Chapter 5. State of the Art

This chapter is introducing notable works and concepts of *researchers* in the field of *control theory*, *software engineering* and other essential fields used in *Detect and Avoid* systems.

5.1. Movement Control

Idea: The key idea is to create *interface* between *controlled plant* (UAS) and *Avoidance Algorithm* to ensure *Concept Reusability* at maximum degree. The concept is the following:

1. *Interface consumes* discrete command chain and guides UAS along the *desired trajectory*.
2. *An interface* can be used to *predict trajectory* based on *initial state* and future command chaining.

Frazzoli Movement Automaton: The following paragraph strongly follows Frazzoli work [1] (sec 3.1-3.5). Frazolli provided the concept of *Movement Automaton* (def. **??**) a specialized type of *Hybrid Automaton* (eq. **??**), the concept is taken from his works [1, 2]. Other aspects and similarities are discussed in this chapter.

The approach was proposed to reduce the *computational complexity problem* of *motion planning*. The quantization of the system dynamics is done through restriction of feasible nominal system trajectories to the *family of time parametrized curves*. These can be obtained by the interconnection of trajectory primitives.

Trajectory primitives are repeatable portions of trajectory (def. [1].3.1). The trajectory primitives are interconnected by *transitions* to create maneuvers (movements) (def. [1].3.3).

By combining movements as a set of trim trajectories the trajectory can be represented as a set of discrete time bounded commands. This is summarized in the definition (def. [1].3.4) based on *hybrid automaton definition* (sec. **??**).

Definition 1 (Maneuver Automaton)**.** *A maneuver automaton over a mechanical control system S, with symmetry group H is described by the following objects:*

1. *A finite set of indices* $Q = Q_T \cup Q_M \in \mathbb{N}$*, where the subscript T relates to trim trajectories, and the subscript M relates to maneuvers;*
2. *A finite set of trim trajectory parameters* $\left(\bar{g}, \bar{\xi}, \bar{u}\right)_q$*; with* $q \in Q_T$*.*
3. *A finite set of maneuver parameters, and state and control trajectories* $(T, u, \varphi)_q$*, with* $q \in Q_M$*.*

4. *The maps Previous* : $Q_M \to Q_T$, *and, Next* : $Q_M \to Q_T$ *such that Previous*(q) *and Next*(q) *give, respectively, the index of the trim trajectories from which the maneuver q starts and ends.*

5. *A discrete state* $q \in Q$.

6. *A continuous state, denoting the position on the symmetry group,* $h \in H$.

7. *A clock state* $\theta \in \mathbb{R}$, *which evolves according to* $\dot{\theta} = 1$, *and which is reset after each switch on q.*

Note. It is apparent that decisions can be made about the future evolution of the system only when the system is executing a trim trajectory (that is, the discrete state is in one of the nodes in the graph). While executing a maneuver the system is committed to it and must keep executing the maneuver until its completion. As a consequence, for motion planning and control design purposes, one can concentrate the study of the evolution of the system on and between nodes.

Architecture: The Movement Automaton can be seen as a consistent hierarchical abstraction of the continuous dynamics, in sense outlined in [3]: *Any sequence of movement primitives generated by the Movement Automaton results by construction in a trajectory which is executable by the full continuous system. We will give a deeper meaning to hierarchical consistency*.

Optimal Path Generation: If the maneuvers are instantaneous (i.e., the UAS can transition instantaneously between two different trim trajectories), Reduction of stronger results obtained by Dubins [4] and Reeds [5] concerning optimal paths for kinematic cars on the plane (see also [6]).

Controllability: The systems controlled by Movement Automaton (as in [7]), is controllable according to our definition, even though it is not small-time controllable [8].

Other Properties: The other properties of movement automaton, like *Stability, Robustness* and other important control properties are proven in [1].

Example: The *example* is given in (fig. 5.1). The *States* (Barrels) are connected by *Transitions* (green arrows).

Hover is the neutral and *initial* state, in this place the UAS stays on place and maintains altitude.

Forward flight is when *UAS* is flying in frontal direction with constant speed. The speed-up and slow-down are incorporated in *Transition* between *Hover* and *Forward flight* states, and it takes some time to execute. *Transitions* between Turning states and *Flight forward* state are almost instant.

Steady turn left/right is when *UAS* is flying in the frontal direction and starts steady turning left or right.

Note. UAS in (fig. 5.1) ignores the vertical maneuvering, and it is expected to fly on horizontal plane.

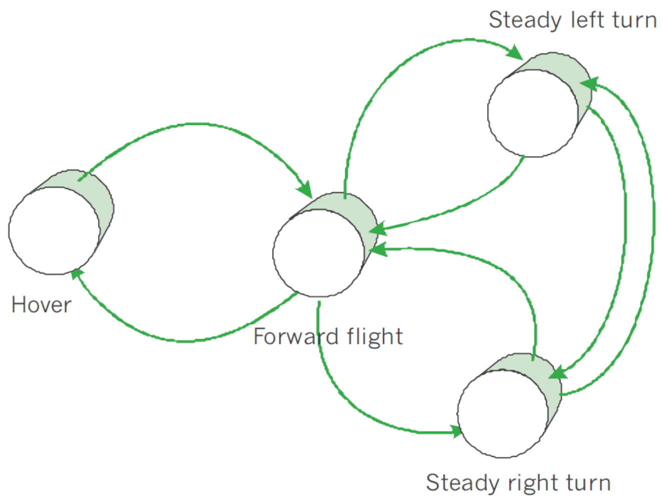


Figure 5.1: Movement Automaton for Copter UAS [1].

Note. The *Movement Automaton* is used as with modification (sec. **??**). The implementation is described in (sec. **??**). The testing configuration was given in (tab. **??**).

5.2. Surveillance

Idea: There is a *need for the abstract representation* of *operational space*, which is independent of used sensors, technologies, information sources. The universal obstacle avoidance system should be *portable* between various platforms. Our previous work *Obstacle avoidance framework based on reach sets* [9] has introduced a concept of the *control interface*. The concept of *control interface* increases portability of the solution.

The original concept used cell status interpretation (boolean values), which is hardwired to LiDAR technology. The basic methods for *Statistical Sensor Fusion* were outlined in [10]. The result of the application of the method is *data fusion interface* (sec. **??**) - interface to fuse sense data from various online, offline, cooperative, non-cooperative sources into single scalable space and trajectory evaluation procedure.

Related work: UAS specific sensor fusion has been proposed by Ramsay in [11]. *Next generation avoidance concept* [12] is introducing concept of higher-level sensor fusion called *data fusion*.

The uncertainty and properties in *Remotely Piloted Systems* have been discussed in [13]. The work provided the concept of various performance ratings like visibility and obstacle rating; more details have been given in [14]. These ratings were modeled only for operator decision making [15], results are usable for automated decision making and space assessment.

5.3. Navigation Algorithms

Idea: The basic idea is to provide hierarchical *navigation frame* with *some optimal path search capabilities*.

Space Assessment: *Probabilistic trajectory assessment* has been firstly proposed in [16] where trajectory was tracking and predicting *safety properties* along.

Game theory viewpoint is firstly used in [17]. Probabilistic path planning using safety zones similar to cell classification of this work has been used in [18].

Probabilistic path search similar to our reach set representation using rapidly exploring path trees have been used in [19, 20]. The relationship between clasic grid search and probabilistic lattice search have been established in [21]. A probabilistic approach for trajectory estimation via reduced lattice search is known from 1986 from work of Gessel [22] lattice paths were enumerated via movement sequences and a similar technique is used in our reach set estimation method using movement automaton. Pruning methods comparison and complexity can be found in [23].

Overall concepts of probabilistic sets have been given by Hirota in [24]. Free flight safety rating similar to our reachability concept has been presented in [25].

Standard Navigation: The standard navigation is given as *expected cost optimization problem* for *future cost function* (eq. **??**). The key concept of navigation algorithm was fully taken from [26]. The decision was made based on navigation survey [27]. The *descent* for landing is out of scope in this work, can be found in [28]. The navigation principle is roughly described in (sec. **??**).

Maze Solving Capabilities: The *maze solving capability* is usable in *controlled airspace* where 2D maze solving algorithms are applicable. The notable implementation was for *micro mouse robot* based on right-hand rule [29]. Flood fill algorithm is partially usable for 3D environment [30]. The application of *maze solving* was given in case study [31].

Hybrid Automaton Path Planning: A hybrid automaton path planning based on $A*$ algorithm was given by Richards in [32]. The key idea was to use *hybrid automaton* (eq. **??**) as a reference generator. This idea was taken and formulated as *Movement Automaton Predictor mode*.

The similar idea where *potential fields* were used as the *intruder model* and path was re-planned based on events is given in [33].

Mode Switch: The *Mode Switch Control* idea has been presented in [34]. There was the definition of behavioral switch between:

1. *Navigation Mode* - navigation control and behavior was used.
2. *Task-Specific Mode* - mode specific for tasks, authors were using modes for search and rescue.

This concept will be reused; the *task-specific mode* will be *Emergency Avoidance Mode* in Our case. The triggering events and switch conditions will be defined in (sec. **??**).

Used Concepts: The *Following concepts* were used in navigation loop:

1. *Standard navigation* took from [26] minor implementation changes using offline optimization. The purpose of the navigation loop is to bring us closest to the waypoint if it is reachable. Navigation example (sec. **??**).

2. *Maze solving capabilities* partially taken as secondary functionality based on [30]. The purpose is *looping prevention*. The example was given in (sec. **??**).

3. *Mode switch* partially taken as the main feature from [34], the triggering events were identified and defined by author and can be found over (chapter **??**).

Shortcomings:

1. *Hierarchical calculation* - there is a need to calculate the *avoidance trajectory* for incremental constraint applications. For example:

   a. Calculate *Minimal escape path* covering physical obstacles and intruders.

   b. Apply next level of constraints, like airspace restrictions and some virtual constraints. Then calculate path if exists, continue.

   c. Apply nice to have constraints, like non-lethal weather, recalculate the path.

2. *Source Reliability Evaluation* - reliability evaluation is an empirical process usually done by hand. The result aggregation is not standardized. There can be multiple sources of the same rating, for example, visibility, which needs to be aggregated into one.

3. *Ambiguous rating definition* - There are multiple definitions especially for *Reachability rating* in works [19, 20, 22].

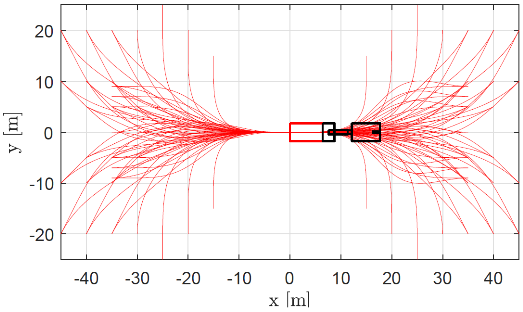
5.4. Reach Set Estimation

Idea: The basic idea for *Discrete Reach set Estimation method* is taken from [35]. The focus of their work is to generate paths that are kinematically feasible. Path following controllers in order to find techniques to stabilize the system around these paths [36, 37].

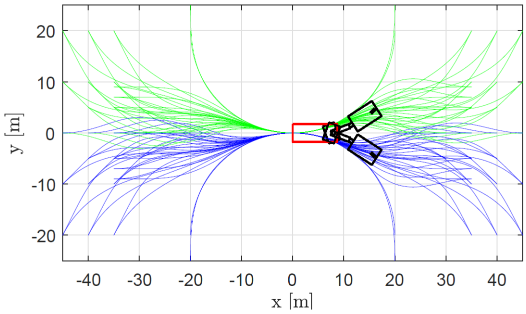
Lattice-based planners have been deployed with great success on several robotic platforms [38, 39, 40, 41, 42]. However, a problem with lattice-based approaches is the exponential complexity in the dimension of the state space which can limit the use for more complicated models.

The optimization problem was solved real-time by *Avocado solver* [43].

Example: The *example of movement lattice* is given in (fig. 5.2). Truck (black rectangle) is towing Trailer (red rectangle). The *state* has only one *reach set impacting variable* - *trailer displacement*. When trailer displacement is 0° (fig. 5.2a) the lattice representation of *Reach Set* is different that in case of small left/right tilt (fig. 5.2b).



(a) Trailer displacement 0°.



(b) 12° (green) −12° (blue).

Figure 5.2: *Movement set primitives* for Lattice-Based Movement Planning. [35].

Benefits: Presented method of *Lattice Search* is de-facto *Reduced Reach Set Approximation* in open space for *Truck-Trailer* system.

The idea of *Movement primitives* is very close to *Movement Automaton* (def. **??**), which can be used as a *control interface* effectively.

The *Constraints* of obstacle set in *Known world* (sec. **??**) are supported to some degree.

Shortcomings: The lattice search approach has the following shortcoming:

1. *Limited system dimension* - given method works in *real time* only if dimension of *system state space* does not exceed 4^{th} rank.

2. *Real-time optimization* - real-time optimization is main cause of *limited system dimension*. If the decision time can be discrete (which movement automaton enforces), then offline optimization can be used.

3. *Continuous space disparity* - the example (fig. 5.2) shows there are member variables of *State Space* which significantly impacts the shape of the lattice (reach set estimation). Space disparity is not a problem in real-time environment. The discretization of *time-domain* raises this as a shortcoming.

4. *Trajectory Tracking* - approach generates *Continuous Domain* reference signal. For *Discrete Domain*, it is necessary to address this issue.

5.5. Software Testing

Idea: *Reuse LSTS toolchain architecture* for DAA testing framework.

LSTS Toolchain: Software architecture used in modern unmanned aerial vehicles must be system independent and scalable. Writing own control software for unmanned aerial vehicle and ground station is unthinkable in the current state of the art. Most notable framework for unmanned aerial vehicle development is the LSTS toolchain from University of Porto [44]. This toolchain is widespread in other universities, and multiple independent applications are based on it [45].

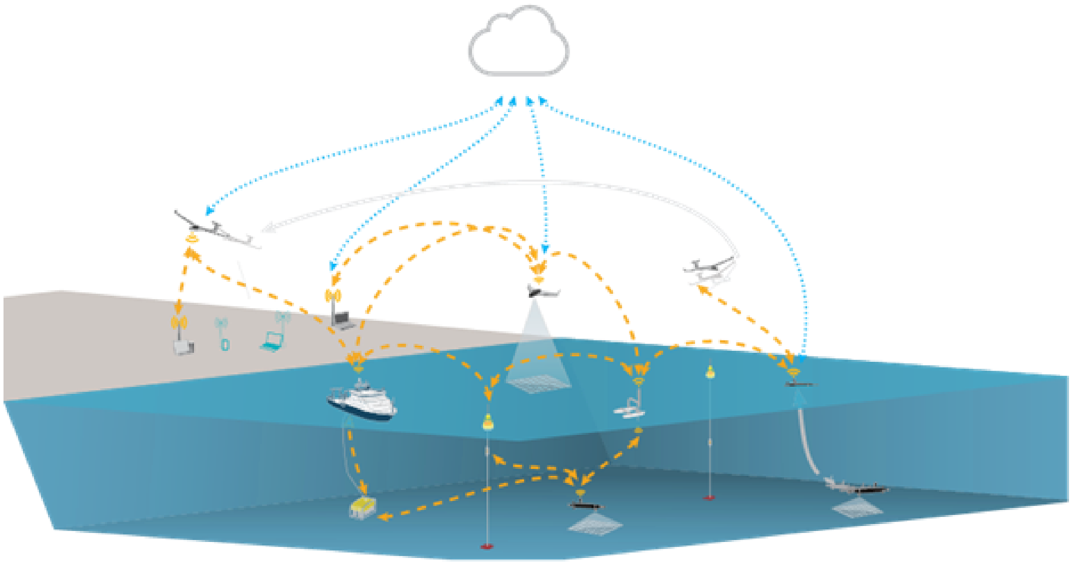


Figure 5.3: Example of LSTS toolchain deployment in a real environment [46]

Example of software architecture implementation is shown in figure 5.3 LSTS HUB (cloud iconography) is collecting all important data from currently executing missions. Data are transferred via REST API (dotted blue lines) to HUB. Commanded vehicles can be unmanned copters, planes, ships, submarines or floating sensors compounds. Each vehicle has installed DUNE which is responsible for vehicle command and ground control station communication. Deployment, a range of command and status messages can vary. The ground station can be implemented on a personal computer (any platform) in NEPTUS environment or mobile platform (Android OS) in ACCU environment. Ground station environments are customizable and open source. The layout of the ground station can be customized to need of current mission via plugins or console configuration. Vehicles and ground stations are communicating via IMC protocol (orange dashed lines). The communication channel is platform independent.

Glued is a Debian-based open source operating system which was initially released in 2010. Over the years it has become fairly widespread and been provided continuous updates. Meanwhile, a notable development community has emerged, where advice and support can be received as more applications are investigated. Some of the merits of the operating system are its reliability and customizability. Operation system is developed for unmanned autonomous vehicles. It is widely used in air, sea and land vehicles. Customization allows the operating system to be tailored to the specific usage. For this purpose, this includes stripping off functionality which is not required, thus releasing resources to be focused on the essential tasks. Glued is the operating system favored by the Beaglebone development community, and thus there exist considerable amounts of helpful documentation for this set-up.

DUNE: Unified Navigational Environment is an onboard software solution for unmanned vehicles. Multiple applications already run on this software. Almost any extension can be added to this to this environment. The software solution provides a means for interacting with the connected components as well as control, navigation, supervision and plan execution. It is both CPU architecture independent and OS independent. It is written in C++ and developed by LSTS: Underwater Systems and Technology Laboratory.

Neptus [46, 47, 48] is a command and control software operated from a ground station. It is designed to operate well together with Dune and was also developed by LSTS. Neptus provides tools for remotely monitoring UAVs and assigning plans and commands in real-time missions, supporting multiple connections dynamically. Furthermore, it provides possibilities for both simulating missions and reviewing previously performed operations. This is presented in a customizable interface equipped with map layers and control panels. It is written in Java and available for both Windows and Linux systems.

The *Inter-Module Communication* [49] (IMC) protocol was developed by LSTS to provide reliable communication between the systems. The protocol is message-oriented, such that messages can be sent and received from a bus which connects independently run threads or systems. Thus it functions as a method of communication between tasks internally in Dune, and can also be passed to and from other vehicles or computers running Dune or Neptus. IMC is platform independent at multiple messages have been already developed and supported by both DUNE and Neptus (around 400 status/command messages).

HUB is communication hub for data dissemination and situation awareness. This module is responsible for complex mission execution when cooperation of multiple pilots/vehicles is required. This module can be imagined as an airport tower (traffic control) center, which is monitoring all flights in the airport (operation site) area.

Movement Automaton Control Marconi used *hybrid automaton* with forced *State Switch* via buffer [50]. The key concept is that *Automata* state switch is forced as *external source command*. Our *Movement Automaton* implementation knows only *forced state switch* like in [2].

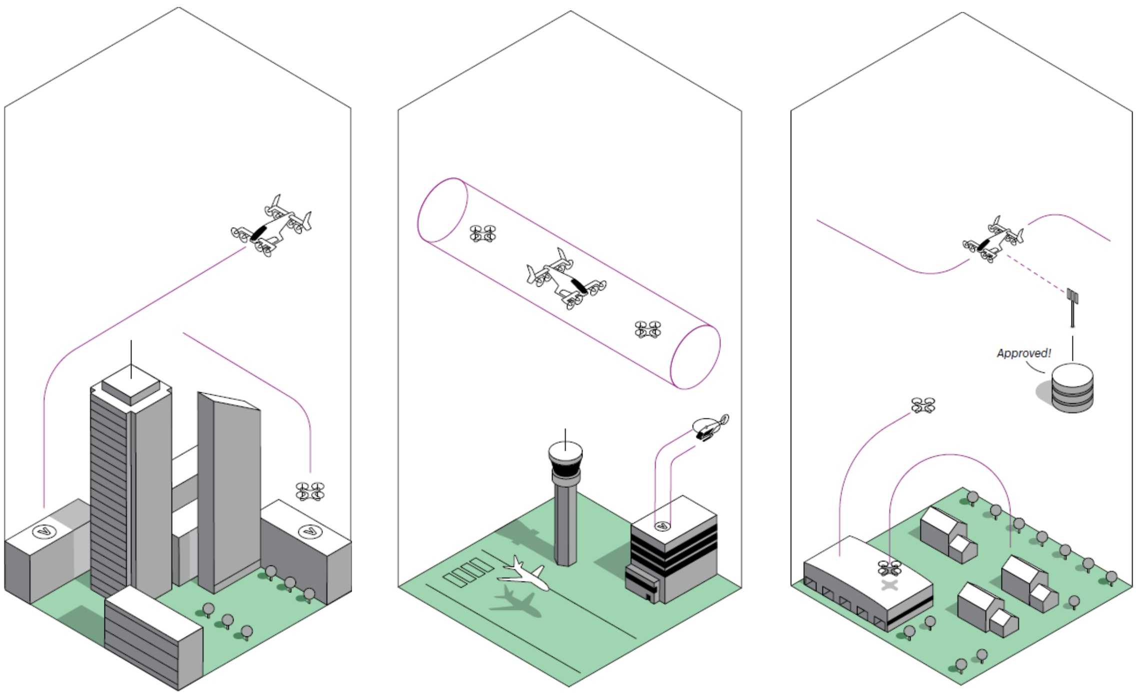
Used concepts: Most of the architecture was re-used in our approach, the concept of *Rule engine* (sec. **??**) was introduced to cover missing *UTM* related functionality. The implementation in Matlab was influenced by Alessandeti works [51, 52]. The other aircraft dynamic and control related concepts were taken from [53].

5.6. UTM Services

Idea: Take the Airbus UTM concept [54] combine it with *EUROCONTROL* concept [55] to obtain a legal framework. *Provide conflict resolution functionality* for *Controlled Airspace*:

1. *Collision Detection* - define minimal required functionality for collision detection.

2. *Collision Resolution* - implement *Rules of the Air* [56].



(a) Free route. (b) Corridors. (c) Fixed route.

Figure 5.4: UTM Operation Modes [54].

UTM Operation Modes: *defined in* [54] are following:

1. *Free route* (fig. 5.4a) is when aircraft can fly any path, so long as their planned path is coordinated with and de-conflicted from the paths of other aircraft by a traffic manager and approved based on calculated risk. Free routing is being introduced worldwide, such as free route airspace. This allows commercial flights to freely plan their route through participating sectors during the cruise. There is less freedom for an aircraft in this situation than in basic flight, since its request may be rejected.

2. *Corridors* (fig. 5.4b) are defined volumes in space, useful for managing airspace in high demand or to manage traffic flow and separation. Coordination is necessary to ensure safety in this airspace. A corridor may take on many different shapes. Aircraft are often guided inside corridors using predetermined routes analogous to approach procedures used worldwide today.

3. *Fixed route* (fig. 5.4c) are used to ensure safety when there is high traffic density or in any location where the structure is required to ensure safe operations. This could include locations such as airports or warehouses. These routes could be constructed or modified

dynamically based on calculated risk. The most restrictive version is a predetermined path, where the only variable is when an aircraft is at a specific point in the path.

Used Concepts: The implementation of our UTM services is focused on *Free route mode* (fig. 5.4a). The *Corridors* and *Fixed routes* are just additional *space/time constraints*.

Bibliography

[1] Emilio Frazzoli. *Robust hybrid control for autonomous vehicle motion planning*. PhD thesis, Massachusetts Institute of Technology, 2001.

[2] Emilio Frazzoli, Munther A Dahleh, and Eric Feron. Trajectory tracking control design for autonomous helicopters using a backstepping algorithm. In *American Control Conference, 2000. Proceedings of the 2000*, volume 6, pages 4102–4107. IEEE, 2000.

[3] George J Pappas, Gerardo Lafferriere, and Shankar Sastry. Hierarchically consistent control systems. *IEEE transactions on automatic control*, 45(6):1144–1160, 2000.

[4] Lester E Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of mathematics*, 79(3):497–516, 1957.

[5] James Reeds and Lawrence Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific journal of mathematics*, 145(2):367–393, 1990.

[6] Philippe Souères and J-D Boissonnat. Optimal trajectories for nonholonomic mobile robots. In *Robot motion planning and control*, pages 93–170. Springer, 1998.

[7] Steven M LaValle. Rapidly-exploring random trees: A new tool for path planning. 1998.

[8] Héctor J Sussmann. Lie brackets, real analyticity and geometric control. *Differential geometric control theory*, 27:1–116, 1983.

[9] 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.

[10] Fredrik Gustafsson. *Statistical sensor fusion*. Studentlitteratur,, 2010.

[11] Subramanian Ramasamy, Roberto Sabatini, and Alessandro Gardi. Avionics sensor fusion for small size unmanned aircraft sense-and-avoid. In *Metrology for Aerospace (MetroAeroSpace), 2014 IEEE*, pages 271–276. IEEE, 2014.

[12] Subramanian Ramasamy, Roberto Sabatini, Alessandro Gardi, and Trevor Kistan. Next generation flight management system for real-time trajectory based operations. *Applied Mechanics and Materials*, 629:344–349, 2014.

[13] Yuriy Chynchenko, Tatyana Shmelova, and Oksana Chynchenko. Remotely piloted aircraft systems operations under uncertainty conditions. *Proceedings of the National Aviation University*, 1(1):18–22, 2016.

[14] T Shmelova, D Bondarev, and Y Znakovska. Modeling of the decision making by uav's operator in emergency situations. In *Methods and Systems of Navigation and Motion Control (MSNMC), 2016 4th International Conference on*, volume 1, pages 31–34. IEEE, 2016.

[15] Volodymyr Kharchenko, Tatyana Shmelova, Yevgeniya Znakovska, Dmitriy Bondarev, and Andriy Stratiy. Modelling of decision making of unmanned aerial vehicle's operator in emergency situations. *Proceedings of the National aviation university*, 1(1):20–28, 2017.

[16] Kwang-Yeon Kim, Jung-Woo Park, and Min-Jea Tahk. Uav collision avoidance using probabilistic method in 3-d. In *Control, Automation and Systems, 2007. ICCAS'07. International Conference on*, pages 826–829. IEEE, 2007.

[17] Rene Vidal, Omid Shakernia, H Jin Kim, David Hyunchul Shim, and Shankar Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE transactions on robotics and automation*, 18(5):662–669, 2002.

[18] Barbara Pfeiffer, Rajan Batta, Kathrin Klamroth, and Rakesh Nagi. Path planning for uavs in the presence of threat zones using probabilistic modeling. *IEEE Trans. Autom. Control*, 43:278–283, 2005.

[19] Mangal Kothari and Ian Postlethwaite. A probabilistically robust path planning algorithm for uavs using rapidly-exploring random trees. *Journal of Intelligent & Robotic Systems*, pages 1–23, 2013.

[20] Lars Blackmore, Hui Li, and Brian Williams. A probabilistic approach to optimal robust path planning with obstacles. In *American Control Conference, 2006*, pages 7–pp. IEEE, 2006.

[21] Steven M LaValle, Michael S Branicky, and Stephen R Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *The International Journal of Robotics Research*, 23(7-8):673–692, 2004.

[22] Ira M Gessel. A probabilistic method for lattice path enumeration. *Journal of statistical planning and inference*, 14(1):49–58, 1986.

[23] Floriana Esposito, Donato Malerba, Giovanni Semeraro, and J Kay. A comparative analysis of methods for pruning decision trees. *IEEE transactions on pattern analysis and machine intelligence*, 19(5):476–491, 1997.

[24] Kaoru Hirota. Concepts of probabilistic sets. *Fuzzy sets and systems*, 5(1):31–46, 1981.

[25] Jacco M Hoekstra, Ronald NHW van Gent, and Rob CJ Ruigrok. Designing for safety: the 'free flight'air traffic management concept. *Reliability Engineering & System Safety*, 75(2):215–232, 2002.

[26] Alessandro Gardi, Roberto Sabatini, and Trevor Kistan. Multi-objective 4d trajectory optimization for integrated avionics and air traffic management systems. *IEEE Transactions on Aerospace and Electronic Systems*, 2018.

[27] Chad Goerzen, Zhaodan Kong, and Bernard Mettler. A survey of motion planning algorithms from the perspective of autonomous uav guidance. *Journal of Intelligent and Robotic Systems*, 57(1-4):65–100, 2010.

[28] Yixiang Lim, Alessandro Gardi, and Roberto Sabatini. Energy efficient 4d trajectories for terminal descent operations. In *International Symposium on Sustainable Aviation*, 2018.

[29] Swati Mishra and Pankaj Bande. Maze solving algorithms for micro mouse. In *Signal Image Technology and Internet Based Systems, 2008. SITIS'08. IEEE International Conference on*, pages 86–93. IEEE, 2008.

[30] Ibrahim Elshamarka and Abu Bakar Sayuti Saman. Design and implementation of a robot for maze-solving using flood-fill algorithm. *International Journal of Computer Applications*, 56(5), 2012.

[31] Quentin Chatelais, Horatiu Vultur, and Emmanouil Kanellis. Maze solving by an autonomous robot. *Aalborg University*, 2014.

[32] Nathan Richards, Manu Sharma, and David Ward. A hybrid a-star automaton approach to on-line path planning with obstacle avoidance. In *AIAA 1st Intelligent Systems Technical Conference*, page 6229, 2004.

[33] Zhuoning Dong, Zongji Chen, Rui Zhou, and Rulin Zhang. A hybrid approach of virtual force and a* search algorithm for uav path re-planning. In *Industrial Electronics and Applications (ICIEA), 2011 6th IEEE Conference on*, pages 1140–1145. IEEE, 2011.

[34] Allison Ryan and J Karl Hedrick. A mode-switching path planner for uav-assisted search and rescue. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 1471–1476. IEEE, 2005.

[35] Oskar Ljungqvist, Niclas Evestedt, Marcello Cirillo, Daniel Axehill, and Olov Holmer. Lattice-based motion planning for a general 2-trailer system. In *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pages 819–824. IEEE, 2017.

[36] Oskar Ljungqvist, Daniel Axehill, and Anders Helmersson. Path following control for a reversing general 2-trailer system. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 2455–2461. IEEE, 2016.

[37] Niclas Evestedt, Oskar Ljungqvist, and Daniel Axehill. Path tracking and stabilization for a reversing general 2-trailer configuration using a cascaded control approach. In *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pages 1156–1161. IEEE, 2016.

[38] Mihail Pivtoraiko, Ross A Knepper, and Alonzo Kelly. Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, 26(3):308–333, 2009.

[39] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. William "red" whittaker. *Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, Dave Ferguson, Autonomous driving in urban environments: Boss and the Urban Challenge, Journal of Field Robotics*, 25(8):425–466, 2008.

[40] Marcello Cirillo. From videogames to autonomous trucks: A new algorithm for lattice-based motion planning. In *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pages 148–153. IEEE, 2017.

[41] Claire J Tomlin, John Lygeros, and S Shankar Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, 2000.

[42] Hong Chen, Carsten W Scherer, and F Allgower. A game theoretic approach to nonlinear robust receding horizon control of constrained systems. In *American Control Conference, 1997. Proceedings of the 1997*, volume 5, pages 3073–3077. IEEE, 1997.

[43] Boris Houska, Hans Joachim Ferreau, and Moritz Diehl. Acado toolkit—an open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

[44] Diego Merani, Alessandro Berni, John Potter, and Ricardo Martins. An underwater convergence layer for disruption tolerant networking. In *Internet Communications (BCFIC Riga), 2011 Baltic Congress on Future*, pages 103–108. IEEE, 2011.

[45] Kanna Rajan, Frédéric Py, and Javier Barreiro. Towards deliberative control in marine robotics. In *Marine Robot Autonomy*, pages 91–175. Springer, 2013.

[46] José Queirós Pinto, Paulo Sousa Dias, Rui Gonçalves, Gil Manuel Gonçalves, João Tasso de Figueiredo Borges Sousa, Fernando Lobo Pereira, et al. Neptus a framework to support a mission life cycle. In *Proceedings of the 7th Conference on Manoeuvring and Control of Marine Craft*, 2006.

[47] Paulo Sousa Dias, Rui MF Gomes, and José Pinto. Mission planning and specification in the neptus framework. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 3220–3225. IEEE, 2006.

[48] Paulo Sousa Dias, Sergio Loureiro Fraga, Rui MF Gomes, Gil M Goncalves, Fernando Lobo Pereira, Jose Pinto, and Joao Borges Sousa. Neptus-a framework to support multiple vehicle operation. In *Oceans 2005-Europe*, volume 2, pages 963–968. IEEE, 2005.

[49] Ricardo Martins, Paulo Sousa Dias, Eduardo RB Marques, José Pinto, Joao B Sousa, and Fernando L Pereira. Imc: A communication protocol for networked vehicles and sensors. In *Oceans 2009-Europe*, pages 1–6. IEEE, 2009.

[50] Lorenzo Marconi, Roberto Naldi, and Luca Gentili. A control framework for robust practical tracking of hybrid automata. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 661–666. IEEE, 2009.

[51] Virtual arena mpc framework for advanced uav-uas simulations. `https://github.com/andreaalessandretti/VirtualArena`. Accessed: 2016-05-30.

[52] Andrea Alessandretti. Notes on trajectory-tracking and path-following controllers for constrained underactuated vehicles using model predictive control.

[53] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. John Wiley & Sons, 2015.

[54] 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.

[55] Andrew Hately. Concept of operations for u-space. Exploratory research call, EUROCONTROL, sep 2018.

[56] AAMI Standard. Recommended practices. *Operation of Aircraft, Annex*, 6, 1986.