Safety Analysis for AACAS

Braden Eichmeier
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
Email: beichmei@andrew.cmu.edu

***Abstract*—Uncooperative aerial traffic is an increased concern throughout controlled airspace largely due to the convenience and availability of consumer drones. This problem is exacerbated due to the commercial potential and expected increased use of autonomous drones for applications such as utilities inspection and package delivery. A recent prototype Autonomous Aerial Collision Avoidance System (AACAS) validated a detect and avoid system for uncooperative obstacles. This paper analyzes three key assumptions in the AACAS avoidance planner and presents modest improvements to adapt to assumption violations. A safety analysis performed in simulation shows the avoidance performance and failure sets within the state space of a simplified drone model. The analysis shows the nominal AACAS controller performs poorly against adversarial obstacles in both a position state space and a velocity state space achieving about a 7% successful avoidance rate in each setting. Introducing adaptive functionality to the safety radius around the agent and a convergence coefficient representing caution drastically improves system performance. Future work for improving the controller involves using a more sophisticated switching function between the go-to-goal behavior and the avoidance behavior.**

I. Introduction

Beyond visual line of sight (BVLOS) operation is a growing interest for several commercial drone applications. However, the Federal Aviation Administration (FAA) prohibits all BVLOS operations without passing an extensive waiver process to guarantee a remotely operated drone can safely perform in a well-defined flight scenario. In well-regulated airspace, these concerns are somewhat alleviated because aircraft transmit their position using a traffic collision avoidance system (TCAS) and such aircraft typically follow established air traffic norms. The expectation of non-cooperative aerial obstacles (whether through lacking communication or decorum) in any aerial environment is increasing as drones become more widely available and used. Certifying BVLOS operation includes the requirements that remote operators or autonomous aircraft must reliably detect and avoid such non-cooperative air traffic.

The Autonomous Aerial Collision Avoidance System (AACAS) was a prototype detect and avoid (DAA) system to demonstrate several technologies for beyond visual line of sight (BVLOS) operation[1]. Validating the AACAS system demonstrated avoidance of a single moving obstacle and multiple stationary obstacles using a DJI M600 Pro drone that flew between two waypoints. A potential field method performed the local planning and obstacle avoidance. The vector field used a simple go-to-goal attraction field for obstacle-free regions. Orbit fields drove the drone around obstacles whenever a predetermined safety radius around the drone was violated. Additional work from the planner accounted for predicting safety zone violation, filtering detection noise, and handling multiple obstacle avoidance. Figure 1 shows a a flow field visualization from early concept development. A summary video of the project may be found here[2].

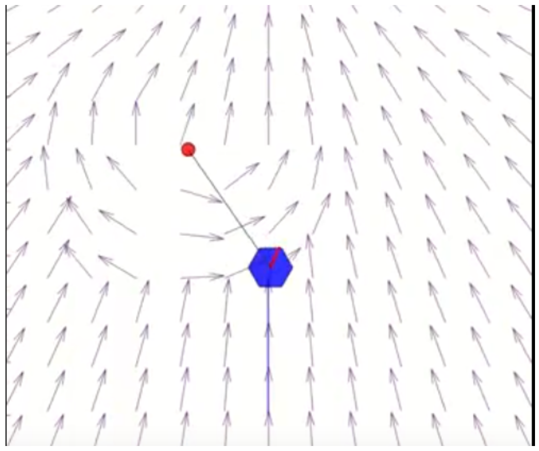
Fig. 1. Hybrid potential field avoidance of a stationary obstacle (red) by the AACAS agent (blue) using attraction and orbit fields.

The vector field controller as implemented was sufficient, but may struggle to succeed outside of structured testing environments. Most of the major flaws in the simple planner revolve around a set of assumptions made for the testing environment and the controller's inability to adapt to broken assumptions. Validating the avoid component of a BVLOS waiver would likely require bounding the environmental assumptions, proving safe operation within assumption bounds, and detailing emergency procedures for unsafe assumption violations.

This paper analyzes a set of assumptions in the AACAS planner through the lens of safe robotics. These assumptions include the maneuverability and intention of incoming obstacles in the environment. The maneuverability assumption assumes a maximum relative velocity between the agent and incoming obstacles. The intention assumption assumes incoming obstacles are at worst mindless, meaning obstacles

[1]https://mrsdprojects.ri.cmu.edu/2020teame/

[2]https://youtu.be/-yjMVBmMjjA

are not aware of the agent. Modifications to the algorithm add an adaptive component to these assumed parameters to improve the performance in broken assumption environments. Finally, preliminary simulation results seek to estimate safety performance of the improved model.

II. Background Information

A. *Potential Field Obstacle Avoidance*

Potential field planning is a well studied approach to low-level robotic control for reactive obstacle avoidance [9]. Originally formulated by Khatib in 1985 [10], these methods create a potential field to push the agent through the state space. To minimize the value of the potential field, desired velocities are computed as the tangent to the potential field at the agent's location. The most basic methods to directly construct a vector field is to create an attractive field towards goal states and repulsive fields away from obstacles [8]. Adouane found more smooth obstacle avoidance through the introduction of orbit fields [1] and later ellipses [2]. Later work in the area developed techniques to handle dynamic obstacles [14][7]. These methods have been applied to various robot models in both simulation and hardware including unicycles [2], UAVs [3], and AUVs [6].

B. *AACAS Assumptions and Limitations*

There are three major assumptions in previous AACAS simulation and validation. Work in this paper seeks to explore the controller's performance when the assumptions are broken and develop strategies to mitigate collision potential.

1) The low-level controllers eliminate the need to consider environment noise.
    - Though the DJI controller effectively mitigated many disturbances. These unmodeled dynamics may be constant, stochastic, or adversarial during validation and applied directly to the state dynamics.
2) The obstacles are less maneuverable than the drone.
    - This was true for all obstacles in the AACAS validation scenarios. However, fixed parameters in the controller are dependant on the relative maneuverability of the two robots. Improving the controller to handle model mismatch will require an adaptive scheme to setting these safety parameters.
3) The obstacles follow up to a 2nd order trajectory and do not react to the robot.
    - Few, if any, dynamic obstacles in the wild will not have any reactionary behaviors. A worst case scenario for the safety planner would be an obstacle that avoids in the same direction, or even pursues the agent.

C. *Control Field Verification*

A common method to validate the safety of a complex control system is the concept of backward reachable sets from an unsafe set within the system state space [13]. In this framework, the analysis attempts to compute all possible states that will inevitably lead to system failure (Fig. 2). Computing this backwards reachable set is both a well-studied and active area of research. General methods to find the backwards reachable set include Hamilton-Jacobi PDE methods [12], Monte-Carlo methods [4], and geometric approaches [11]. Recent work in the area has also attempted to find the Region of Attraction towards an unsafe set using the forward reachable set from the agent's current position [5].

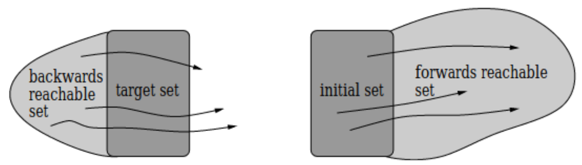

Figure 1: Difference between backwards and forwards reachable sets.

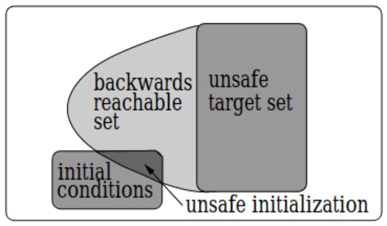

Figure 2: Using the backwards reachable set to verify safety.

Fig. 2. Emergency conditions arise when there is an overlap between system state belief and the backwards reachable set to the unsafe target set.

III. Methods

A. *Kinematic Models*

AACAS uses a hexrotor drone that can be well described with a 3-1/2 dimensional point mass with state and control vectors directly corresponding to the 3D Cartesian position and heading. We simplify the model to a planer dubins car model, which is a unicycle with constant velocity, to simplify the control scheme. Altitude changes are limited to emergency maneuvers only. Equations 1 shows the kinematics for the unicycle model:

$$\dot{x} = \frac{d}{dt}\begin{bmatrix} X \\ Y \\ \theta \end{bmatrix} = \begin{bmatrix} u_v * \cos\theta \\ u_v * \sin\theta \\ u_\omega \end{bmatrix} + D \qquad (1)$$

where D are disturbances from the environment.

Potential field approaches like that used in the AACAS system directly commands the system's velocity. Underlying controllers readily commanded the DJI M600 Pro to a desired velocity. For the purpose of this safety analysis we use a simple state feedback controller to indirectly smooth the control inputs.

$$\dot{u_\omega} = -K_\omega * [\theta - \theta_{des}] \qquad (2)$$

B. Agent Controllers

For this scenario there will be two types of agents within the environment: an aware agent that modifies its behavior based on the other agents, and an adversarial agent that attempts to drive the system into an unsafe state. The ignorant agent will simply follow a pre-determined time parameterized path throughout the simulation episode. These paths will follow up to a second degree polynomial. The environment aware controller will follow a vector field control module that avoids other obstacles in the environment en route to a goal position. The adversarial controller drives its agent towards the other agent. For the purpose of this paper there will only be two agents in each test scenario.

1) Vector Field Control: Vector field control is formulated by computing the instantaneous velocity for a given state relative to other agents in the environment (Eqn. 5). The vector function would be a hybrid piece-wise function conditioned upon the distance to the nearest obstacle. Note: the following state vectors assume only 2D Cartesian coordinates.

$$u_{des} = \begin{cases} GoToGoal(x_a, x_g), & d_g \geq r_{safe} \\ Orbit(x_a, x_o), & otherwise \end{cases} \quad (3)$$

$$GoToGoal(x_a, x_g) = v_{max} * (x_g - x_a) * (1 - e^{-d_g^2}) \quad (4)$$

$$Orbit(x_a, x_o) = \begin{bmatrix} \gamma & -\omega \\ \omega & \gamma \end{bmatrix} .* (x_a - x_o) \quad (5)$$

$$\gamma = k_\gamma * (r_{safe}^2 - d_o^2) \quad (6)$$

where
x_a is the (x,y) position of the agent,
x_g is the (x,y) position of the goal,
x_o is the (x,y) position of the obstacle,
d_g is the distance between the agent and the goal,
d_o is the distance between the agent and the obstacle,
r_{safe} is the desired orbit radius,
γ is the convergence rate to the orbit, and
ω = [-1,1] dictates the direction of orbit

The direction of the orbit rotation is dictated as moving towards the left or right of the nearby obstacle relative to the goal point. Some consideration is also given to account for moving obstacles. The command sent to the low-level controller is simply the orientation vector from the vector field (u_{des}).

In the previous implementation of AACAS the safety radius was set with a constant predetermined parameter. This was a value derived from testing in simulation with objects following straight-line motion. A simple method to make this safety parameter adaptive is to sum the velocities of the agent and obstacle (Eqn. 7).

$$r_{safe} = v_{agent} + v_{ob} \quad (7)$$

However, Eqn. 7 does not account for the maneuverability (safe control bounds) of the system. The original AACAS system flown on the DJI M600 Pro was highly maneuverable, so this consideration was not necessary. In a more general sense, the safety radius to begin evasive maneuvers should consider how quickly the agent may change directions. If $u_{\omega,max}$ is the maximum allowable control input, the safety radius calculation may be modified as follows:

$$r_{safe} = (v_{agent} + v_{ob})/u_{\omega,max}^2 \quad (8)$$

The other fixed parameter in the original system is the coefficient for convergence to the orbit (k_γ). This parameter can be seen as a caution parameter. At 0, the agent orbits the obstacle without converging to the orbit. At high values the desired velocity points directly towards the orbit radius and effectively turns the inside of the orbit radius into an avoidance field. An effective controller would contain the the low caution behavior for non-adversarial obstacles, but avoidance behavior in the worst case scenario. We achieve such behavior by computing the convergence constant using a smoothing function (Eqn. 9).

$$k_{gamma} = 0.001 + 2 * exp(-d_o) \quad (9)$$

2) Adversarial Control: The adversarial controller drives the agent directly towards another agent. This problem can simply be formulated as driving the agent towards a point on a moving trajectory. Adversarial obstacles for this system will follow the same dubins car kinematics as the AACAS agent. The AACAS controller is replaced with an attraction field towards a point ahead of the agent. Placing the adversarial goal directly on the agent results in sub optimal adversarial control where the adversary regularly overshoots the agent. The lead distance in front of the adversarial target is set as half the distance between the two agents (Eqn. 10).

$$x_{g,o} = x_a + d_o * \begin{bmatrix} \cos\theta_a \\ \sin\theta_a \end{bmatrix} \quad (10)$$

C. Verification

Verification of the controller utilizes the framework of an evader-pursuer game. In this game, the control agent (evader) attempts to reach a goal position while avoiding collisions, and the adversarial agent (pursuer) attempts to collide with the control agent. In this game the agents will have perfect knowledge of the other's state and the pursuer acts after the evader. The game ends after a time threshold has been reached, the evader reaches the goal position, or the pursuer reaches the evader. Two state space explorations will use uniform grid sampling to find the backwards reachable set from collisions. Limited analysis will also be used to evaluate the controller's performance with restricted control inputs.

1) Relative Position State Space: The first test is to explore the relative positions between evader and pursuer with equal velocity that will lead to collisions. An evader is initialized at the origin aligned with a goal point along the x-axis. To provide a worst case scenario, the pursuer is positioned with an orientation aligned with the adversarial goal point. A 20 unit by 20 unit grid is explored with each axis discretized into 50 samples. The x-discretization is [0,20] and the y-discretization is [-10,10]. Figure 3 shows a graphical representation of this state space.

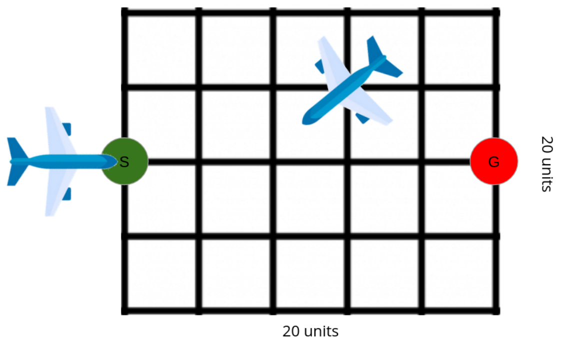

Fig. 3. Position reachability test with the evader (left) navigating from start (green) to goal (red) while evading pursuer (center).

2) Relative Velocity State Space: The second test explores the evader's success at avoiding the pursuer while varying the velocities. Ideally the two tests would be performed in the same state space analysis, but they are separated to facilitate displaying results. In this test, the evader maintains the previous setup to navigate to a goal position. The pursuer is initialized 20 units along the evader's preliminary path of travel, oriented towards the evader. This test shows a worst case head on collision scenario. The state space is explored as a 2D velocity space where one axis represents the pursuer's velocity and the other represents the evader's velocity. Both axes explore a velocity range of 1-11 units/second. Figure 4

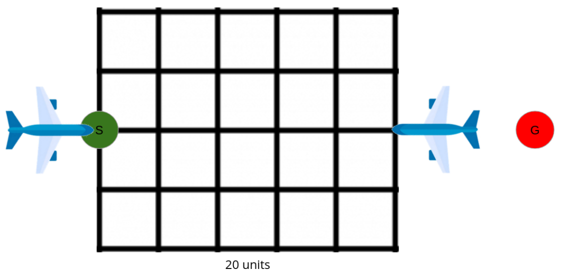

Fig. 4. Velocity reachability test with the evader (left) navigating from start (green) to goal (red) while evading pursuer (right).

IV. Results

A. Position State Space

Despite excellent performance against dynamic and static obstacles, Fig. 5 shows the evader's performance poor performance with an adversarial obstacle. This performance shows a poor tuning of both the safety radius and orbit convergence coefficient. The black regions show the backwards reachable set to the unsafe collision set. White regions show the safe set. The dashed red line on the left shows the collision set (2 unit radius from evader). The orange dot shows the evader's starting position and the blue point shows the goal position.

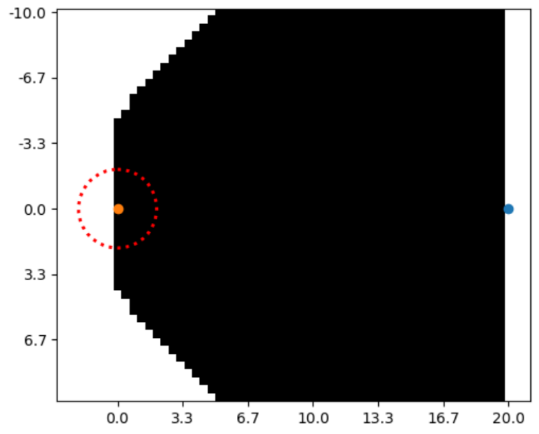

Fig. 5. The baseline AACAS system resulted in a successful avoidance rate of 7.44%.

The next figures (Figs. 6-7) show the performance of the adaptive parameter schemes. Figure 6 shows the convergence to orbit results. The backwards reachable set still covers the majority of the state space, but the safe regions on the upper and lower left sections expand to a more safe solution. The white samples in the black region near the evader are believe to be artifacts from the simulation, or sub-optimal performance from the pursuer. In either case, these samples should be within the backwards reachable set. Implementing the safety radius adaptation scheme shows significant improvement. In this setup the backwards reachable set is reduced to a small area directly in front of the evader (Fig. 7).

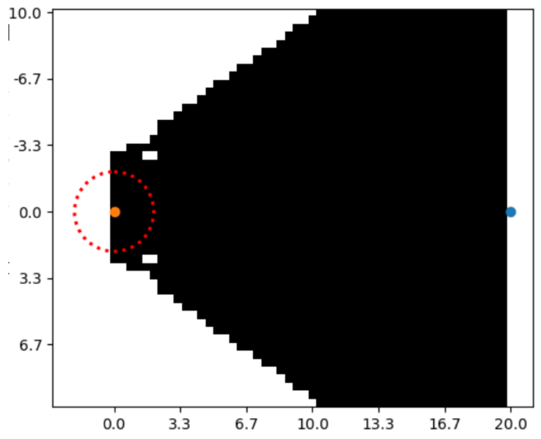

Fig. 6. Implementing the adaptive convergence method improves the evader's success rate to 20.72%. The white dots in the unsafe region are believed to be artifacts in from the simulation.

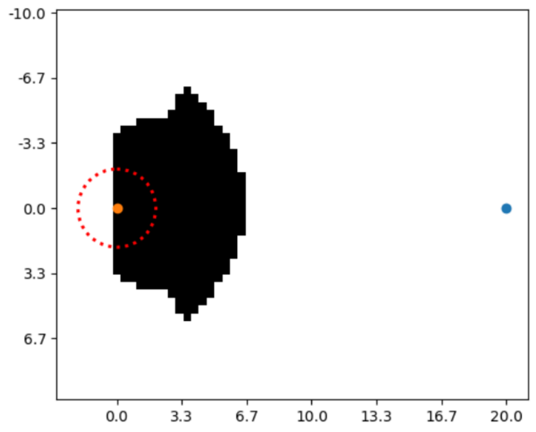

Fig. 7. Implementing the safety radius adaptation method improves the evader's success rate to 85.36%

The final test explored the effectiveness of both adaptations, and showed improved performance over the adaptive safety radius case (Fig. 8). This shows both techniques are needed for optimal performance. An interesting note is that the radius of the unsafe set is smaller, and there are large safe regions that cut into the unsafe set. However, these regions may be artifacts from the simulation or a sub-optimal pursuer model.

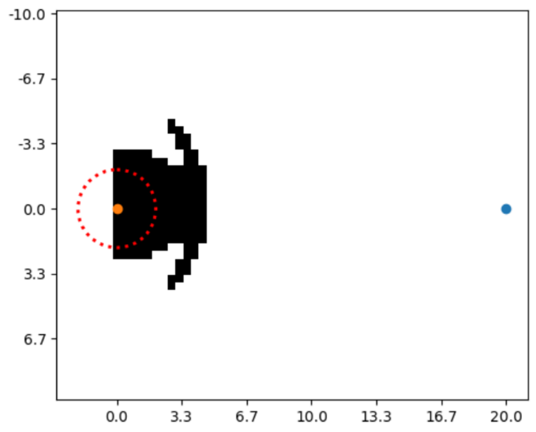

Fig. 8. Implementing the both adaptations improves the evader's success rate to 93.36%

1) Restricted Control Inputs - Position: This section analyzes the adaptive controller's performance with unbounded control commands. This is done by repeating the position state space analysis and restricting the magnitude of the control input for both the evader and the pursuer to a limited value (u_{max}). Figure 9 shows the results of limiting the control values to 0.5, 1, 2, and 3 units/s. At high control values the system reflects the results from the unbounded case. At low control values the unsafe set extends in non-intuitive areas. This is believed to be in part because the pursuer is placed 20 units away from the evader, but the safety radius has expanded beyond this point. As such, the pursuer starts within an unsafe region where the evader can not change direction in time.

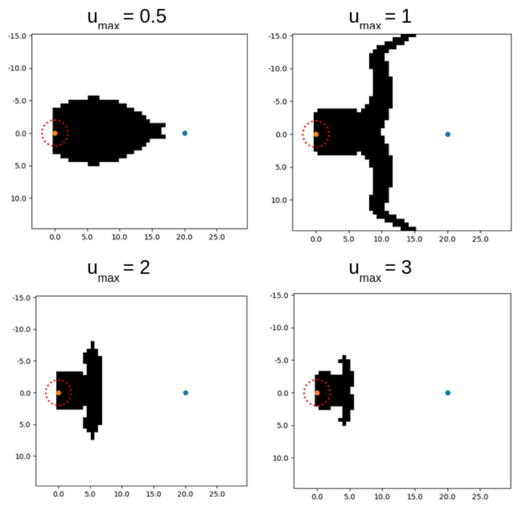

Fig. 9. Limiting the control input results in success ranges of 84-95%.

B. Velocity State Space

The velocity tests follow a similar performance pattern as the position tests. The convergence adaptation (Fig. 10) performs similarly to the original system (Fig. 11) with a slight improvement. These tests show a safe region wherever the evader is faster than the pursuer up to a certain velocity. Applying the safety radius modification greatly expands the safe set to a near diagonal across the state space (Fig. 12), and applying both modifications slightly expands the safe set further by removing a small bump near low velocities (Fig. 13). Table I shows the results for the position and velocity state space tests.

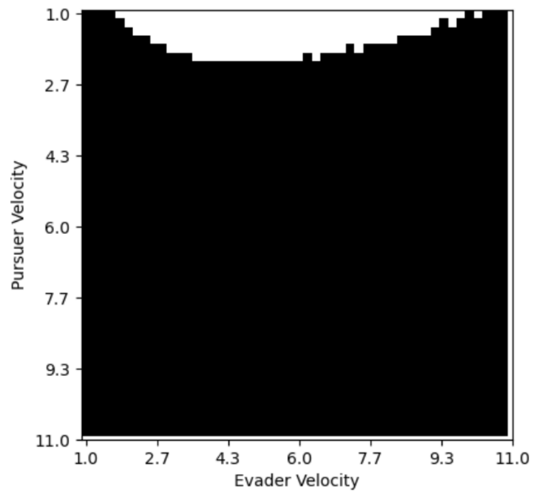

Fig. 10. The baseline AACAS system resulted in a successful avoidance rate of 7.2%.

1) Restricted Control Inputs - Velocity: I repeated the velocity state space tests using the bounded control actions discussed in IV-A1. Unlike the position tests there were

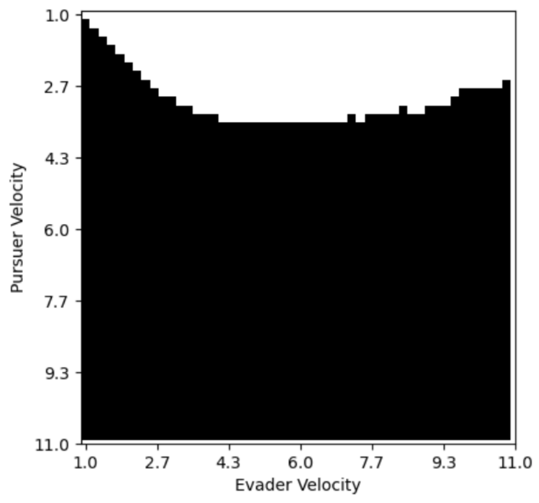

Fig. 11. Implementing the adaptive convergence method improves the evader's success rate to 20.88%

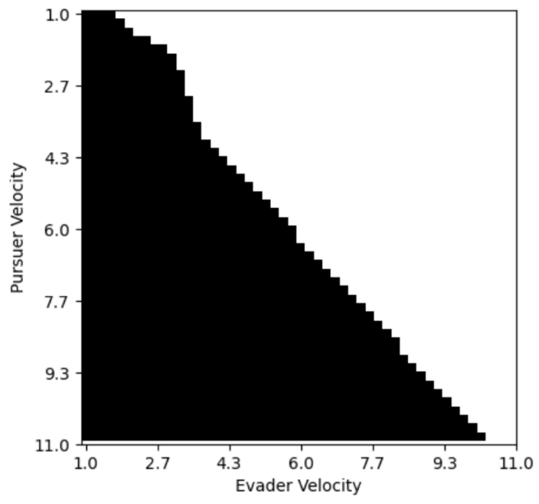

Fig. 12. Implementing the safety radius adaptation method improves the evader's success rate to 50.28%

TABLE I
AACAS AVOIDANCE SUCCESS RATES

	Original	Convergence	Radius	Both
Position	7.44%	20.72%	85.36%	93.36%
Velocity	7.2%	20.88%	50.28%	52.08%

only negligible differences compared to the limited control situation. At very low control bounds the unsafe set expanded to include more states at high evader velocities. However, like the position tests, this was likely due to the pursuer beginning the test inside the evader's avoidance radius. Because the plots do not offer new information, they are omitted for brevity.

V. CONCLUSION

This paper reviews a rudimentary obstacle avoidance control system for a DJI M600 Pro drone. Analysis of the algorithm showed a few key assumptions in the system performance and

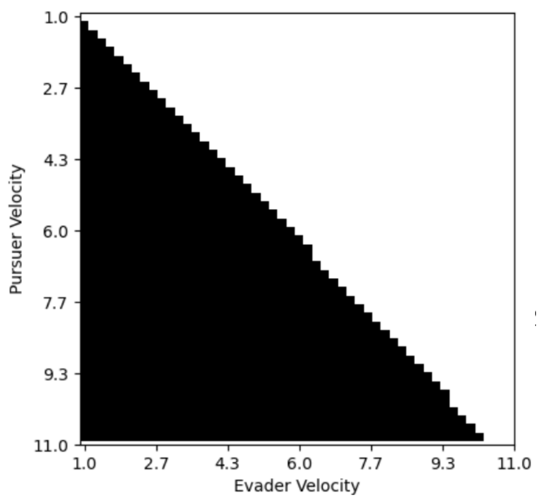

Fig. 13. Implementing the both adaptations improves the evader's success rate to 52.08%

environment that limited the system's safety in deployment. The system was modified to adaptively compute the safety avoidance radius to consider obstacle velocity and safe control bounds and rate of convergence to the safety radius. Testing the original and new controller schemes in an adversarial evader-pursuer game shows significantly better performance from the adaptive system.

Through this analysis I learned one of the major safety problems with any system is making assumptions for the algorithm, robot, and environment. In testing, physical and simulation, the nominal controller behaved well. However adversarial testing revealed severe deficiencies in what was believed to be a satisfactory controller. Even after my presentation, I appreciated the comments highlighting my oversight of bounded control performance that would further limit the system. Note, I modified my presented work in Eqn. 8.

A practice I hope to take away from this paper is to systematically identify and analyze all assumptions leading to hard-coded tuned parameters. Next, a safe analysis would consider how these parameters may be replaced with more robust functions to improve safety and performance.

VI. FUTURE WORK

A severe limitation of the hybrid switching system presented for the AACAS controller is the rudimentary distance based switching function. As previous discussion highlighted, many key assumptions are associated with the size of the safety radius. This simplistic binary switching condition is in contrast to a more sophisticated approach such as wrapping the nominal controller with a safety controller. The outer controller could run a QP-optimization to provide the minimum required safety maneuver. Implementing the switching function with a more elegant switching behavior is the logical next step for improving the AACAS system.

References

[1] L. Adouane. Orbital obstacle avoidance algorithm for reliable and on-line mobile robot navigation. 2009.

[2] Lounis Adouane, Ahmed Benzerrouk, and Philippe Martinet. Mobile robot navigation in cluttered environment using reactive elliptic trajectories. *IFAC Proceedings Volumes*, 44(1):13801–13806, 2011. ISSN 1474-6670. doi: https://doi.org/10.3182/20110828-6-IT-1002.03433. URL https://www.sciencedirect.com/science/article/pii/S1474667016458428. 18th IFAC World Congress.

[3] H. Chen, K. Chang, and C. S. Agate. Uav path planning with tangent-plus-lyapunov vector field guidance and obstacle avoidance. *IEEE Transactions on Aerospace and Electronic Systems*, 49(2):840–856, 2013. doi: 10.1109/TAES.2013.6494384.

[4] A. Devonport and M. Arcak. Data-driven reachable set computation using adaptive gaussian process classification and monte carlo methods. In *2020 American Control Conference (ACC)*, pages 2629–2634, 2020. doi: 10.23919/ACC45564.2020.9147918.

[5] Ahmed El-Guindy, Dongkun Han, and Matthias Althoff. Estimating the region of attraction via forward reachable sets. 05 2017. doi: 10.23919/ACC.2017.7963126.

[6] Xiaojing Fan, Yinjing Guo, Hui Liu, Bowen Wei, and Wenhong Lyu. Improved artificial potential field method applied for auv path planning. *Mathematical Problems in Engineering*, 2020:1–21, 04 2020. doi: 10.1155/2020/6523158.

[7] S.S. Ge and YJ Cui. Dynamic motion planning for mobile robots using potential field method. *Autonomous Robots*, 13:207–222, 11 2002. doi: 10.1023/A:1020564024509.

[8] M. A. Goodrich. Potential fields tutorial. 2002.

[9] M. Hoy, A. Matveev, and A. Savkin. Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey. *Robotica*, 33:463–497, 2015.

[10] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 500–505, 1985. doi: 10.1109/ROBOT.1985.1087247.

[11] Giovanni Miraglia and Loyd Hook. Geometric verification of vector fields for autonomous aircraft navigation. *IEEE Access*, PP:1–1, 02 2021. doi: 10.1109/ACCESS.2021.3062662.

[12] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on Automatic Control*, 50(7):947–957, 2005. doi: 10.1109/TAC.2005.851439.

[13] C. J. Tomlin, I. Mitchell, A. M. Bayen, and M. Oishi. Computational techniques for the verification of hybrid systems. *Proceedings of the IEEE*, 91(7):986–1001, 2003. doi: 10.1109/JPROC.2003.814621.

[14] Qi Zhang, Shi-guang Yue, Quan-jun Yin, and Ya-bing Zha. Dynamic obstacle-avoiding path planning for robots based on modified potential field method. In De-Shuang Huang, Kang-Hyun Jo, Yong-Quan Zhou, and Kyung-sook Han, editors, *Intelligent Computing Theories and Technology*, pages 332–342, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-39482-9.