Behaviour Modelling of Computer-Generated-Forces in Beyond-Visual-Range Air Combat

Fabian Reinisch*, Michael Strohal† and Peter Stütz‡
Institute of Flight Systems, Universität der Bundeswehr Munich, Werner-Heisenberg-Weg 39, 85577 Neubiberg, Germany

Keywords: Computer Generated Forces, Behaviour Modelling, Beyond-Visual-Range, Air Combat, Pilot Training, Behaviour Tree.

Abstract: Beyond-visual-range (BVR) engagements are getting more and more frequent in modern air combat. One of the key challenges for pilots here is manoeuvre planning, which reflects their decision-making capacity and can determinate success or failure. To ensure pilot training employing virtual BVR air combat simulations yields success, high accuracy levels of the computer-generated forces (CGFs) are essential. To achieve this, it is substantial to not only replicate and simulate the physical properties of the entities to a sufficient degree, but also to provide them with a close-to-human-like behaviour. In this paper, we propose a general concept to tackle these challenges: First, we introduce flight motion dynamic models (aircraft, missiles, chaff) as well as a jammer. Then, we analyse the workflow of a typical beyond-visual-range air combat engagement, separating it into attack, self-defence and decide. Within this context, we introduce Behaviour Trees as a method to model these tasks and explain its benefits. Further plans include the verification and validation of the CGF behaviour within future experimental campaigns that consist of human-controlled opponent aircrafts (pilots) flying against the CGFs. Finally, we provide an outlook to future work in where we intent to employ reinforcement learning for tasks containing many degrees of freedom.

1 INTRODUCTION

In recent developments, beyond-visual-range (BVR) air combat has become the most frequent type of air combat engagement. In order to be able to prepare for these situations, pilots need to have sufficient knowledge and training to react appropriate to the actions and manoeuvres of the adversary. Since this training process is very costly and time-consuming (preparation of aircrafts, maintenance), using training simulations greatly helps overcoming these limitations. However, in order to maximize the benefit for pilots, simulating a detailed physical representation as well as human-like (which includes imperfect/faulty) behaviour of the adversary side is an indispensable prerequisite. Since these CGFs need to be able to handle different air combat situations, designing this type of behaviour is a problem incorporating many degrees of freedom.

Often, Off-The-Shelf products such as STAGE (Presagis, 2016) or VBS (Bohemia Interactive) are prominently used to model behaviour in military simulations. While these can support AI to some degree, (Toubman et al., 2016 - 2016) concluded that these products don't have the ability to model behaviour through adaptive processes and many even still rely on forms of scripting. Aggravating, they outlined a lack of AI methods in these Off-The-Shelf packages, which would be beneficial to explore new air combat strategies and model CGF behaviour.

Today, existing papers mostly focus on dogfights while BVR air combat research is still uncommon. Additionally, most of the rare papers that do research BVR air combat only focus on a very small subset of the whole air combat workflow such as target detection and tracking (V. Chandrakanth et al., 2022) or engagement support (Joao P. A. Dantas et al., 2021), however research incorporating the entire BVR air combat workflow is still a mostly

* https://www.unibw.de/lft/personen/fabian-reinisch-m-sc
† https://www.unibw.de/lft/personen/dr-ing-akdir-michael-strohal
‡ https://www.unibw.de/lft/personen/univ-prof-dr-ing-peter-stuetz

Reinisch, F., Strohal, M. and Stütz, P.
Behaviour Modelling of Computer-Generated-Forces in Beyond-Visual-Range Air Combat.
DOI: 10.5220/0011306600003274
In *Proceedings of the 12th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2022)*, pages 327-335
ISBN: 978-989-758-578-4; ISSN: 2184-2841

unexplored field. Nevertheless, to be able to train pilots for this type of air combat, the full workflow of an BVR air combat scenario needs to be modeled. While some subtasks from the workflow can be implemented using rule-based mechanisms like finite state machines or behavior trees (which are part of our concept as well), it can be challenging to implement tasks containing many degrees of freedom when being limited to these techniques. Here, machine learning methods, and especially reinforcement learning (RL), can be a promising approach to tackle these challenges (Dongyuan Hu et al., 2021; Haiyin Piao et al., 2020). Using these methods, the CGFs are able to train their behavior themselves to learn suitable actions for different situations.

Our goal is to review different AI technologies towards their suitability of generating intelligent CGF behavior. Within this context, we aim to model a BVR air combat workflow with the goal to implement the most promising AI approaches in our CGF simulation environment. During a later stage, one or multiple pilots/subject-matter-experts (SMEs) will pilot the friendly/blue side and fly against the adversary/red side (AI-controlled CGFs) in order to validate the CGFs air combat behavior. Here, aside from validation, special focus will also be placed on the topic of verification, so the accuracy of the CGF dynamic can be ensured.

Multiple approaches have been undertaken to automate CGF behaviour, (Toubman et al., 2016 - 2016) outlines a concept to tackle this challenge by proposing AI methods which could be used to generate all the needed CGF manoeuvres during the whole engagement. However, this approach is less flexible and the selected method might work well for some phases of the engagement but not so well at others. Instead of this, a different concept would be dividing the beyond-visual-range air combat workflow into different tasks. With this approach, aside from keeping overview and structure, this opens up the advantage of being able to process each task using an AI method that suits it well.

Here, the SMEs who are supposed to validate the CGF behaviour in our experimental campaign, specified multiple AI method requirements. These include, but are not limited to: the CGF behaviour should be explainable, reproduceable, changeable and defined. One particular AI method that fulfils all these upper requirements are Behaviour Trees (BTs), which additionally have storable rulesets so a certain behaviour can also be replayed. In (Siqi Yi et al., 2021), it was further demonstrated that using BTs, it's possible to "perform a series of actions to react to adverse situation", concluding that BTs are able to adapt dynamically to changes in the engagement. Because of these advantages, BTs could be a solid foundation for modelling the air combat workflow and some of its subtasks.

For tasks requiring more complex air manoeuvring, as well as for performing threat analysis, machine learning methods, especially reinforcement learning, are an option. There has been significant progress within this domain within the last years as shown in AlfaGo (David Silver et al., 2016), AlfaGo Zero (David Silver et al., 2017), and AlfaStar, making them a promising approach to tackle these complex decision problems.

2 DYNAMIC MODELS

In order to obtain accurate and realistic results, our requirements include validated and verified dynamic models. We intent to include aircrafts, missiles, radar, chaff and jammer. We have chosen MATLAB Simulink (MathWorks) for constructing the models, since it provides a well-known engineering environment and offers built-in verification capabilities (Test Harness). These models are then converted into C++ Code using Autocode and then embedded into our experimental system.
This system is implemented using ROS (Open Robotics, 2020) and communicates with an external VR flight simulator that will be used by the pilots who are flying against the CGFs. This simulator and the CGFs are communicating using DIS. Finally, we are using Tacview (Raia Software Inc.) as a Debriefing Tool.

2.1 Aircraft Model

The fighter jet models (CGFs) are constructed using a modular, generic and dynamic model, which can be fed with physical data describing (instantiate) the respective aircraft.

The aircrafts' state, at a time, is represented using 9 continuous variables: $\{x, y, z, \psi, \theta, \varphi, v_x, v_y, v_z\}$. These represent positions in north (x), east (y) and down (z) as well as its orientation: roll, pitch and yaw and its velocity $\{v_x, v_y, v_z\}$. Additionally, the CGFs are simulated with a radar model which reproduces representative ranges and the limitations of a real radar. This is important, since the AI methods need to make behaviour decisions using the same air picture a real pilot would have access to. For more information regarding flight dynamics, refer to (Zipfel, 2007).

Usually, two or more aircrafts are involved in a BVR air combat scenario and each aircraft has its own set of state variables. When making manoeuvring decisions, the relative geometry between the aircrafts is relevant. Here, we compute the same state variables containing the relative values between two aircrafts $\{x, y, z, \psi, \theta, \varphi, v_x, v_y, v_z\}$. However, in this case, we additionally include the relative azimuth and elevation angles.

The aircrafts (CGFs) can be controlled using flight control manoeuvres, tactical manoeuvres and other actions. Flight control manoeuvres are similar to an auto-pilot, an example would be to hold a certain heading or accelerate towards a certain speed. Tactical manoeuvres are flight instructions with reference to another aircraft, for example flying a pure pursuit. These data of these two command types is specified as a floating-point number (continuous value in a constraint range). Other actions consist of commands that are able to deploy chaff, control the radar/jammer or fire a missile. There are 11 different control commands so far with the possibility of adding more if needed.

When a CGF received a flight control manoeuvre, it doesn't immediately set the aircraft physical state to the given target value, but smooths out its trajectory so that the CGF behaves according to real-world flight dynamics.

2.2 Missile Model

The missiles' and aircrafts' states are similarly composed. Meaning it contains the nine state variables $\{x, y, z, \psi, \theta, \varphi, v_x, v_y, v_z\}$, however, additionally has a variable to indicate whether the missile hits the ground. It also contains a variable which gets triggered if the missile has hit a target (in order to mimic real-world hit probabilities, we apply a PK rate here). Missiles are initialized using the same state as the aircraft state which fires it.

The missile itself can't be controlled using commands after it was launched. However, it contains its own radar model which is able to direct the missile towards a moving target. During this navigation process, proportional navigation is used.

2.3 Chaff Model

Aircrafts are able to deploy chaff during an engagement. Chaff consist of many metal/metallized articles and are used to confuse radar systems. In our setup, their functionality is mainly used to divert missiles from their original target so their trajectory is directed towards the chaff clouds instead of an aircraft. The CGFs deploy chaff clouds at once (with a predefined delay) to ensure the chaff radar signature gets big enough so the missiles' radar beam gets stuck on the chaff cloud.

2.4 Jammer Model

In general, a jammer is used to jam/confuse radars, either targeting another aircraft or a missile. We aim to implement multiple types of jammers that can be used by the CGFs:

- Spot noise jammer: This is used to just emit strong waves
- False targets: This jammer type is used to simulate non-existent aircrafts on the opponents' radar
- Towed decoy: This jammer consists of a separate object which is towed behind the aircraft using a rope. It's used to divert missiles away towards the decoy.

3 BVR AIR COMBAT WORKFLOW

We propose dividing the beyond-visual-range air combat workflow into three big task loops: Attack, Self-Defence and Decide. We define the CGFs being inside the attack loop when an offensive strategy is being executed, on the opposite side, they are traversing the self-defence loop when execution of defensive strategies has priority. Lastly, the decide block is designated to determine in which of the two major loops the CGFs should be situated according to the current air picture as well as deciding if/when the CGFs should terminate the mission.

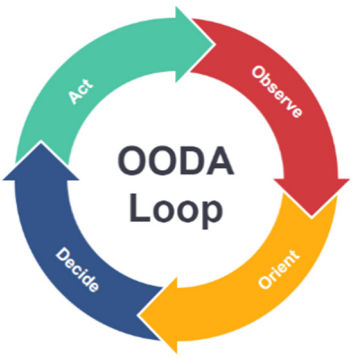

Figure 1: OODA Loop. This loop forms the basics of the workflow.

The majority of the tasks follow the observe–orient–decide–act (OODA) principle (Richards, 2020). It's a tried and tested principle which has proven its advantages when used in decision systems (Henry Leung, 2018) and combat simulations (Aya Fusano et al., 2011). This loop structure ensures that the individual tasks always start with collecting and processing data, so they are able to make informed decisions with up-to-date information.

3.1 Attack

The attack loop is the default for the CGFs, meaning when there is no special reason to go defensive, the CGFs behave according to the attack loop.

We worked out multiple tasks within the attack loop. At each point during the engagement, a CGF is performing one or two tasks of this loop (some tasks can overlap) while at the same time performing checks whether the current task is finished: If yes, move to the next task.

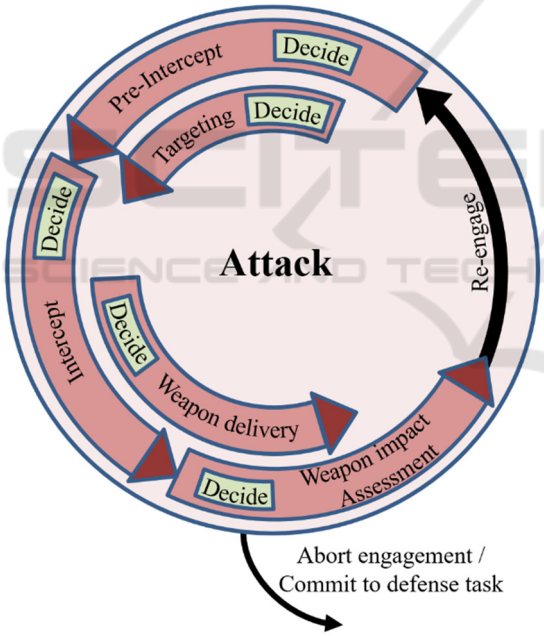

Figure 2: The attack loop workflow as well as its different tasks are visualized here.

Following are the attack tasks:

- Pre-Intercept: At the beginning of the scenario, the CGFs are flying an initial Combat air patrol (CAP) in order to be ready to engage the blue side and shorten response times. This continues, until the distance between the blue side and the CGFs (red side) has fallen below a certain threshold. After this happened, the CGFs are instructed to fly in formation towards the aircrafts from the blue side.
- Picture Targeting: After enemy contact has been reported, the Picture Targeting is performed: This task consists of two subtasks, first, the risk of each blue aircraft with respect to the scenario is rated. This rating process incorporates parameters from the air picture (see Section 3.3) like speed, height, distance, and also the targets probable strategy and intention. Following this, the forces of the blue side are mapped to the CGFs, giving them targets to focus on.
- Intercept: Here, summarized, a flight path is planned and executed with respect to their assigned target aircraft from the blue side. The goal is to obtain a superior position that fulfils the shot criteria with respect to the target. These include optimizing its aspect angle, height (energy) and distance. The own risk level, the predicted target intention/strategy and the targets weapon-engagement-zone (WEZ) also affect the shot criteria. The decision which manoeuvres should be executed to achieve the intercept goals are evaluated constantly, so the CGF is able react to sudden movement changes from the target.
- Weapon delivery: If the shot criteria are fulfilled, a decision is made whether a missile should be fired at all and if yes, the proceeds firing the missile(s). Following this, the CGF needs to support its missile before it becomes active and autonomous. A deeper analysis of this task can be found in (P. Ruther et al. 2022).
- Weapon impact assessment: 10 – 30 seconds after the shot, it can be determined whether the target was hit or not. After this, a decision is made whether the CGF will go back to formation, re-engage (go to pre-intercept / targeting) or go out and terminate the mission (see decide loop).

3.2 Self-defence

When the CGFs are facing threatening situations, they switch to the self-defence loop. As with the attack loop, we divided it into different tasks, however, the current task of each CGF here depends on the danger it currently faces.

The different self-defence tasks fulfil the following functions:

- Track avoidance: In this stage, the CGF is not yet within the weapon engagement zone (WEZ) of the blue side. Therefore, its main goal is, aside from disrupting the opponent (chaff, jammer), to escape the threat and switch back to the attack loop. This can be done by already selecting potential new targets and start planning for an intercept towards it.
- Shot avoidance: If the GCF finds itself within the WEZ of an aircraft from the blue side, but no missile has been fired towards it yet, shot avoidance is executed. Here, the main task consists of executing manoeuvres to get out of the WEZ together with the use of chaff, jammer and optimizing the aspect angle. Special focus is also put on avoiding the no-escape-zone of the blue side.
- Defeat enemy weapon: This is the worst case, avoiding the WEZ of the blue side didn't work and a missile already has been fired towards the CGF. Now, avoiding the missile is the main goal. To achieve this, the CGF is using chaff and jammer (towed decoy) to confuse the missiles radars as well as trying to manoeuvre out of the missile trajectory.

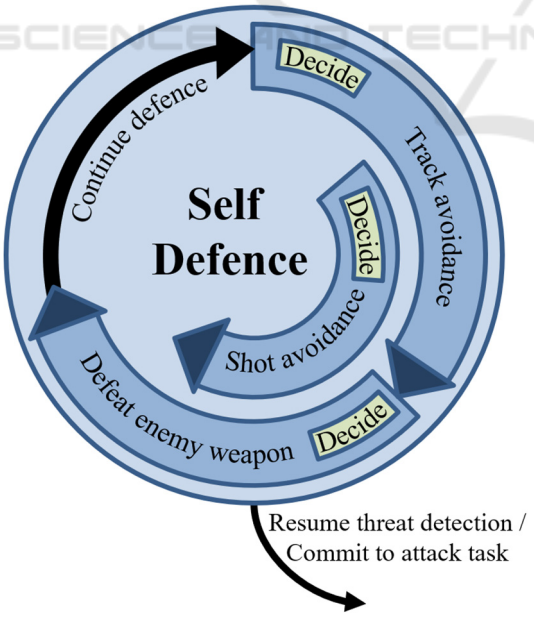

Figure 3: The workflow as well as the different tasks of the self-defence loop are visualized here.

3.3 Decide

The decide loop is not a task sequence like the attack and defence loops. Its job is to decide whether the CGFs should go offensive, behave defensive or terminate their mission. This decision is evaluated constantly during the engagement and depends on multiple properties, which are divided into static/dynamic mission information as well as detection and identification (see below). These properties are then combined to form the air picture which forms the base for making an informed decision.

Static mission information contains fixed properties that are already known in advance prior to the engagement. This includes the own/enemy's, possible physical limitations of the aircraft types or pilot-related limitations (expert level) as well as further fixed properties from used aircrafts/missiles/jammers.

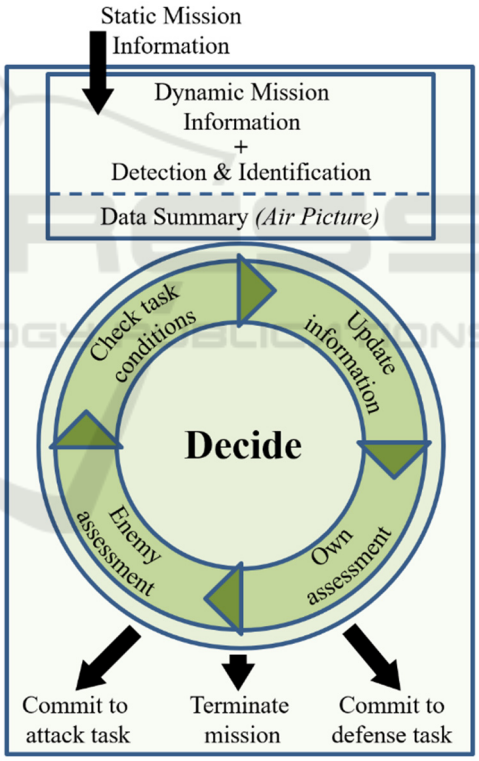

Figure 4: Decide process. It's responsible to choose whether the CGFs should go offensive or defensive.

Dynamic mission information contains data regarding the current state during the engagement. Examples here include the own mission intent and risk levels as well as variable information regarding the current mission like fuel or weapon status.

Detection and identification are the processes of, firstly, collecting data from aircraft sensors (radar, radar-warning-receiver, link) including uncertainties, and in the second step, analysing this data in order to identify opponent aircrafts (aircraft type, etc.). All the collected and processed information then forms the so-called air picture which is used in the following decision process.

The decide process itself starts by collecting and updating all the information from the air picture. It then proceeds to perform the own assessment in which the CGFs own chances are evaluated. This also includes checking whether the CGF is in a threatening situation, e.g. is tracked/attacked. Afterwards, in the enemy assessment, the risk/potential superiority of the enemy, including its intention, etc. are taken into consideration. Finally, the current task conditions are checked. The purpose of this is to decide whether the CGFs can proceed in their current task or if the situation has changed fundamentally and the task needs to be aborted (go from offensive to defensive or the other way). This check varies depending on the task the CGFs currently are dealing with.

4 GENERATING BVR AIR COMBAT CGF BEHAVIOUR

Since we are dividing the BVR air combat workflow into different tasks (Section 3), we have the advantage of being able to compare and evaluate multiple AI methods with respect to CGF behaviour generation for each task separately. While this is a continuous process and will be part of our future research, we additionally need a way to guide the CGFs between the different tasks: Meaning to check whether the current task is finished/needs to be aborted as well as invoke the execution of the following task.

4.1 Behaviour Trees: Overview

Behaviour Trees are a mathematical model used for task execution and decision making. Their origins can be tracked back to the game industry, where their initial purpose was to model Non-player character (NPC) behaviour. Derived from finite state machines (FSM), they were meant to replace FSMs in video games. Nowadays, they are a well-established model that is present in many game frameworks such as Unreal Engine and Pygame. BTs can be artificially created using AI algorithms (Luis Peña et al., 2012; Evgenii Safronov et al., 2020; Matteo Iovino et al., 2021) or manually designed by humans (Francesco Rovida et al., 2017; Enrique Coronado et al., 2018; Chris Paxton et al., 2017). More possible uses span from robot control systems (Özer Özkahraman & Petter Ögren, 2020; Oliver Biggar & Mohammad Zamani, 2020) or human-robot interaction (Dianmu Zhang & Blake Hannaford, 2020) to even machine learning (Bikramjit Banerjee, 2018).

In our case, we decided to employ BTs for guiding the CGFs between the different tasks because they fulfil all the requirements given by the SMEs and provide a solid foundation to model the workflow. Since the SMEs are from a different domain, it is also appreciated by them that BTs are also suitable for non-expert programming (David C. Shepherd et al., 2018; Enrique Coronado et al., 2018). Because of these reasons, we additionally intent to explore the use of BTs for modelling other tasks within BVR air combat as well.

4.2 Behaviour Trees: Application

Since we are implementing our system in ROS2 (Open Robotics, 2020), we have decided to use the BT implementation 'BehaviorTree.CPP' (Davide Faconti, 2018). It supports a seamless integration into ROS and also comes with an editor that allows visualization (see Figure 5) and modification of the trees using a user-friendly GUI interface called 'Groot'. Within our experimental apparatus, we intent to employ a BT calling different AI methods (see Figure 6) depending on which tasks should be executed and then redirect their output back to the CGFs.

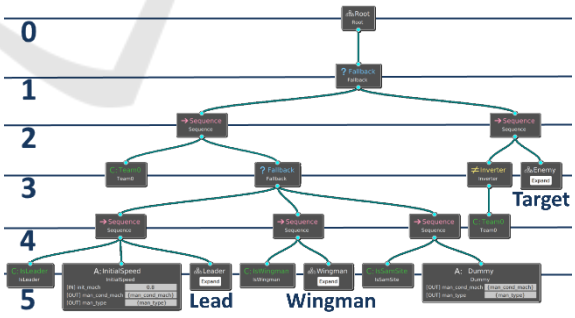

Figure 5: Behaviour Tree containing 6 main levels capable of executing a Baseline Intercept including the three subtrees: Lead, Wingman and Target. Level 0 consists of the root node. Level 1 is responsible to determine to which forces the aircraft belongs to. Level 2 only executes sequential commands. Level 3 performs a check whether the aircraft is Lead/Wingman, or contains its behavioural subtree if it's the Target. Level 4 mainly executes sequence commands again. Finally, at Level 5 reside the subtrees for Lead/Wingman behaviour as well as initialization commands.

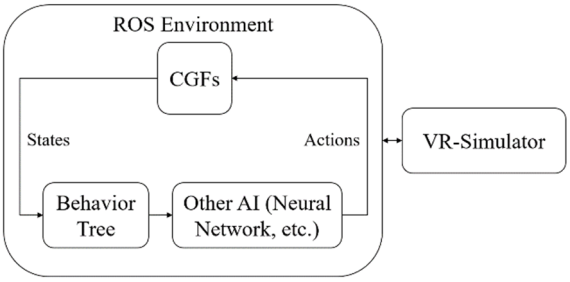

Figure 6: Behaviour Tree workflow in our experimental system. Our implementation (within ROS) consists of the CGFs as well as their behaviour which will be generated using BTs calling different AI methods depending on the current air combat task. ROS is communicating with an external VR-Simulator in which pilots (humans) can fly against our CGFs.

In order to prove the suitability of BTs for our aim, we implemented multiple BVR air combat sample scenario behaviours using BTs. One of these engagements consists of a Baseline Intercept in which two CGFs (lead: red, wingman: brown) are flying towards one target aircraft (blue) in order to identify it (see Figure 7). Using our implementation, we were able to successfully execute the Baseline Intercept using BTs (see Figure 8), therefore showing their ability to dynamically adapt to scenario changes during the execution and generate CGF behaviour accordingly. Finally, in this sample scenario, the advantage of subtrees can be seen as well, since the behaviour for Leader, Wingman and Target (see Figure 5) is modelled using these, so a structured view can be preserved.

5 CONCLUSION AND FURTHER OUTLOOK

In our concept, together with pilots, we elaborated a BVR air combat workflow and divided it into different parts. Combining this with validated and verified dynamic models, we set the foundation to obtain accurate and realistic simulation results. Behaviour Trees have been proven to be a suitable method to model tasks during the engagement. In the future, we intent to further evaluate different AI methods, especially with respect to tasks incorporating more degrees of freedom. Initial work within this field can already be read in (P. Ruther et al., 2022).

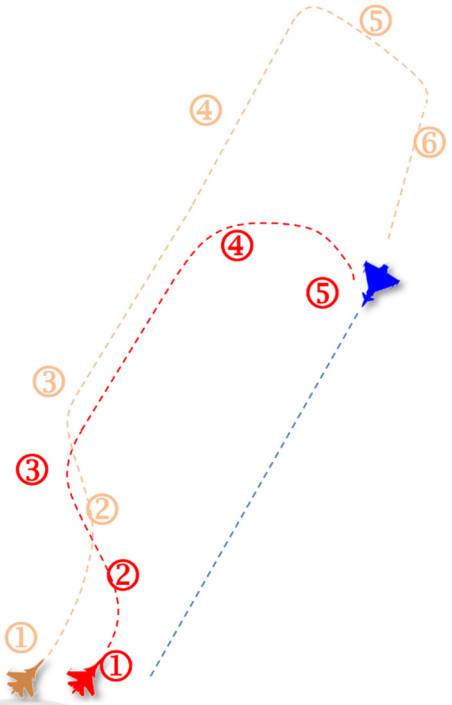

Figure 7: Execution of the sample scenario “baseline intercept”. Lead (red) starts with a pure pursuit towards the blue target (1), then flies different headings (2, 3) to approach the target, followed by continuing its pure pursuit (4) until the final heading (5) is reached. Wingman (brown) starts by flying different headings (1-4) in order to get behind the target, then proceeds with turning (5) and finalises with a pure pursuit (6) towards the target.

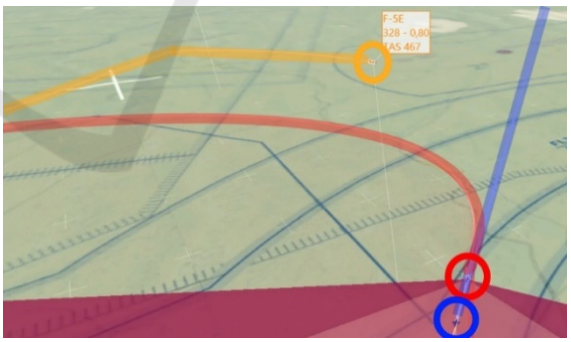

Figure 8: Debriefing tool Tacview (Raia Software Inc.) visualising the execution of a Baseline Intercept (see Figure 7) at stage 5 using BTs for air combat behaviour generation. Visualised are Lead (red), Wingman (brown) and Target (blue) as well as their flown trajectories.

REFERENCES

Aya Fusano, Hiroshi Sato, & Akira Namatame (2011). Multi-Agent Based Combat Simulation from OODA and Network Perspective. In David Al-Dabass,

Alessandra Orsoni, Richard J. Cant, & Ajith Abraham (Eds.), *Proceedings of the 13th UKSim-AMSS International Conference on Computer Modelling and Simulation, Cambridge University, Emmanuel College, Cambridge, UK, 30 March - 1 April 2011* (pp. 249–254): IEEE Computer Society.

Bikramjit Banerjee (2018). Autonomous Acquisition of Behavior Trees for Robot Control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018* (pp. 3460–3467): IEEE.

Bohemia Interactive. *VBS*.

Chris Paxton, Andrew Hundt, Felix Jonathan, Kelleher Guerin, & Gregory D. Hager (2017). CoSTAR: Instructing collaborative robots with behavior trees and vision. In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017* (pp. 564–571): IEEE.

David C. Shepherd, Patrick Francis, David Weintrop, Diana Franklin, Boyang Li, & Afsoon Afzal (2018). [Engineering Paper] An IDE for Easy Programming of Simple Robotics Tasks. In *18th IEEE International Working Conference on Source Code Analysis and Manipulation, SCAM 2018, Madrid, Spain, September 23-24, 2018* (pp. 209–214): IEEE Computer Society.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P. Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, & Demis Hassabis (2016). Mastering the game of Go with deep neural networks and tree search. *Nat*, *529*, 484–489.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P. Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, & Demis Hassabis (2017). Mastering the game of Go without human knowledge. *Nat*, *550*, 354–359.

Davide Faconti, E. (2018). *BehaviorTree.CPP*.

Dianmu Zhang, & Blake Hannaford (2020). IKBT: Solving Symbolic Inverse Kinematics with Behavior Tree (Extended Abstract). In Christian Bessiere (Ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020* (pp. 5145–5148): ijcai.org.

Dongyuan Hu, Rennong Yang, Jialiang Zuo, Ze Zhang, Jun Wu, & Ying Wang (2021). Application of Deep Reinforcement Learning in Maneuver Planning of Beyond-Visual-Range Air Combat. *IEEE Access*, *9*, 32282–32297.

Enrique Coronado, Fulvio Mastrogiovanni, & Gentiane Venture (2018). Development of Intelligent Behaviors for Social Robots via User-Friendly and Modular Programming Tools. In *2018 IEEE Workshop on Advanced Robotics and its Social Impacts, ARSO 2018, Genova, Italy, September 27-29, 2018* (pp. 62–68): IEEE.

Evgenii Safronov, Michele Colledanchise, & Lorenzo Natale (2020). Task Planning with Belief Behavior Trees. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021* (pp. 6870–6877): IEEE.

Francesco Rovida, Bjarne Großmann, & Volker Krüger (2017). Extended behavior trees for quick definition of flexible robotic tasks. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017* (pp. 6793–6800): IEEE.

Haiyin Piao, Zhixiao Sun, Guanglei Meng, Hechang Chen, Bohao Qu, Kuijun Lang, Yang Sun, Shengqi Yang, & Xuanqi Peng (2020). Beyond-Visual-Range Air Combat Tactics Auto-Generation by Reinforcement Learning. In *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020* (pp. 1–8): IEEE.

Henry Leung (2018). An Integrated Decision Support System Based on the Human OODA Loop. In Yingxu Wang, Sam Kwong, Jerome Feldman, Newton Howard, Phillip C.-Y. Sheu, & Bernard Widrow (Eds.), *17th IEEE International Conference on Cognitive Informatics & Cognitive Computing, ICCI*CC 2018, Berkeley, CA, USA, July 16-18, 2018* (p. 4): IEEE Computer Society.

Joao P. A. Dantas, Andre N. Costa, Diego Geraldo, Marcos R. O. A. Máximo, & Takashi Yoneyama (2021). Engagement Decision Support for Beyond Visual Range Air Combat. *CoRR*, *abs/2111.03059*.

Luis Peña, Sascha Ossowski, José Marña Sánchez, & Simon M. Lucas (2012). Learning and evolving combat game controllers. In *2012 IEEE Conference on Computational Intelligence and Games, CIG 2012, Granada, Spain, September 11-14, 2012* (pp. 195–202): IEEE.

MathWorks. *MATLAB*.

Matteo Iovino, Jonathan Styrud, Pietro Falco, & Christian Smith (2021). Learning Behavior Trees with Genetic Programming in Unpredictable Environments. In *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021* (pp. 4591–4597): IEEE.

Oliver Biggar, & Mohammad Zamani (2020). A Framework for Formal Verification of Behavior Trees With Linear Temporal Logic. *IEEE Robotics Autom. Lett.*, *5*, 2341–2348.

Open Robotics (2020). *ROS2: Foxy Fitzroy*.

Özer Özkahraman, & Petter Ögren (2020). Combining Control Barrier Functions and Behavior Trees for Multi-Agent Underwater Coverage Missions. In *59th IEEE Conference on Decision and Control, CDC 2020, Jeju Island, South Korea, December 14-18, 2020* (pp. 5275–5282): IEEE.

P. Ruther, M. Strohal, & P. Stütz (2022). Conceptual approach for optimizing air-to-air missile guidance to enable valid decision-making.

Presagis (2016). *STAGE*.

Raia Software Inc. *Tacview*.

Richards, C. (2020). Boyds OODA Loop. *Necesse*, 142–165.

Siqi Yi, Stewart Worrall, & Eduardo M. Nebot (2021). A Persistent and Context-aware Behavior Tree Framework for Multi Sensor Localization in Autonomous Driving. *CoRR*, *abs/2103.14261*.

Toubman, A., Roessingh, J. J., van Oijen, J., Lovlid, R. A., Hou, M., Meyer, C., Luotsinen, L., Rijken, R., Harris, J., & Turcanik, M. (2016 - 2016). Modeling behavior of Computer Generated Forces with Machine Learning Techniques, the NATO Task Group approach. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 1906–1911): IEEE.

V. Chandrakanth, V. S. N. Murthy, & Sumohana S. Channappayya (2022). UAV-based autonomous detection and tracking of beyond visual range (BVR) non-stationary targets using deep learning. *J. Real Time Image Process.*, *19*, 345–361.

Zipfel, P. H. (2007). *Modeling and simulation of aerospace vehicle dynamics.* (2nd ed.). AIAA education series. Reston, Va.: American Institute of Aeronautics and Astronautics.