IEICES

3nd International Exchange and Innovation Conference on Engineering & Sciences
Kyushu University, Fukuoka, Japan
October 19-20, 2017

A new game-based methodology for discovering optimal escape maneuver

M. A. Rushdi[1, 2], A. H. Kassem[1], G. M. El- Bayoumi[1]
[1]Cairo University, [2]Future University in Egypt.
Mostafa.Roshdi@fue.edu.eg

Abstract: *This paper represents a novel game-based methodology for the possible discovery of aircraft optimal escape maneuver against an attacking missile. This discovery could be achieved through a mathematically-correct game of target-attacker where a human player controls the target, trying to evade the missile. The game is distributed to some ordinary persons (typically young kids) who are asked to play the game and to get the highest scores possible. Best escape maneuvers are collected, analyzed, and optimized, to find the optimal human-based escape maneuver. The game is based on 2D point-mass models for target and attacker and a proportional navigation law for missile guidance. The game is developed using Unity, a free cross-platform game engine. The preliminary results obtained suggest the production of enhanced versions of the game with more desirable outcomes, viz., to utilize collective human-brain capability in producing a new guidance law which could possibly compete with existing ones.*

Keywords: Optimal escape maneuver, Proportional navigation guidance law, Unity game.

1. INTRODUCTION

One of the most recent developments in computer technology is online games, which are video games played by humans over (typically versus) a computer or a computer network such as the Internet. The past two decades witnessed the emergence of novel paradigms for games played by humans versus computers. These paradigms include:

- **Competitions** between humans and computers in games of increasing difficulty, including the games of Chess [1], Arimaa [2], Jeapordy! [3], and Go [4]. Typically, the rivalry is between a single human champion (such as Kasparov) against a state-of-the-art computer (such as Deep Blue) limited to a specific domain (such as Chess). The sole purpose of the competition is to decide whether the human or the computer wins more games within the same match.
- The **gamification** phenomenon [5-7], which is the use of game design elements in *non-game* or *learning* contexts, a trend related to human-computer interactions in the form of serious games, pervasive games, alternate reality games, or playful design.
- The paradigm of **games with a purpose** [8-12], which aims to utilize the billions of hours spent (wasted!) by contemporary humans in playing computer games. This paradigm channels game playing into useful work by directing people playing computer games to simultaneously solve large-scale problems without consciously knowing about this and, hence, without losing the element of fun or entertainment. Many large-scale open problems can be solved using collective human brainpower in this unique way. Examples include language translation, monitoring of security cameras, improving Web search, and text summarization. With the paradigm of games with a purpose, hundreds of millions of people can collaborate on the same problem via the Internet,
- **Human Computation** [13], which is the idea of using human processing power to perform tasks that computers cannot *yet* perform or solve problems still *intractable* for computers, usually in an enjoyable manner. Human Computation is also viewed as systems of computers and large numbers of humans that work together in order to solve problems that could not be solved by either computers or humans alone.
- **Crowdsourcing** [14], which is the act of taking a job traditionally performed by a designated agent (an employee) and outsourcing it to a generally large group of people in the form of an open call. Whereas human computation replaces computers with humans, crowdsourcing replaces traditional human workers with members of the human public.

This paper introduces yet another novel paradigm of a game played by humans versus computers [15]. This paradigm resembles that of games with a purpose, but it will neither handle a large-scale open problem nor require a dramatically huge number of human players. The current paradigm might also be viewed as an offshoot of that of human computation, wherein we relax the requirement of dealing with tasks that computers cannot yet perform, and relegate it to handling somewhat sophisticated and complex (but not intractable) tasks. The current paradigm is to solve problems requiring mathematically-elaborate game-theoretic techniques (that already have many automated algorithmic solutions) by requesting a relatively small number of people (~ 50 persons) to play a computer game, constructed to result in a correct solution and, at the same time, is enjoyable. The basic idea is that people will play a game to be entertained, not to solve a problem—no matter how noble and glorious the objective is. The game played constitutes a human-computer competition, but unlike the Kasparov-Deep Blue one, neither the human players are necessarily highly-competent or champions, nor is the computer necessarily of super characteristics. The task encountered herein of designing an online game is much like designing an algorithm. In fact, this game must be proven correct, and its efficiency might be analyzed, so that more efficient versions should always be sought so as to supersede less efficient ones. Instead of using a silicon processor, these "algorithms" run on a processor consisting of the brains of ordinary humans interacting with a computer.

The problem considered herein is an important *pursuit-evasion problem* that involves two agents, the target (aircraft) and the attacker (missile) [15-22]. The attacker missile pursues the target aircraft, and a dynamic or differential game arises in which the target tries to maximize the separation between it and the attacker, while the attacker tries to minimize this separation. We present a game-based methodology to find the optimal escape maneuver for the target (represented by a human player) against the attacking missile (represented by the computer). The missile tracks the target according to the 2D guidance law of proportional navigation. The cost function used maximizes the missile acceleration, time to interception and the miss distance. We construct a mathematically-correct game of target-attacker and let many ordinary people play it individually from the target side, and assign a score to every player that is proportional to the cost function. We note that unlike computer processors, humans require some incentive to become part of a collective computation. Therefore, we provide incentives for the human players to make the most of their perception capabilities, natural intelligence, prudent judgment, problem-solving skills, and readiness for fast response. We find the best escape maneuver by collecting and analyzing data of the escape maneuver of the human players. The game is developed using Unity, a free cross-platform game engine.

The remainder of this paper is organized as follows. In section 2, we digress a little bit to offer a brief characterization of our game and describe the general steps of game design. Section 3 presents the missile model to be supplied to the computer. Section 4 briefly describes the game engine used and explains how our game is built. Section 5 is devoted to data analysis, while Section 6 concludes the paper.

2. ON GAME NATURE AND DESIGN

Since our game is a *skill-and-action* game, it is characterized by *real-time* play, and it would definitely benefit of heavy emphasis on graphics and sound and also of the use of joysticks or paddles rather than just a keyboard. Moreover, the primary skills demanded of the players of our game are psychomotor skills including hand-eye coordination and fast reaction or response time. The game is somewhat related to *combat games*, but it is not a combat game *per se*, since it does not involve a direct mutual violent confrontation between two opponents. The human player, placed in the role of the aircraft, must avoid being captured or hit by the attacking computer-controlled missile, but cannot benefit of the wisdom that retaliatory attack is the best way of evasion or defense. In fact, the game is a *purely defensive* one, in the sense that the player never has the opportunity to attack the enemy missile. The game seems somewhat to be a *race game* since it involves some sort of race between two opponents, but this race is not a straightforward one and it encompasses certain strategic elements.

Game design is an artistic process as well as a technical one, and hence its essence is to manage the integration of these two dissimilar processes [23]. The purpose of the design is to create outlines of three interdependent structures: the I/O structure, the game structure, and the program structure [23]. The

I/O structure constitutes the language for communicating information between the computer and the player. The game structure is a means of figuring out how to distill the goal and topic of the game into a workable system, and identifying some key element or elements around which the game is built. The program structure is the organization of main code, subroutines, interrupts, and data that make up the entire program. All three structures must be created simultaneously, for they must work in harmony. Decisions primarily relating to one structure must be checked for their impacts on the other structures. Once all three structures are made to work in compatibility with one another, it is necessary to evaluate the overall design for the most common design flaws that plague games and to assure that the design satisfies the design goals. The design phase is followed by the programming phase. Programming itself is straining, strenuous and tedious work, requiring attention to detail more than anything else. It might prove to be the hardest of all phases, though its inherent difficulty is somewhat eased by the availability of game engines such as Unity, which is free and simple-to-use beside being able to work on almost any computer and allowing any novice user to download and use its program, and to avoid some of the complex programming techniques through utilizing pre-constructed building blocks.

3. MISSILE MODEL

This Section reviews the equations of 2D proportional navigation [15, 19] and summarizes the information fed into the computer so as to simulate the missile motion. Figure 1 outlines the 2D engagement geometry of the missile and aircraft.

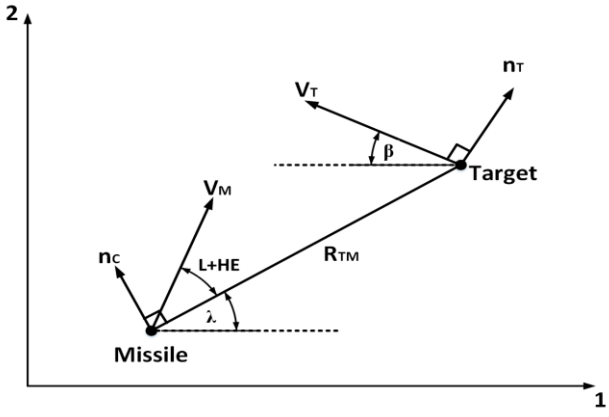

Fig. 1. Two-dimensional missile-target engagement geometry.

The two components of target velocity in 2D are:

$$V_{T1} = -V_T \cos(\beta) \quad (1)$$

$$V_{T2} = V_T \sin(\beta) \quad (2)$$

The relative missile-target separations are:

$$R_{TM1} = R_{T1} - R_{M1} \quad (3)$$

$$R_{TM2} = R_{T2} - R_{M2} \quad (4)$$

From the previous two equations, we get

$$R_{TM} = \sqrt{R_{TM1}^2 + R_{TM2}^2} \quad (5)$$

The line of sight angle is:

$$\lambda = tan^{-1}(\frac{R_{TM2}}{R_{TM2}}) \quad (6)$$

The missile lead angle is:

$$L = sin^{-1}(\frac{V_T \sin(\beta + \lambda)}{V_M}) \quad (7)$$

The angle between the downrange axis and the V_m vector is $\theta = \lambda + L$, while the missile velocity components are:

$$V_{M1} = V_M \cos(\theta + HE) \quad (8)$$

$$V_{M2} = V_M \sin(\theta + HE) \quad (9)$$

The relative velocity components are:

$$V_{TM1} = V_{T1} - V_{M1} \quad (10)$$

$$V_{TM2} = V_{T2} - V_{M2} \quad (11)$$

We define the closing velocity as the negative rate of change of the distance from the missile to the target ($V_c = -\dot{R_{TM}}$). Differentiating equation (5), we obtain

$$\dot{R_{TM}} = \frac{1}{2}(R_{TM1}^2 + R_{TM2}^2)^{\frac{-1}{2}}[2R_{TM1}\dot{R_{TM1}} + 2R_{TM2}\dot{R_{TM2}}]$$

which reduces to

$$V_c = -\dot{R_{TM}} = -\frac{R_{TM1}V_{TM1} + R_{TM2}V_{TM2}}{R_{TM}} \quad (12)$$

To obtain the line of sight rate $\dot{\lambda}$, we differentiate (6) using the rule $\{d(tan^{-1}x) = \frac{dx}{1+x^2}\}$ to obtain

$$\dot{\lambda} = \left[\frac{1}{1+\left(\frac{R_{TM2}}{R_{TM1}}\right)^2}\right]\left(\frac{\dot{R_{TM2}}}{R_{TM1}}\right) \quad (13)$$
$$= \frac{{R_{TM1}}^2}{{R_{TM1}}^2 + {R_{TM2}}^2}\left[\frac{R_{TM1}\dot{R_{TM2}} - R_{TM2}\dot{R_{TM1}}}{R_{TM1}}\right]$$
$$= \frac{R_{TM1}V_{TM1} - R_{TM2}V_{TM1}}{{R_{TM1}}^2}$$

The magnitude of the missile guidance command is

$$n_c = \acute{N}V_c\dot{\lambda} \quad (14)$$

4. GAME DESCRIPTION

This Section briefly introduces the game engine Unity and explains how the first version of our game (henceforth referred to as Evasion#1).

4.1 Introduction to Unity

Unity is a cross-platform game engine (Fig. 2), used to develop video games for PC, consoles, mobile devices and websites. It works with the C# programming language, and can be downloaded free of charge from (https://unity3d.com/), which is the official site of its developer (Unity Technologies).

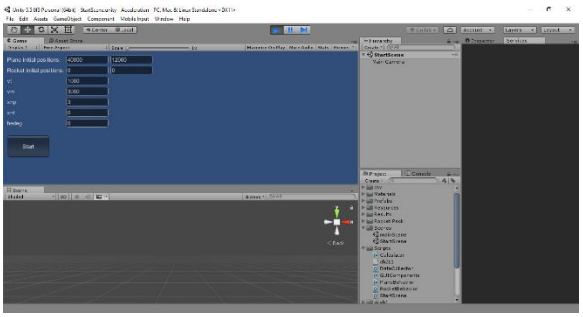

Fig. 2. A typical interface of the Unity game engine.

4.2 Methodology of the game:

Figure 3 illustrates a typical snapshot or computer screen for Evasion#1, showing the two objects that comprise the game, viz., the plane and the missile. The background has stationary shapes (such as clouds) that set a frame of reference allowing the human player to visually estimate the speeds of the plane and missile. The player controls the acceleration of the target (XNT) using the Up/Down arrows on the keyboard to increase/decrease this acceleration. Missile motion is computer-controlled according to proportional navigation. The game has a few adjustable parameters (see Fig. 2), and hence might be adapted for other pursuit-evasion problems.

Fig. 4. A typical snapshot for Evasion#1.

Each game consists of two scenes:

- Start or initial scene.
- Play scene, updated each frame per second.

In the start scene, there are data fields such as position and velocity, which are destroyed at the end of the scene. Therefore, it is necessary to save all needed information in a file called "player prefs". The play scene consists of several objects (currently three). Each object usually has its own script, which must contain two points:

- Start: initialize.
- Update: every frame.

The present three objects of the game are:

a) **Plane:** with a script containing some commands to control the plane by increasing and decreasing the target acceleration by the upward and downward arrows, respectively,

b) **Missile:** which does not contain a script of its own, since the equations controlling its behavior are within the script of the fictitious "Data collector" object,

c) **Object "Data collector":** whose script consists of

- Start
  - Initializing the variables in the equations,
  - Setting the location of the real objects (plane and missile).
- Update
  - Getting plane location,
  - Executing PN equations,
  - Updating rocket location (according to PN equations),
  - Updating information to be printed to Excel,
  - Checking the condition for game termination, and if true, reloading the start scene.

5. PRELIMINARY RESULTS

Our experience with Evasion#1 demonstrates that the game is working correctly. Figures 5-7 illustrate some of our preliminary results, obtained by the player with the best score so far. The target acceleration curve in Fig. 6 exhibits a *Barrel-Roll* behavior, one of the best known practical periodic maneuvers [19]. The interception in Fig. 5 of the target by the missile seems disappointingly quick, but this happens for the extremely unfavorable conditions of a large missile speed to target speed ratio of three, a short initial separation of forty thousand feet, a limited rate of change for the target acceleration, and an unlimited fuel supply for the missile.

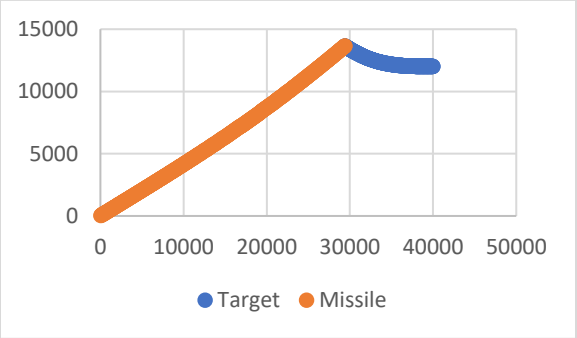

Fig. 5. Target-Attacker 2D trajectory.

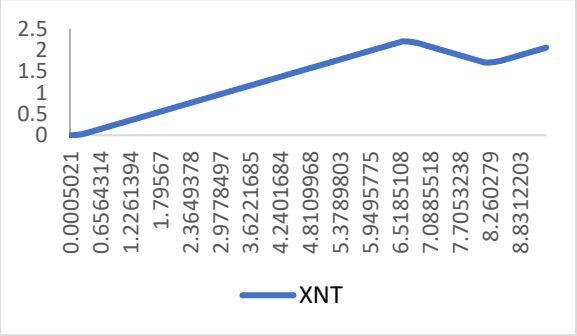

Fig. 6. Optimized target acceleration versus time.

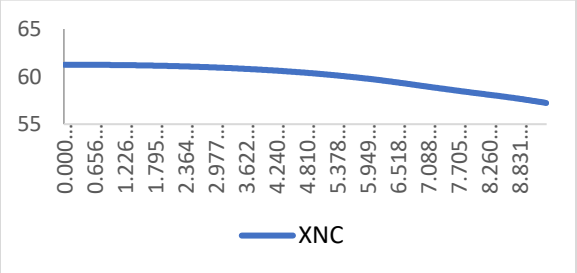

Fig. 7. Missile acceleration versus time.

6. CONCLUSION

A novel paradigm of games played by humans versus computers is introduced and exemplified by a game handling the sophisticated and complex task of solving a classical pursuit-evasion problem that has many automated algorithmic solutions based on mathematically-elaborate game-theoretic techniques. The game requests a small number of ordinary people to play a computer game, which is constructed to result in a correct solution and (as incentive to players) is enjoyable. The preliminary results obtained via our first game version Evasion#1 are reasonable and encouraging. In fact, human players unknowingly obtained a target acceleration of a *Barrel-Roll* shape, one of the best known policies for practical periodic maneuvers. The results suggest the production of enhanced versions of the game with more desirable outcomes. Specifically, in our next version (Evasion#2), the plane will no longer be destined to be intercepted by the missile, since we will impose the restriction that the missile has a limited reservoir of fuel that could be drained as a result of clever target maneuver. The ultimate goal is to test human brain capability to work collectively to produce a new guidance law which could possibly compete in accuracy with the existing ones, while surpassing them in the simplicity of the way it is produced. For future work, the present approach should be evaluated as a sort of Reinforcement Learning (RL) [24] to assess the delay between actions and reward. Typically, a player takes a long series of actions (moving the aircraft up and down) before receiving the reward (the final score). So, it is hard to decide which of the actions were responsible for the eventual payoff.

REFERENCES

[1] Hsu, F. H., *Behind Deep Blue: Building the Computer That Defeated the World Chess Champion*, Princeton University Press, (2002).

[2] Syed, O., and Syed, A., Arimaa - A new game designed to be difficult for computers. *The International Computer Games Association (ICGA) Journal*, 26(2): 138–139, (2003).

[3] Ferrucci, D., Levas, A., Bagchi, S., Gondek, D., and Mueller, E. T., Watson: Beyond jeopardy!, *Artificial Intelligence*, 199: 93-105, (2013).

[4] Gelly, S., and Silver, D., Monte-Carlo tree search and rapid action value estimation in computer Go. *Artificial Intelligence*, 175(11): 1856-1875, (2011).

[5] Seaborn, K., & Fels, D. I. (2015). Gamification in theory and action: A survey. *International Journal of Human-Computer Studies*, 74, 14-31.

[6] Dubbels, B. (2013). Gamification, serious games, ludic simulation, and other contentious categories. *International Journal of Gaming and Computer-Mediated Simulations (IJGCMS)*, 5(2), 1-19.

[7] Boughzala, I., & Michel, H. (2016, January). Introduction to the Serious Games, gamification and innovation minitrack. In *IEEE 49th Hawaii International Conference on System Sciences (HICSS)*, (pp. 817-817).

[8] Von Ahn, L. (2006). Games with a purpose. *IEEE Computer*, 39(6), 92-94.

[9] Von Ahn, L., & Dabbish, L. (2008). Designing games with a purpose. *Communications of the ACM*, 51(8), 58-67.

[10] Siorpaes, K., & Hepp, M. (2008). Games with a purpose for the semantic web. *IEEE Intelligent Systems*, 23(3), 50-60.

[11] Good, B. M., & Su, A. I. (2011). Games with a scientific purpose. *Genome Biology*, 12(12), 135.

[12] Jain, Shaili, & David C. Parkes. (2008). A game-theoretic analysis of games with a purpose. In C. H Papadimitriou, S. Zhang (editors), *Internet and Network Economics*, pp. 342-350. Berlin: Springer. Previously published in Lecture Notes in Computer Science 5385: 342-350.

[13] Quinn, A. J., & Bederson, B. B. (2011, May). Human computation: a survey and taxonomy of a growing field. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1403-1412). ACM.

[14] Brabham, D. C. (2008). Crowdsourcing as a model for problem solving: An introduction and cases. *Convergence*, *14*(1), 75-90.

[15] Rushdi, A. M. (2017). *Analysis and Simulation of the Target-Attacker and the Target-Attacker-Defender Problems*, Unpublished MS Thesis, Department of Aerospace Engineering, Cairo University, Giza, Egypt.

[16] B. L. Stevens & F. L. Lewis (2003). *Aircraft Control and Simulation*, 2nd ed. Hoboken, NJ, USA: John Wiley & Sons, Inc.

[17] Perh, D. (2011). A study into advanced guidance laws using computational methods, Unpublished Master Thesis, Naval Postgraduate School Monterey, CA, USA.

[18] Ghose, D. (2012). A Taxonomy of Guidance Laws, Chapter 6 in *Guidance of Missiles*, Lecture notes for a web course offered by the Indian Institute of Science, Bangalore, India through the National Program on Technology Enhanced Learning (NPTEL), Accessed on July, 7, 2017. http://nptel.ac.in/courses/101108054/module5/lecture11.pdf.

[19] Zarchan, P. (2012). *Tactical and Strategic Missile Guidance. Sixth Edition*. Vol. 239: Progress in Astronautics and Aeronautics, American Institute of Aeronautics and Astronautics, Reston, VA, USA.

[20] Eklund, J. M., Sprinkle, J., & Sastry, S. S. (2012). Switched and symmetric pursuit/evasion games using online model predictive control with application to autonomous aircraft. *IEEE Transactions on Control Systems Technology*, 20(3), 604-620.

[21] Hsueh, M. H., Wang, T. K., & Fu, L. C. (2014). Integrated game based guidance with nonlinear autopilot design for maneuvering target interception. *Asian Journal of Control*, 16(2), 431-440.

[22] Sun, C., He, F., Ma, L., Wang, Y., & Yao, Y. (2014, June). A guidance control law design based on evolutionary game. In *IEEE 2014 11th World Congress on Intelligent Control and Automation (WCICA)*, (pp. 4939-4944).

[23] Crawford, C. (1982). *Art of Computer Game Design*. Berkeley, CA: Osborne/McGraw Hill. Electronic version made by Washington State University, Vancouver, Washington, USA, Available at http://www.vancouver.wsu.edu/fac/peabody/game-book/Coverpage.html.

[24] Liu, J., Liu, S., Wu, H., & Zhang, Y. (2009). A pursuit-evasion algorithm based on hierarchical reinforcement learning. In International IEEE Conference on Measuring Technology and Mechatronics Automation, 2009. ICMTMA'09. (Vol. 2, pp. 482-486).