal: UAS is obligated to follow up committed mission plan with given precision. There is one to five percent allowed deviations for ATM mission plans. Similar rates are achievable according to [8]. This requirement is given by [1] ICAO 4444 document for ATM operations.

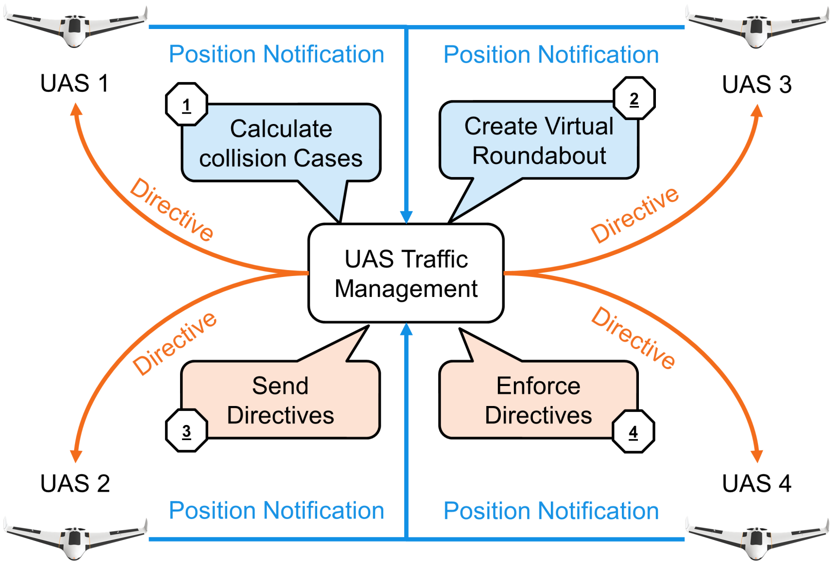


Figure 6.2: Cooperative conflict resolution via UTM authority.

Cooperative Conflict Resolution (fig. 6.2) shows a functional diagram of one *UTM time-frame* there are following actors:

1. *Unmanned Autonomous System* (UAS) equipped with necessary navigation and communication modules, providing the unique *identification number*.

2. *UAS Traffic Management* (UTM) posing as the central authority for given *airspace cluster*.

The following steps are executed during *Cooperative conflict resolution*:

1. $UAS_* \rightarrow UTM$ *Send position notification* - each *UAS* is notifying the authority (UTM)

2. $\circlearrowleft$ UTM *Calculate collision Cases* - UTM gathers data and predicts possible collisions then it tries to link them and manage the situation.

3. $\circlearrowleft$ UTM *Create virtual Roundabout* - active collision cases are aggregated into a virtual roundabout.

4. $UTM \rightarrow UAS_*$ *Send directives* - UTM sends commands to UAS systems which need to change their planned trajectories.

5. $UTM \rightarrow UAS_*$ *Enforce directives* - UTM is periodically checking constraints imposed in previous *decision frames*.

6.8.3 Non-Cooperative Conflict Resolution

Idea: There is *main UAS(1)* which is flying in open *non-controlled* airspace. Other UAS are operating in its vicinity. It is expected that they are claiming their *planned trajectories*. The *Main UAS(1)* detects the collision with other *UAS*(2-4).

There is no *final decision maker* nor *supervising authority*; all communication participants have a similar level of rights.

Note. There is an assumption that other airspace users are behaving like intruders, without intent to destroy or harm. The *adversarial behavior* is not accounted. The response from an *intruder* is not mandatory in *non-controlled* airspace.

Goal: Provide *mutual avoidance mechanism* in *non-controlled* airspace. Let us consider the equal standpoint of all airspace attendants.

Conflict Resolution: The conflict resolution depends on current mode and *handshake* between airspace attendants. The non-cooperative behavior has been implemented as follows:

1. *Navigation mode* - every *airspace attendant* is calculating own *collision cases* and checking the behavior of the other (virtual UTM).
2. *Emergency avoidance mode* - is depending on communication mode:
   a *Response mode* - claiming separation methods and using avoidance mechanism (Avoidance grid with intruder model in our case).
   b *Blind mode* - every conflict side picks own strategy respecting given *rules of the air*.

Note. Intruder Intersection model selection: UAS based on Event detects possible collision for some reason UTM directive is out of the question, then try to claim separation (body volume intruder model (sec. **??**)), If separation fails, go full survival mode (uncertain intruder model (sec. **??**)).

Special Cases in Manned Aviation: There are IFALPA reports which can give us an overview of *enforced non-cooperative* mode causes in *controlled airspace*:

1. *VFR disabled* - flying in fog or thick clouds can render pilot vision, similar to UAS cameras/LiDAR.

2. *IFR equipment broke* - the sensor malfunction is more likely to happen due to the lesser redundancy in UAS systems.

3. *C2C Link disabled* - communication loss is more likely to happen, due to the lesser redundancy.

4. *ATM failure* - the ground control module of UTM can also fail.

Note. Traffic management related fails are lesser than 0.001 cases per one flight (according to IFALPA [10]).

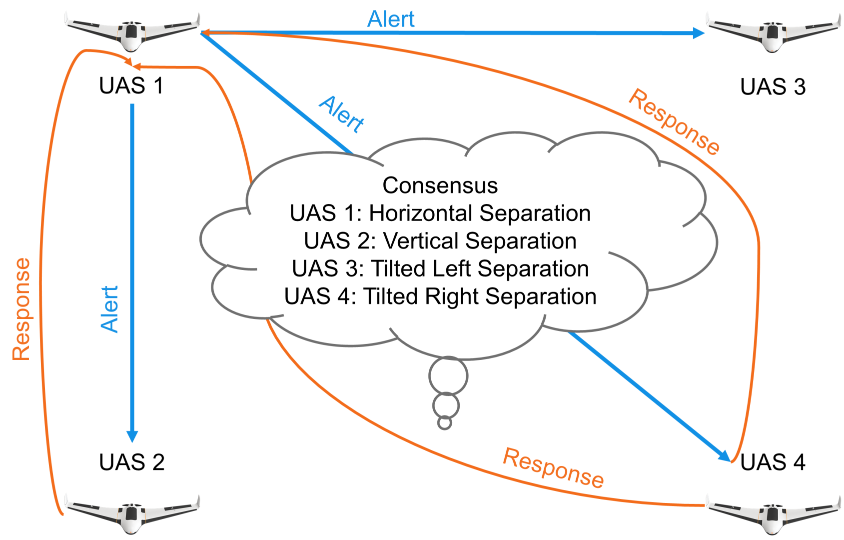


Figure 6.3: Non-cooperative conflict resolution via UAS claims.

Response mode scenario example: The *main UAS(1)* is going to collide with other *UAS*(2-4):

1. $UAS(1) \rightarrow UAS(2-4)$ sends position and heading notification.

2. $\circlearrowleft UAS(2-4)$ calculates possible collisions.

3. $UAS(2-4) \rightarrow UAS(1)$ sends a response to the *main UAS(1)* with claimed separation mode.

4. $\circlearrowleft UAS(1)$ acknowledges proposed *separation modes*.

5. $\circlearrowleft UAS(1-4)$ avoids each other using claimed separation mode because every *UAS* achieved *consensus*.

Note. The mutual consensus is not usually achieved via C2 communication. The most common case is *assuming separation mode*. This case is shown in (sec. **??**)

6.8.4 Handling Head-on Approach

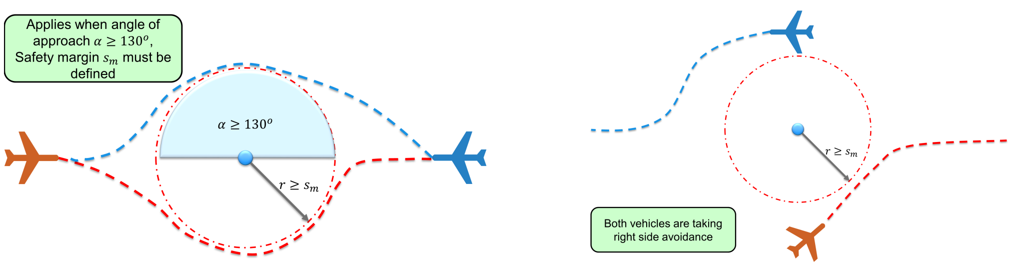
Goal: Identify required parameters sufficient for automatic solution of *Head-on collision* situation.

VFR: The *Visual Flight Rules* (VFR) are specified in annex 2 [11], and there is a *Head-on* approach for two or more air crafts. The definition is rather vague: "The pilot should diverge from original heading to the right to create sufficient, safe space for avoidance."

IFR: The *Instrument Flight Rules* in annex 2. [11] and 11. [12] are defining the boundaries and events for success full *Head-on resolution* in larger detail.

The parameter values are useless due to the UAS scaling factor; the following parameters can be used in UTM:

1. The *angle of approach* $\geq 130°$ - the minimal planar angle between aircraft positions and expected collision point is in the interval $[130°, 180°]$.
2. *Minimal detection range* - the minimal detection range of head-on collision is $2 \times turningRadius + safetyMargin$.
3. *Safety margin* - during avoidance all aircraft keeps mutual distance at least the value of safety margin.



(a) Detection. (b) Resolution/Closing.

Figure 6.4: Head-on approach detection/resolution/Closing

Triggering Events: The *head-on approach* (fig. 6.4) *triggering events* are the following:

1. *Detection* (fig. 6.4a) - the *collision case* is open when *collision point* with the respective angle of approach is detected. This must happen until the *point of no return* is achieved.
2. *Resolution* (fig. 6.4b) - the *virtual* roundabout is enforced until the closing condition is met.
3. *Closing* (fig. 6.4b) - based on the condition that all vehicles are heading away from *collision point* and their mutual heading is neutral or opposite.

Virtual roundabout: The *flight levels* can be abstracted as the *virtual 2D surface*. The *airspace attendants* are moving on virtual routes which can cross each other. The idea is to create virtual roundabout with enforced velocity to enable smooth collision avoidance.

1. *Center* - the center defined in *airspace cluster* local coordinate system (flight level defining the horizontal placement).

2. *Diameter* - the minimal distance to *center*, accounting the *wake turbulence* and other phenomena.

3. *Enforced velocity* - all attendants at *virtual roundabout* keeps the same velocity. It helps to keep constant mutual distances.

6.8.5 Handling Converging Maneuver

Goal: Identify *required parameters* sufficient for automatic solution of *Converging Maneuver.*

VFR: The *Visual Flight Rules* (VFR) are specified in annex 2 [11]. The rule is different from *Head-on Approach* (sec. 6.8.4) because multiple roles are depending on the relative aircraft position:

1. *Avoiding Aircraft* - there is an aircraft on the relative right side (blue).

2. *Right Of the Way (ROA) Aircraft* - there is an aircraft on the relative left side (red).

The *avoiding aircraft* should take the *right of the way aircraft* from behind, with sufficient *safety margin*, and return to original *heading* afterward. The *magnitude* of *avoidance curve* must consider *wake turbulence* and other impacts of *avionic properties.*

Note. This rule is applied only when both *aircraft* belong to the same *maneuverability class* [11].

IFR: The *Instrument Flight Rules* in annex 2. [11] and 11. [12] are defining *converging maneuver* in detail.

The *parameters* from a *head-on approach* can be reused:

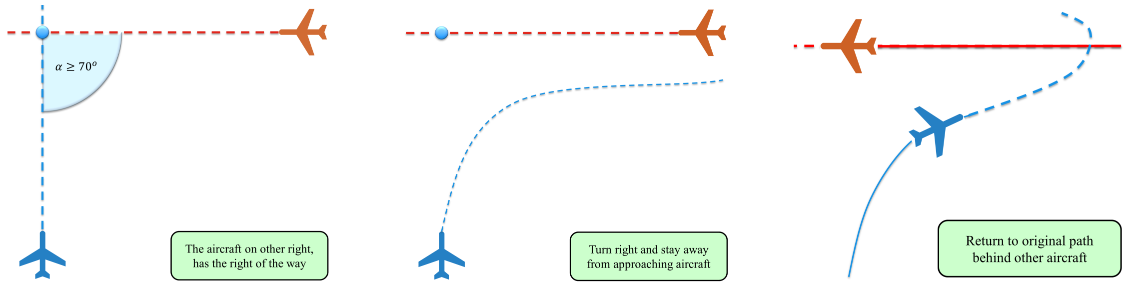
1. $70° \leq$ the *Angle of Approach* $< 130°$ - the minimal planar angle between aircraft position and expected collision point is in the interval $[70°, 130°[$.

2. *Minimal detection range* - given as $turningRadius + safetyMargin$, while *safety margin* is accounting all impact factors.

3. *Safety margin* - during avoidance all aircraft keeps mutual distance at least on the value of *Safety Margin.*

Note. The lesser *angle of approach* induces stronger wake turbulence impact on avoiding aircraft. This results in an increase of *safety margin*.

The *wake turbulence* is represented as a droplet at the back of the plane. *Wake turbulence range* can be calculated based on wake turbulence cone.



(a) Detection. (b) Resolution. (c) Closing

Figure 6.5: Converging maneuver Detection/Resolution/Closing

Triggering Events: The *converging maneuver* (fig. 6.5) *triggering events* are the following:

1. *Detection* (fig. 6.5a) - The *avoiding airplane* (blue) detects *collision point* (blue circle) which satisfy the *converging maneuver conditions*. The distance between *aircraft position* and *collision point* is lesser than the *detection range*.

2. *Resolution* (fig. 6.5b) - the *Right Of the Way aircraft* (red) stays at the original course. The *avoiding aircraft* (blue) follows the *parallel* to another *plane*. The distance of *avoiding plane* to *other plane trajectory* is greater or equal to *safety margin*.

3. *Closing* (fig. 6.5c) - when both planes have an opposite heading, and they miss each other the converging maneuver can be closed. The *avoiding airplane* will return to *original trajectory* while keeping the distance from *another plane* (red) at greater or equal to *safety margin*.

6.8.6 Handling Overtake Maneuver

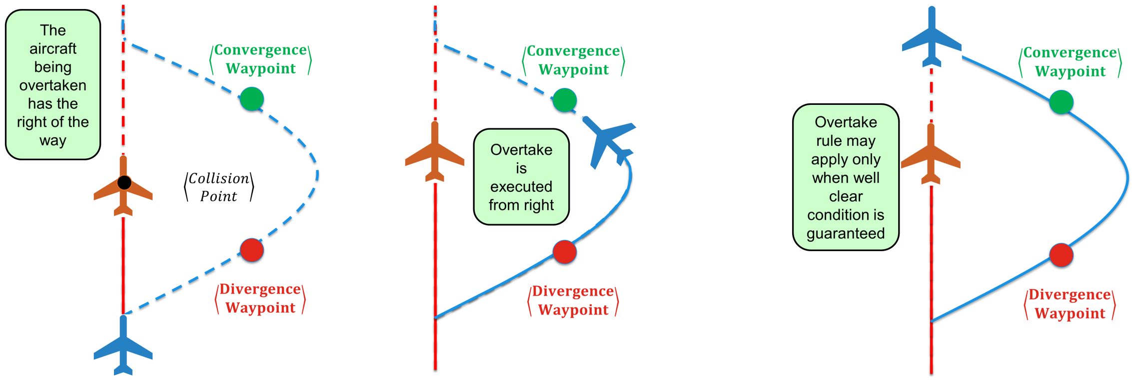
Goal: Identify *required parameters* sufficient for automatic solution of *Overtake Maneuver*

VFR: The *Visual Flight Rules* (VFR) are specified in annex 2 [11]. The rule states that faster air traffic attendant may overtake slower one, from right side keeping sufficient distance (*safety margin*). There are two forced roles:

1. *Overtaking* - faster aircraft with similar heading cruising in similar altitude than *overtaken* (blue). It is expected that *faster aircraft* has maneuvering capability to avoid slower aircraft.

2. *Overtaken* - slower aircraft which keeps the *Right of the way*

Note. This rule is applied only when both aircraft have the same maneuverability class [11]. The overtake is considered *borderline emergency maneuver* in controlled airspace because the aircraft tend to keep similar velocity in similar cruising altitude. The overtake is usual in *non-controlled airspace.*



(a) Detection. (b) Resolution. (c) Closing.

Figure 6.6: Overtake maneuver Detection/Resolution/Closing

IFR: The *Instrument Fight Rules* in annex 2. [11] and 11. [12] are defining the converging manual in detail:

1. $0° \leq$ the *Angle of Approach* $< 130°$ - the minimal planar angle between aircraft position and expected collision point is in the interval $[0°, 70°[$
2. *Minimal Detection Range* - given as $2 \times reactionTime \times speedDifference$.
3. *Safety Margin* - during avoidance the overtaking aircraft keeps the minimal distance of *wake turbulence* of overtaken aircraft in own flight altitude.

Note. The *Safety Margin* is sufficiently small because speed difference is usually much lesser than in case of *Head-on approach.* The *Wake turbulence* can be avoided completely by taking the higher altitude level than overtaken aircraft.

Triggering events:

1. *Detection* (fig. 6.6a) - occurs when the distance between *overtaking* (blue) and overtaken (red) is approaching *minimal detection range* or double of *safety margin.* If the performance of *overtaking aircraft* (blue) allows taking *sharp right side to overtake* the *Maneuver starts*, otherwise *overtaking aircraft* (blue slows down) and keeps at least *safety margin distance* to avoid *wake turbulence.*
2. *Resolution* (fig. 6.6b) - *overtaken* (red) is keeping same heading and *speed* during overtake maneuver. The *overtaking* (blue) projects two waypoints: *Divergence* and

Convergence keeping the required separation minimum during overtake. Then the *overtaking* (blue) diverges heading to *Divergence waypoint*. When the *Divergence waypoint* is reached by *overtaking* (blue) aircraft, it changes to *original heading*.

3. *Closing* (fig. 6.6c) - the *closing* of *Overtake* starts when *overtaking* aircraft (blue) have sufficient lead over *overtaken* aircraft (red). The *overtaking* aircraft (blue) can safely change the heading to the original waypoint.

Constant Cruising Speed: Most of the traffic attendants at same flight level have similar (close to constant) cruising speed. Lower flight levels are for slower turbo-prop planes, and higher altitudes are for jet planes. It is stated that this principle will persist even when UAS will be integrated [13, 14, 15] in multiple air-traffic models.

6.8.7 Position Notification

Motivation: The *position notification* (tab. 6.1) is designed for further *collision case resolution* (sec. 6.8.8). It is similar to ADS-B[1] message information.

The main purpose is to broadcast the *position notification* in *controlled aerospace*. The broadcast for *non-controlled* airspace needs to contain *intruder properties*, *preferred separation mode* and *near-miss margin*.

Position: The position is defined in *Global Coordinate System* using GPS for latitude and longitude. The barometric altitude is required for controlled airspace, preferred for non-controlled airspace.

Heading: The *Linear Velocity* combined with heading in standard *North-East* coordinate frame is used.

Flight Levels: The *flight level* is notified to UTM for *collision detection* purposes. There is a *main flight level* where *aircraft* belong physically. There is a *passing flight level* form which/to which is aircraft emerging [1].

Aircraft Category: The aircraft category impacts the prioritization of *role assessment* by UTM/ATM. The following categorization is proposed by *manned aviation pilot community*, from the highest to the lowest right of the way priority:

1. *Manned aviation in distress* [11] - the aircraft with impaired capability switched to emergency mode. The emergency mode is usually acknowledged by the authority in controlled airspace.

2. *Balloon* (manned) [11] - the aircraft with *altitude* control and very slow dynamics implying very low maneuverability.

[1] ADS-B versions and message containment: `https://mode-s.org/decode/adsb/version.html`.

3. *Glider* (manned) [11] - the aircraft with *full control* but without own *propulsion*. The overall *maneuverability* is good, but the *velocity* changes are impossible with sufficient flexibility.

4. *Aerial towing* (manned) [11] - the towing aircraft usually have *own propulsion* and full maneuverability, the only constraint is *towed load*. The towed load decreases overall maneuverability.

5. *Airship* (manned) [11] - the airship have *own propulsion* and full maneuverability, the constraint is low acceleration/deceleration and huge turning radius.

6. *Other manned aviation* [11] - containing all vehicles with the required level of *airworthiness* for given operational *altitude*. They usually have required maneuverability.

7. *UAS Autonomous* (proposed) [16] - containing all autonomous UAS, the lower flexibility is expected at the beginning of integration.

8. *Remotely Piloted Aerial System (RPAS)* (proposed) [16] - has lesser priority due to the higher response rate of the pilot.

Note. This categorization reflects only Pilot community statement; the general priority rule is broken, because maneuverability and vulnerability should always be considered as a key decision factor.

Maneuverability: The maneuverability is the real key factor in priority assessment. The components of maneuverability are *maximal/mean acceleration/deceleration*, *climb/ descent rate* and *turning ratio/radius*. The comparison can be made by solving *pursuit problem* using *Reach Sets* [17, 18].
The *Maneuverability categorization* is based on *original aircraft priority categorization* [11] accounting UAS/RPAS as equal to *manned aviation*. The ordered list from the highest to the lowest priority goes as follows:

1. *Impaired control* (Distress aircraft) - any aviation attendant in distress has the priority in case of the conflict occurrence.

2. *Altitude control/No* (Balloon, Hovering aircraft) - the balloon type crafts do not have any type of propulsion, and horizontal movements follow the airflow in given altitude.

3. *Full control/No propulsion* (Gliders of any sort) - the gliders can control their horizontal position, but there are limits to altitude control and acceleration/deceleration.

4. *Full control/Linear propulsion* (Any aircraft of plane type) - the *towing aircraft's* and *airplanes* belong there; the difference is the *flexibility* of *maneuvering*.

5. *Full control/VTOL capability* (Any aircraft with VTOL) - the *other aircraft* capable of doing on-spot-turn. The typical representative is *quad-rotor copter*.

There are other aspects like *minimal required* acceleration/deceleration/turn ratio to operate in a selected segment of the *airspace*. These should be specified later by *Minimum Operational Performance Standards* (MOPS).

	Position
latitude	based on GPS/IMU sensor fusion.
longitude	based on GPS/IMU sensor fusion.
altitude	barometric altitude *Above Mean Sea Level* (AMSL).
	Heading
orientation	orientation in standard North-East coordinate frame.
velocity	relative UAS velocity.
	Flight Levels
main	flight level, where UAS mass center belongs
passing	flight level, during climb/ascend, or when distance of UAS mass center to flight level boundary $\leq 250ft$.
	Registration
registration ID	is unique registration number *to be issued* by local aviation authority for UTM communications purposes.
flight code	or mission code is a unique identification number for approved mission plan which is going to be flown by UAS.
UAS name	optional UAS identifier to increase human recognition.
	Categorization
craft category	ICAO main category, based on vehicle type.
maneuverability	secondary categorization is specifying size class, horizontal/vertical turning radius, minimal and maximal cruising speed.
	Safety margins
universal	minimal safety margin for any avoidance situation
head-on	minimal distance from other similar maneuverability class aircraft in case of a head-on approach.
converging	minimal distance from other similar maneuverability class aircraft in case of the converging maneuver.
overtake	minimal distance from other similar maneuverability class aircraft in case of overtake maneuver.
wake angle	for wake turbulence cone.
wake radius	for wake turbulence cone.

Table 6.1: Time-stamped *position notification* structure.

Safety Margins: The *Safety Margin* for *Well Clear Condition* value is based on the *situation*. There is also a *universal safety margin* which guarantees the minimal safety for encountering intruder.

The most prevalent effect is *Wake turbulence*, therefore, *wake turbulence cone* angle $[0\circ - 90\circ]$ and radius.

The *safety Margin* for situation-based avoidance is given by the list of supported maneuvers; there is converging (sec. 6.8.5), head-on (sec. 6.8.4), overtake (sec. 6.8.6) safety margins.

6.8.8 Collision Case

Collision Case Purpose: There is a need for detection and tracking of possible *controlled airspace traffic attendants* collisions. The presented *collision case structure* (tab. 6.3) is a minimalist reflection of *ATM* requirements. Following aspects of *collision case* life cycle are explained in this section:

1. *Base terminology* - the definition of *enforcement procedure* and difference between *Resolution* and *Mandate* from UTM authority. The *severity issue* is open.

2. *Calculation of single case for single decision frame* - step by step calculation and threat evaluation. Prequel to the *life-cycle*.

3. *Life cycle* gives outlook on how collision case data are handled through a longer period, notably: *Opening*, *collision point handling*, *safety margin handling*, and, *Closure*.

4. *Merge procedure for multiple cases in a single cluster* - the naive *merge procedure* to solve *multiple collision cases* via the *virtual roundabout*.

Resolution/Mandate Enforcement: *Enforcement procedure* is consisting from *Threat detection phase* and *Mitigation phase*. The *mitigation phase* is a time interval when *UTM* decision is enforced. The decision the UTM is enforcing is delivered in the form of *Resolutions* and *Mandates*.

A *Resolution* is an order from the *UTM* authority which is followed by subjected UAS. The *subjected UAS* can determine own behavior to some extent. When there is an emerging threat or another destructive event, like a new non-cooperative adversary, the UAS is allowed to broke *resolution*.

A *Mandate* is an order from the *UTM* authority which can not be broken at any cost. The example of the *mandate*: UAS is flying in the airspace, the passenger in distress needs it to safely land. The UAS must obey mandate even at the event of own destruction.

Threat Severity Evaluation: The threat severity evaluation is omitted partially, all threats are considered as equal. All commands from *UTM authority* will be considered as *resolutions*.

Calculation procedure: Collision case is calculated for two *Registered UAS systems* in *Unified UTM time-frame*. The *unified UTM time-frame* is a short period in future when the anticipated situations are predicted.

1st The *position* and *orientation* are adjusted according to the *mission plan*. Our implementation uses *Movement Automaton* as a predictor:

$$\begin{aligned} adjustedPosition &= Position(Trajectory(notifiedState, futureMovements)) \\ adjustedOrientation &= Orientation(Trajectory(notifiedState, futureMovements)) \end{aligned} \tag{6.2}$$

For other cases standard linear prediction can be used:

$$\begin{gathered} adjustedPosition = notificationPosition \times notificationVelocity \times timeDifference \\ adjustedOrientation = notificationOrientation \end{gathered} \tag{6.3}$$

2nd The *maneuverability*, *craft category*, *registration ID* are taken from *position notification*.

3rd *Collision case check procedure* goes like follows:

1. *Operation space checks* - the controlled airspace and flight level must match for proceeding.

2. *Maneuverability/Category check* - the maneuverability and UAS category must match. If there is mismatch, then the right of the way is forced to the vehicle with higher priority.

4th *Linear Intersection test* is designed to calculate *closest distance* and *time* of *linear trajectory projections*. First, for given *velocity* and *position* for UAS1 and UAS2 the helper variables are calculated:

$$\begin{aligned} &A = \|velocity_1\|^2 \\ &B = 2 * ({velocity_1}^T \times position_1 - {velocity_2}^T \times position2) \\ &C = 2 \times {velocity_1}^T * velocity_2 \\ &D = 2 * ({velocity_2}^T \times position_2 - velocity_2{}' \times position_1); \\ &E = \|velocity_2\|^2; \\ &F = \|position_1\|^2 + \|position_2\|^2; \end{aligned} \tag{6.4}$$

Then the projection parameters can be calculated:

$$\begin{aligned}
time &= \frac{-B - D}{2 \times A - 2 \times C + 2 \times E} \\
destination_i &= position_i + velocity_i \times time, \quad i \in \{1,2\} \\
collisionPoint &= \frac{destination_1 + destination_2}{2} \\
collisionDistance &= \|destination_1 - destination_2\|
\end{aligned} \tag{6.5}$$

If $time < 0$ the trajectories are diverging from each other (because the closest points already occurred). The procedure ends, the *collision flag* is not raised.

If $time > timeMargin$ the trajectories will get close to each other, but in further future and changes are anticipated. The procedure ends, the *collision flag* is not raised.

If $0 \leq time \leq timeMargin$ the trajectories are converging to each other and distance needs to be checked. If $distance \leq collisionMargin$ then *collision flag* is raised and *collision point* is set.

Note. Collision Margin is some number which is determined based on aircraft category and maneuverability. Our work defines collision margin as follow:

$$collisionMargin = \forall situation : \max \begin{Bmatrix} safetyMargin(situation, UAS1) \\ +safetyMargin(situation, UAS2) \end{Bmatrix} \tag{6.6}$$

Where the *safety margin* for every possible situation is evaluated for both *UAS*.

5th The *trajectory* intersection is *Movement Automaton* specific collision detection method. Its based on the assumption that *UTM* has the following information from *mission plan*:

1. *UAS state* - not only *position*, *orientation*, and, *velocity* vectors, but other mathematical model parameters mandatory for *movement automaton*.

2. *Movement Automaton* - movement automaton for our UAS system, so that UTM can use it in predictor mode.

3. *Future Movements set* - up to reasonable prediction horizon $timeMargin$.

The *Movement Automaton* can be used as trajectory prediction for initial system state and future movements. The prediction function (eq. 6.7).

$$Prediction : UAS \times state \times futureMovements \rightarrow [x, y, z, t] \in \mathbb{R}^4 \tag{6.7}$$

Note. Then prediction for UAS1 is $Prediction_1$, and for UAS 2 $Prediction_2$, the predictions are synchronized meaning that time at position i is equal in both discrete trajectory matrices.

The *collision distance* for predictor (eq. 6.7) is given as minimal distance of projected synchronized trajectories for UAS1 and UAS2. In our discrete enviroment, the *collision distance* is given as (eq. 6.8).

$$collisionDistance = \min \left\{ \|point_1 - point_2\| : \forall \begin{pmatrix} point_1 \in Prediction_1, \\ point_2, \in Prediction_2, \\ t_1 \sim t_2 \end{pmatrix} \right\} \tag{6.8}$$

If $collisionDistance \leq collisionMargin$ condition is met, *collision flag* is set.

The collision point is then calculated as mean of *UAS positions* in prediction at a time when the distance is minimal. The final collision point is arithmetic mean of two positions (eq. 6.9).

$$collisionPoint = \frac{point_1 - point_2}{2} : \begin{pmatrix} point_1 \in Prediction_1, \\ point_2, \in Prediction_2, \\ t_1 \sim t_2 \text{ at minimal distance} \end{pmatrix} \tag{6.9}$$

Note. Collision point is overwritten by trajectory intersection (specific) method; the *linear intersection* is considered a *general collision detection method*. The collision detection method in future UTM system needs to be determined. The *Trajectory intersection* method presented in this work is one of the possible candidates.

6th *Role determination* phase is invoked if and only if previous conditions are met and *collision flag* with *collision point* exists.

There is *adjusted position* of each UAS used as verticals and *collision point* used as a center. The first step is normalization of adjusted position around collision point for both UAS:

$$normalized_i = adjustedPosition_i - collisionPoint, \quad i \in \{1, 2\} \tag{6.10}$$

Then the right-hand coordinate system internal angle calculation method is used:

$$angleOfApproach = \left| \operatorname{atan2} \begin{pmatrix} normalized_1 \times normalized_2, \\ normalized_1 \circ normalized_2 \end{pmatrix} \right| \tag{6.11}$$

Based on the *angle of approach* the *scenario type* is decided like follows:

1. $130° \leq angleOfApproach \leq 180°$ - the scenario type is set as *Head On Approach* (sec.6.8.4)

2. $70° \leq angleOfApproach < 130°$ - the scenario type is set as *Converging Maneuver* (sec.6.8.5)

3. $0^\circ \leq angleOfApproach < 70^\circ$ and *different speed* - - the scenario type is set as *Overtake Maneuver* (sec.6.8.6)

Based on *relative position* and *scenario type*, the *avoidance role* like follows:

1. *Head On Approach* enforces the following:
   a. The *avoidance role* is set as *RoundAbounting* for both UAS.
   b. None of the *UAS* does have the *Right Of the Way*.
2. *Converging Maneuver* enforces the following:
   a. *UAS* without free right side has a role set as *Converging*.
   b. *UAS* with free right side has the *Right Of the Way*.
3. *Overtake Maneuver* enforces the following:
   a. *Slower UAS* has *Overtaken* role with *Right Of the Way*.
   b. *Faster UAS* has *Overtaking* without *Right Of the Way*.
   c. *Faster UAS* mission plan is altered with *divergence and convergence waypoints*.

7th *Safety Margin Calculation* Is invoked when the collision case is *Active*. The *Active Collision Case* in this time-frame means that *Collision Flag* is raised. The *avoidance role* determines *safety margin calculation*.

If *Head-On Approach* is case type of *Head collision case* then *safety margin* is calculated as the maximum of the sum of *default* margins or *head on* margins:

$$safetyMargin = \max \begin{Bmatrix} default(UAS1) + default(UAS2), \\ headOn(UAS_1) + headOn(UAS_2) \end{Bmatrix} \tag{6.12}$$

If *Converging Maneuver* is case type of *Head collision case* then *safety margin* is calculated based on *avoiding UAS* as the maximum of opposing UAS *default margin* and avoiding *converging margin*:

$$safetyMargin = \begin{cases} uas1.role = Converging : & \max \begin{Bmatrix} default(UAS2), \\ converging(UAS1) \end{Bmatrix} \\ uas1.role = Converging : & \max \begin{Bmatrix} default(UAS1), \\ converging(UAS2) \end{Bmatrix} \end{cases} \tag{6.13}$$

If *Overtake maneuver* is case type of *Head collision case* then *safety margin* is calculated as the maximum of *default, overtaking, overtaken* margins of both UAS:

$$safetyMargin = \max \begin{Bmatrix} default(UAS1), default(UAS2), \\ overtaken(UAS_1), overtaking(UAS_2), \\ overtaking(UAS_1), overtaken(UAS_2) \end{Bmatrix} \tag{6.14}$$

Collision Case Chaining is procedure when multiple active collision cases for different *time-frame* are chained and creates the time ordered series of *collision cases.* There are two notable instances in the *chain*:

1. *Head Collision Case* - Collision case when the first danger was detected. The notable parameters are *collision point* and UAS *avoidance roles* because these are enforced by the *Rule engine* (sec. **??**). The *head collision case* is first in the chain.
2. *Tail Collision Case* - Collision case when the *collision danger* was not detected. The *tail collision case* is last in the chain.

Note. The *Chaining* of *collision cases* is rather primitive and sensitive for errors/noise.

The *Consistency of Avoidance Maneuver* is ensured by enforcing *head collision case* parameters.

Data for both attendants

adjusted position	predicted from previous *position notifications* (6.1) data at the time of *UTM decision frame* start.
adjusted orientation	predicted from previous *position notifications* (6.1), *mission plan*, and *expected velocity.*
velocity	proclaimed velocity for given *UTM decision time frame.*
registration ID	is unique registration number issued by the local aviation authority
craft category	from *position notifications* (6.1).
maneuverability	from *position notifications* (6.1).
mission plan	is acquired from *allowed mission registers* where it has been registered prior UAS flight
safety margins	list of all safety margins derived based or craft categorization or overridden by *position notifications* (6.1).
avoidance role	is given based on situation evaluation.
trajectory prediction	simulated based on *position notification* (6.1) and *mission plan.*

Table 6.2: Collision case structure attendant data.

Collision Cases Merge also known as *Collision Point Adjustment Procedure* purpose it to *merge* multiple collision cases into one general collision case. The clustering is used to identify *airspace congestion events* [19]. Example of *airspace clustering* is given it [20].

The main idea is to *encapsulate multiple collision cases* into one virtual roundabout to ease *traffic load* [21]. The potential risk on *turbo roundabouts* have been outlined in [22].

There are *active collision cases* in a focused *cluster* in *controlled airspace.* The multiple collision cases can pop up at different *start times*, and they can be active for a different *period*.

The *Collision point* is replaced with the *roundabout center* point (eq. 6.15). The *roundabout center* is calculated as weighted average of *active collision cases* collision points. The *weight* $\in [0,1]$ depending on severity rating of collision case.

$$roundaboutCenter = \frac{\sum_{\in Cluster}^{\forall collisionCase} collisionCase.collisionPoint \times weight}{|collisionCase \in Cluster|} \tag{6.15}$$

Note. The weight in (eq. 6.15) is set to 1 for all time; the weight calculation needs to be determined in future works.

The *smallest circle problem* defined and solved in [23, 24] is used to determine the safety margin in our approach. The *naive approach* determining *roundabout safety margin* is to take the maximum of all open case *safety margins* including default ones (eq. 6.16).

$$safetyMargin = \max \begin{Bmatrix} case.UAS_i.roundaboutSafetyMargin, \\ case.UAS_i.defaultSafetyMargin \end{Bmatrix}, \\ \forall case \in Cluster, \quad UAS_i \in \{1,2\} \tag{6.16}$$

Collision case calculated data	
linear intersection	is predicted on attendants *position*, *heading*, *velocity*, based on *maneuverability* certain thresholds are applied to determine safety properties.
trajectory intersection	is predicted on attendants *position*, *velocity*, *heading*, and *related mission plans*, based on *maneuverability* certain thresholds are applied to determine safety properties.
collision point	is created if there is the risk of medium/short period collision, if head collision case has not been closed, collision point is inherited.
adj. collision point	is created if there exists at least one active collision case in the nearby surroundings of this case collision point (cluster).
angle of approach(α)	is calculated based on attendants *velocity* and *position*, the range is $[0°, 180°]$, it determines *primary avoidance roles.*
safety margin	is calculated based on *avoidance roles*, *maneuverability*, collision indicators, and *angle of approach.*
margin adjustment	is calculated based on *linked collision cases*, *estimation errors* and *weather.*
linked cases	contains a list of collision cases which are active and can have an impact on this *collision case.*
head case	is a reference to collision case in the time frame when it was first opened.

Collision case indicators	
linear intersection	indicates if there was a safety breach on linear trajectories estimation with the risk of direct collision.
trajectory intersection	indicates if there was a breach on trajectory estimation, with the risk of direct collision.
well clear breach	indicates if *linear projection* or *trajectory projection* breaches *well clear barrel* in *controlled airspace.*
active case	indicates if the case is still open.

Table 6.3: Collision case structure for given decision time-frame.

6.8.9 Weather Case

Motivation: The weather, as defined in (eq. **??**), impacts flight and system dynamics; therefore it impacts the *reach set* is impacted. The *weather impact* can be solved by policy application:

1. *Weather Acceptance* - for bigger *UAS* the normal weather impact does not pose a significant risk. The *segmented movement automaton* (def. **??**) with *Weather situation* as the discrete state is used.

2. *Weather Avoidance* - all *weather* impact zones are considered as hard constraints with protective *soft-constraint* around.

3. *Combined approach* - depending on the type of impact and declared UAS impact resistance the zones are divined into *soft* and *hard* constraints.

Note. This work handles small *UAS* avoidance; these are very sensitive to any weather impact; therefore *Weather impacted areas* will be considered as *hard constraints with soft constraint protection zone.*

The original *weather impact zone* is considered as obstacle body and enforces the body margin.

The surroundings of *weather impact zone* up to *safety margin* distance are considered as *soft constraint zone* (implemented as a bloated polygon).

Purpose: The *weather case* (tab. 6.4) is broadcasted by *Airspace Authority* to *impacted area*, each *UAS* then change their mission according to *their maneuvering capabilities.* Each trajectory must lead away from the *constrained area.* The algorithm used for intersection selected based on [25] the selected algorithm *Shamos-Hoey* [26].

Constrained Area: Constrained area can be defined as *static* (sec. **??**) or dynamic constraint (sec. **??**). The *constraint center* is defined on horizontal plane like follow:

$$ConstraintCenter = center \in [latitude, longitude] \tag{6.17}$$

The *Convex Polygon* boundary is defined on horizontal plane, contains at least 3 vertexes:

$$ConvexPolygon = \{point_i : point_i \in [latitude, longitude], i \geq 3\} \tag{6.18}$$

The *Vertical constraint* is defined as *range of barometric altitude* (Above Mean Sea Level):

$$VerticalConstraint = [startAltitude, endAltitude] \tag{6.19}$$

Additional parameters : Following additional parameters with additional purpose can be attached to *Weather Constraint*.

1. *Type* - defines required resistance - moisture, temperature, wind.
2. *Severity* - defines the impact for each *aircraft category*, this is used in soft/hard type assessment.
3. *Duration* - start and end of *constraint* validity, if not defined valid for all *UAS mission time*.
4. *Velocity* - velocity and last position assessment time.

Note. Our implementation does not consider the *type* or *severity*. All *weather impact* is considered as a *hard constraint*. The velocity differentiates *static* $(= 0)$/*moving* (> 0) *constraints*.

Avoidance System: Resolve similar to *Converging/Overtake Maneuver* depending on the *angle of approach*. The *virtual roundabout* is utilized for *static constraints*; the *intruder model* is utilized for *dynamic constraints*.

Constrained area	
center position	is given as a geometrical *center point of the boundary*.
boundary	is represented as a *convex polygon* on the latitude-longitude plane.
start altitude	is lover boundary barometric altitude given at above mean sea level, where given weather factor has a significant impact.
end altitude	is upper boundary barometric altitude given at above mean sea level, where given weather factor has a significant impact.
Additional parameters	
type(s)	lists weather events occurring in the *constrained area*.
severity list	is recorded for each plane *category*
start	indicates when weather constraint was established.
expected end	of weather constraint.
velocity	indicates if weather phenomenon is moving.
Miscellaneous	
previous	reference to *weather constraint* decision time-frame data.
impacted	list of possibly impacted attendees (planes which obtained divergence order or warning from UTM).

Table 6.4: Static/Dynamic weather constraint for given decision time-frame.

Bibliography

[1] ICAO. 4444: Procedures for air navigation services. Technical report, ICAO, 2018.

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

[3] Thomas P Spriesterbach, Kelly A Bruns, Lauren I Baron, and Jason E Sohlke. Unmanned aircraft system airspace integration in the national airspace using a ground-based sense and avoid system. *Johns Hopkins APL Technical Digest*, 32(3):572–583, 2013.

[4] Karthik Balakrishnan, Joe Polastre, Jessie Mooberry, Richard Golding, and Peter Sachs. The roadmap for the safe integration of autonomous aircraft. Blueprint for the sky - Airbus, www.utmblueprint.com, sep 2018.

[5] Nico Zimmer, Jens Schiefele, Keyvan Bayram, Theo Hankers, Sebastian Frank, and Thomas Feuerle. Rule-based notam & weather notification. In *Integrated Communications, Navigation and Surveilance Conference (ICNS), 2011*, pages O1–1. IEEE, 2011.

[6] Thomas Prevot, Joseph Rios, Parimal Kopardekar, John E Robinson III, Marcus Johnson, and Jaewoo Jung. Uas traffic management (utm) concept of operations to safely enable low altitude flight operations. In *16th AIAA Aviation Technology, Integration, and Operations Conference*, page 3292, 2016.

[7] Nico Zimmer and Keyvan Bayram. Selective weather notification, March 18 2014. US Patent 8,674,850.

[8] Subramanian Ramasamy, Roberto Sabatini, and Alessandro Gardi. Towards a unified approach to cooperative and non-cooperative rpas detect-and-avoid. In *Fourth Australasian Unmanned Systems Conference*, 2014.

[9] Subramanian Ramasamy, Roberto Sabatini, A Gardi, and Yifang Liu. Novel flight management system for real-time 4-dimensional trajectory based operations. In *proceedings of AIAA Guidance, Navigation, and Control Conference*, 2013.

[10] Branka Subotic, Arnab Majumdar, and Washington Y Ochieng. Recovery from equipment failures in air traffic control (atc): The findings from an international survey of controllers. *Air Traffic Control Quarterly*, 15(2):157–181, 2007.

[11] ICAO. Annex 2 (rules of the air). Technical report, ICAO, 2018.

[12] ICAO. Annex 11 (air traffic services). Technical report, ICAO, 2018.

[13] Alexandre Bayen, Pascal Grieder, George Meyer, and Claire J Tomlin. Langrangian delay predictive model for sector-based air traffic flow. *Journal of guidance, control, and dynamics*, 28(5):1015–1026, 2005.

[14] Parimal Kopardekar and Sherri Magyarits. Dynamic density: measuring and predicting sector complexity [atc]. In *Digital Avionics Systems Conference, 2002. Proceedings. The 21st*, volume 1, pages 2C4–2C4. IEEE, 2002.

[15] MP Helme, K Lindsay, SV Massimini, and G Booth. Optimization of traffic flow to minimize delay in the national airspace system. In *Control Applications, 1992., First IEEE Conference on*, pages 435–437. IEEE, 1992.

[16] Confesor Santiago and Eric R Mueller. Pilot evaluation of a uas detect-and-avoid system's effectiveness in remaining well clear. In *Eleventh UAS/Europe Air Traffic Management Research and Development Seminar (ATM2015)*, 2015.

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

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

[19] Karl Bilimoria and Hilda Lee. Analysis of aircraft clusters to measure sector-independent airspace congestion. In *AIAA 5th ATIO and16th Lighter-Than-Air Sys Tech. and Balloon Systems Conferences*, page 7455, 2005.

[20] CR Brinton and S Pledgie. Airspace partitioning using flight clustering and computational geometry. In *Digital Avionics Systems Conference, 2008. DASC 2008. IEEE/AIAA 27th*, pages 3–B. IEEE, 2008.

[21] M Ebrahim Fouladvand, Zeinab Sadjadi, and M Reza Shaebani. Characteristics of vehicular traffic flow at a roundabout. *Physical Review E*, 70(4):046132, 2004.

[22] Raffaele Mauro and Marco Cattani. Potential accident rate of turbo-roundabouts. In *4th International Symposium on Highway Geometric DesignPolytechnic University of ValenciaTransportation Research Board*, 2010.

[23] Jack Ritter. An efficient bounding sphere. *Graphics gems*, 1:301–303, 1990.

[24] Emo Welzl. Smallest enclosing disks (balls and ellipsoids). In *New results and new trends in computer science*, pages 359–370. Springer, 1991.

[25] Jon Louis Bentley and Thomas A Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on computers*, (9):643–647, 1979.

[26] Michael Ian Shamos and Dan Hoey. Geometric intersection problems. In *17th annual symposium on foundations of computer science*, pages 208–215. IEEE, 1976.