A Study on Teamwork in a Dynamic Task

Cvetomir M. Dimov (cdimov@andrew.cmu.edu)
John R. Anderson (ja@cmu.edu)
Shawn A. Betts (sabetts@andrew.cmu.edu)
Dan Bothell (db30@andrew.cmu.edu)
Department of Psychology, Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA 15213, USA

Abstract

Skill acquisition experiments have rarely focused on collaborative tasks. Here we attempt to fill this gap with a study on teamwork in a dynamic task. The task - Coop Space Fortress - is computer game, in which subjects fly spaceships to destroy a space fortress. This task presents two challenges: learning how to fly a spaceship in a frictionless environment and developing a strategy on how to coordinate. When learning to play this computer game, subjects not only master the game controls but also typically settle on team roles to more efficiently achieve their goal, despite not being allowed to communicate. The data from this study will pave the way to an ACT-R model of teamwork in a dynamic task.

Keywords: skill acquisition, dynamic task, teamwork, Space Fortress, ACT-R

Introduction

From stumbling into our first steps, to learning a foreign language in middle school and our first mathematical analysis class at university, our lives are replete with various tasks that we master to different extents. It is astounding how skilled we can become after a sufficient amount of practice: the tightrope walker was once a toddler falling after a couple of steps; Shakespeare was once mumbling incomprehensible words and even Euler – the most prolific mathematician ever – was once unable to count.

This gradual shift of the unskilled becoming fully proficient has been characterized as proceeding in distinct phases. Specifically, Fitts (1964, Fitts & Posner, 1967) described motor skill as progressing through three phases: a Cognitive Phase, an Associative Phase, and an Autonomous Phase. Anderson (1982) also adopted the understanding that skills go through three phases and applied this to cognitive skills, whereby he modeled the successive periods of skills acquisition in the cognitive architecture ACT*. Others too have accommodated the idea that skill acquisition is a 3-phase process (e.g., Ackerman, 1988; Kim, Ritter, & Koubek, 2013; Rosenbaum, Carlson, & Gilmore, 2001).

Similarly to ACT*, in ACT-R (Anderson, 2007) – the current version of the architecture – skill transitions from a slower and more deliberative stage to a faster and more automatic stage. This architecture has been applied to model skill acquisition in a variety of tasks, such as solving linear equations (Anderson, 2005), a complex aviation task (Taatgen et al., 2008) and past tense learning (Taatgen & Anderson, 2002). Further support of ACT-R's characterization has been provided in neuroimaging studies, which uncovered qualitative changes in the recorded neural patterns as subjects become more proficient (i.e., in solving pyramid problems; Tenison, Fincham, & Anderson, 2016). Finally, a modification of the architecture was used to model an amount of transfer between skill acquisition tasks not completely accounted for by ACT-R (Taatgen, 2013).

Cognitive Skill Acquisition

ACT-R does not adhere to a 3-phase view of skill acquisition. Instead, this architecture models each subcomponent of the skill as transitioning from a declarative to a procedural endpoint. At one extreme, the cognitive system only has declarative knowledge about a certain task domain. This knowledge is typically stored in terms of operators in declarative memory, which are composed of three pieces of information: the state in which they apply, the action that should be taken, and the state that results after that action is taken. Operators are the building blocks of a subject's mental model of the task and they are typically acquired when reading the task instructions. When an unskilled subject faces a task, operators are retrieved to determine what action should be taken next.

These operators are gradually converted to procedural knowledge through a process called *production compilation* (Taatgen & Anderson, 2002). When an operator is compiled into a production, its actions are directly performed by that production without the need to retrieve the operator. The result is, first, a faster execution of that action as the time cost of retrieval is no longer incurred and, second, retrieval processes are no longer occupied and can be used for other purposes. Moreover, it is possible for two subsequent actions to be complied into a single action if there is no conflict of cognitive resources. The relative rate at which operators are compiled is a function of how often they are evoked, meaning that different subcomponents of skill can be proceduralized to different extents at certain time points.

Dynamic Tasks

The majority of skill acquisition tasks modeled with ACT-R follow a linear perception-cognition-action pattern. Yet, real-world tasks are complex and dynamic, meaning that they involve the coordination of cognitive, perceptual and motor activities in an ever-changing, yet predictable world. To investigate the applicability of ACT-R's approach to dynamic tasks, learning in the arcade game Space Fortress was addressed (Anderson et al., 2019). Space Fortress was selected because it is simple enough to be suitable for an experiment, challenging at first and still learnable within a single experimental session. In addition to its dynamic nature, Space Fortress differs from the majority of skill acquisition tasks modeled with ACT-R in that it requires learning to tune skill to features in the environment so that actions are successful. To this end, ACT-R was extended with a new module – the Controller (Anderson et al., 2019). The model of this dynamic task underwent the same process of skill acquisition that other models did by gradually compiling operators into productions. However, the increased complexity of this task relative to others meant that the model spent much more time compiling operators than simpler models do. Moreover, while operators in declarative memory were being proceduralized, the Controller module was tuning actions to relevant environmental features.

Teamwork

One aspect of skill acquisition that has been rarely researched in the lab is how people learn to execute a novel task while working as a team. In a team, individuals' tasks become interdependent and their goals shared (Dyer, 1984). To achieve high performance, each team member needs to successfully manage the tasks that are independent of the other team members (i.e., *taskwork*) and the tasks that are intertwined with the others (i.e., *teamwork*; Salas, Cooke, & Rosen, 2008). Both taskwork and teamwork depend on the processes of encoding, storage and retrieval of information, while two additional factors are key to teamwork. Specifically, *shared cognition* (i.e., shared mental models and situation awareness; Salas & Fiore, 2004) and communication facilitate coordination and cooperation between team members. To investigate how people learn to work in a team in a dynamic task, we created a new cooperative computer game, Coop Space Fortress.

Coop Space Fortress

Space Fortress has a history in the study of skill acquisition dating back to the end of the 80's (Donchin, 1989; Frederiksen & White, 1989; Gopher et al., 1989). The goal of the game is to accumulate as many points as possible, which can be achieved by destroying a fortress located in the center of the screen while avoiding crashing into a rectangle, which defines the playing field. We relied on the Pygame implementation of Space Fortress (Destefano & Gray, 2008) to create a cooperative version of the game – Coop Space Fortress.

In Coop Space Fortress, two players control two ships (see Figure 1). Their goal is to destroy a fortress in the center of the screen. However, the fortress has an impenetrable shield around it (the small hexagon), which is only partially disabled when the fortress shoots a missile. When this happens, the back of the fortress is no longer shielded and the fortress can be destroyed. Consequently, for the team to destroy the fortress, one ship needs act as a bait: it needs to enter the big hexagon, which triggers the fortress to aim and shoot at that ship if the ship moves sufficiently slowly. While the fortress is shooting and its back is exposed, the other ship needs to navigate behind the fortress, aim at it and destroy it (see Figure 2). To keep things simpler, we did not allow players to communicate. Thus, players needed to figure out their roles based solely on the common instructions that they received.

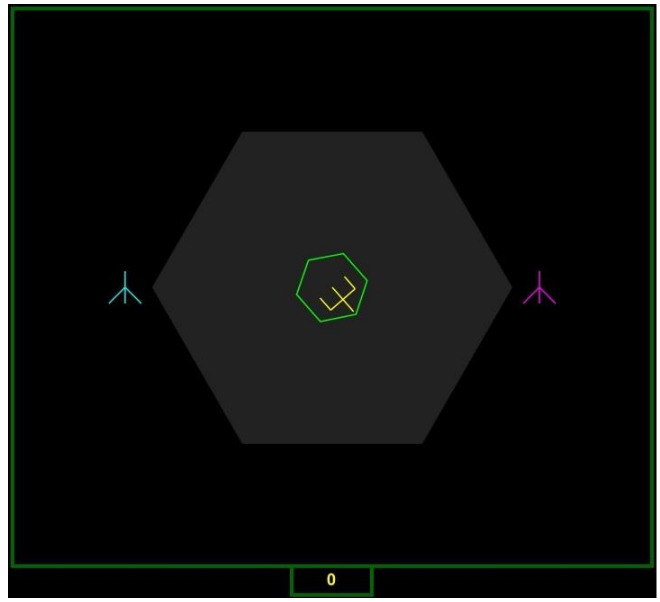

Figure 1: Start of the game. Both players are outside of the hexagon and the fortress has no target. The players should enter the hexagon and try to destroy the fortress.

When the fortress is destroyed, the score is incremented by 100 points. The ships need to then exit the hexagon. When both ships are outside of it, the fortress respawns and the ships can again attempt to destroy it. When outside the hexagon, the ships should avoid hitting the outer border (big square), because they would explode and reduce their common score by 100 points. In addition to penalizing deaths, reckless shooting is also penalized by 10 points for each missile that does not hit the fortress.

Navigation in Coop Space Fortress relies on three actions: rotating clockwise (key "D"), rotating counterclockwise ("A") and thrusting ("W"), while shooting is achieved with the spacebar. Despite having only 4 actions overall, learning to play Coop Space Fortress is a challenging task. A major difficulty is that frictionless space is counterintuitive to operate in. First, the ship's orientation is independent of its direction of flight. Moreover, the ship does not slow down on its own and no breaks are available. Instead, to slow down one needs to turn in a direction opposite the flight direction and thrust. Similarly, moving in a desired direction requires thrusting in a direction, whose vector sum with the flight velocity results in the desired flight path. Another challenge for players is learning how key press durations map to acceleration or rotation rate.

In addition to learning how to control the ship, Coop Space Fortress poses the additional challenge of coordinating with a teammate, because unless each player does their task, no player will earn any points. For example, the player that acts as a bait needs to stay inside the hexagon and fly at a slow speed while the shooter is aiming and shooting. Note that if the bait accidentally exits the hexagon or is shot down by the fortress, the shooter becomes the bait and both players need to reset their current goals. Similarly, if the shooter does not succeed in commanding the ship with enough proficiency to destroy the fortress, the team will perform poorly. Finally, both players need to exit the hexagon for the fortress to respawn. As a consequence, the final performance in the game is an interaction between the skills of each player: If even one player struggled to performs his/her task, the common score would remain low. On the other hand, if each player performed at a reasonable level, the common score would increase.

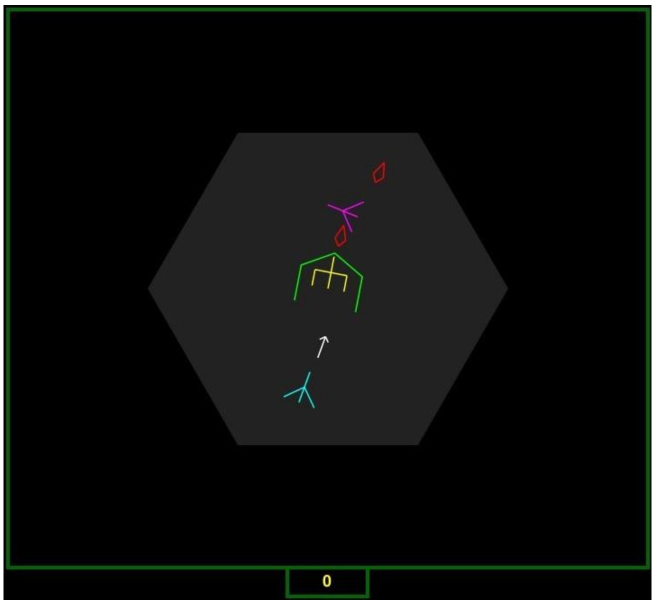

Figure 2: Players coordinating: one player acts as a bait, while the other is shooting at the fortress. The fortress, having shot at the first player, has its back exposed. Once the fortress is destroyed, the two players should exit the big hexagon so that the fortress respawns.

Methods

Participants

Thirty subjects (13 males, mean age: 22.4 years, min age: 18, max age: 35) from the Pittsburgh area, mostly students from Carnegie Mellon University and the University of Pittsburgh, participated for money, which included a base payment ($15) and a bonus payment (Mean: $2.03, Min: $0.10, Max: $11.85) based on their performance. Pairs of participants were formed either randomly, restricted by participants' availability, or by asking participants to bring another participant to play with. Informed consent approved by the Carnegie Mellon University Institutional Review Board (IRB) was obtained from each participant. The data of the first pair of subjects was not included in the analysis as it was not completely recorded. Only 4 of the 28 subjects reported having played a similar game (2 reported Asteroids, while Snake and Minecraft were each considered similar by a single subject each) in the post-experimental questionnaire.

Procedure

The experiment consisted of 4 tasks: (1) a demographics questionnaire, (2) game instructions, (3) 20 3-minute-long rounds of playing Coop Space Fortress, and (4) a feedback questionnaire. A task needed to be completed by both participants before the subsequent one could be started. Participants were given a 1-minute break after 10 games. The overall experiment took between 1h15min and 1h20min of participants' time.

Demographics questionnaire. This questionnaire consisted of general demographics questions and of game-related questions. The general demographics questions inquired about the subject's sex, age, ethnicity, and field of study. The game-related questions requested information about the subject's video game experience, such as whether they ever played or currently play video games, the frequency of play, the platform they played on and the preferred genre of video games.

Post-experimental questionnaire. The post experimental questionnaire elicited information about a subject's experience during the experiment. It inquired what difficulties subjects faced during gameplay, what strategies they attempted and what strategy they finally settled on.

Results

All pairs of participants exhibited learning in the course of the 20 games of Coop Space Fortress. Figure 3 shows the average, minimum and maximum points obtained by subject pairs over the course of the 20 games. On average, teams monotonically increased their performance as the experiment progressed. Yet, there was a substantial variability in the amount of points achieved in a game.

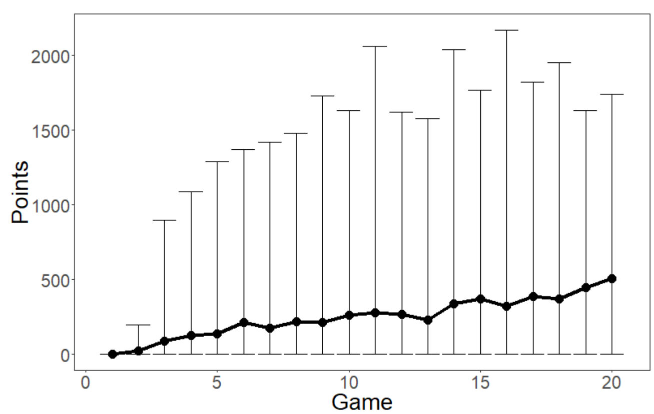

Figure 3: Game score progression over 20 3-minute games. Error bars represent minimum and maximum points

achieved in a game. The mean score increases steadily, but the variability is large.

A major reason for the large variability in score is the between-subject variability (Figure 4), which likely reflects prior experience with video games. Note that total points result from an interaction of the ability of both players: If one player is of low skill, the pair would not reach a high total score no matter how skilled the second player. The skewed distribution of average team score is likely a consequence of this interaction. A second major contributor to the large variability in score is the game-to-game variability within pairs of subjects (error bars in Figure 4).

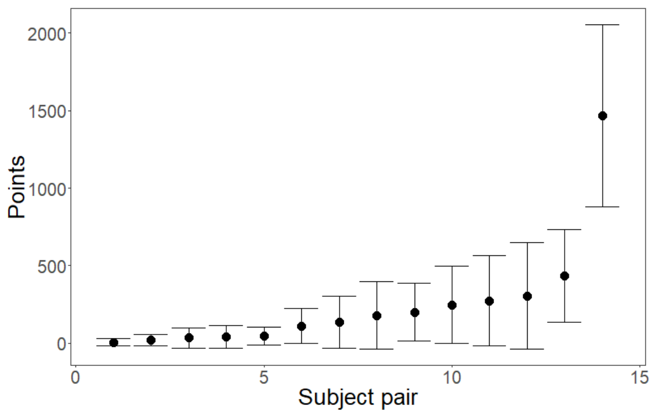

Figure 4: Average score per game for each of the 14 pairs of subjects. Error bars plot standard deviations. There is a large variability in skill between pairs of subjects.

Points are determined to a large extent by the number of fortress kills and number of player deaths. Not surprisingly, as the game progresses, players become better at destroying the fortress and less likely to die (see Figure 5).

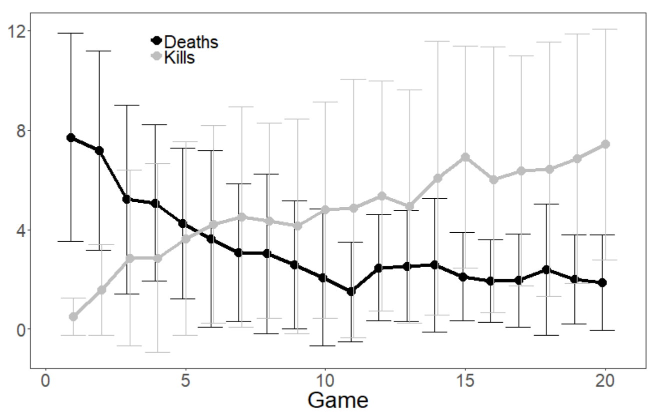

Figure 5: Average number of fortress kills and average number of deaths over 20 3-minute-games. Error bars represent standard deviations. Players progressively become better at killing the fortress and avoiding crashing into obstacles.

The primary cause of death in the beginning of the game is hitting the outer border of the game field (i.e., the large square), which reflects players' poor navigation abilities. As players become more skilled at controlling the ship in the frictionless environment, they also almost never hit the outer border and their total number of deaths decreases substantially.

Pairs of subjects differ significantly in skill. Where some subjects are highly skilled at coordinating their actions and aiming accurately while moving fast and, consequently, are able to achieve a lot of kills, other subjects' poor navigation skills force them to fly at a slow speed to avoid crashing into an obstacle, which leads them to reaching a lower number of kills per unit time. Moreover, these subjects are also typically worse at aiming and precisely navigating their ship to successfully coordinate with each-other.

Individual Skill Acquisition

Learning to navigate is a primary challenge in the frictionless environment. In addition to becoming better and pressing keys at durations that would lead to the intended ship state, players also learn to stay within reasonable ranges of their flight speed, because too high speeds easily lead to losing control over the ship and crashing (Figure 6).

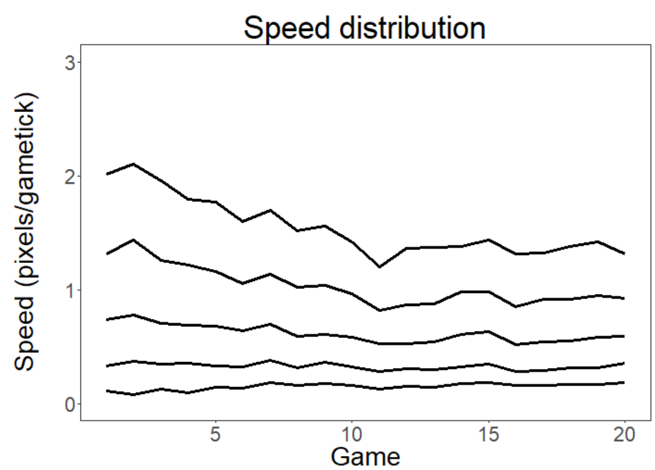

Figure 6: Speed distribution over 20 games. Players learn that excessive speed leads to loss of ship control and thrust less. The lines correspond to the 10th, 25th, 50th, 75th and 90th percentile.

Another key component of the game is the ability to destroy the fortress, which requires learning how to aim and when to shoot. In the initial games, players shoot more frequently overall (Figure 7a) and need many more shots to achieve a fortress kill (Figure 7b). As players become more skilled, they asymptote towards needing 2 shots per fortress kill on average and reduce their total number of shots overall, which further increases their point total as each missed shot leads to a 10-point penalty.

Cooperation Strategy

No matter how skilled at flying the ship, aiming and timing a shot, players still need to coordinate their action in order to achieve a high score in the game. An efficient strategy would exploit the strengths of each player and allow players to learn quickly. As indicated by in-game variables and subject reports, the majority of teams settled on a cooperation strategy that required each player to adopt a specialized role, whereby one player acted as a bait, while the second one as a shooter. Specifically, out of the 14 teams, in 9 at least 2/3 of all kills were committed by one player (see fraction of fortress kills per player in Figure 8).

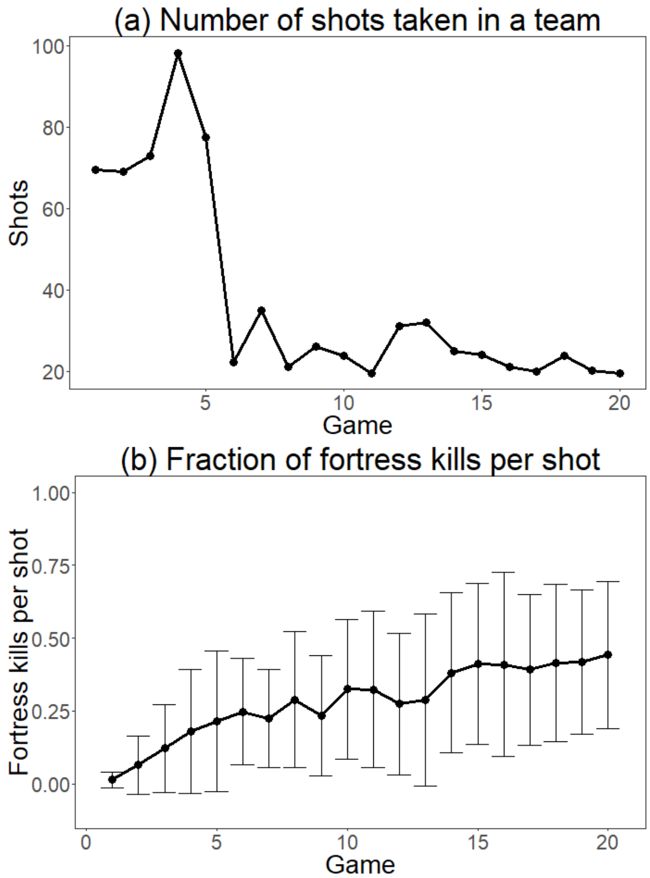

Figure 7: (a) Total number of shots and (b) fortress kills per shot over 20 games, averaged over 14 subject pairs. Error bars plot standard deviation. Teams become more efficient at destroying the fortress as the game progress.

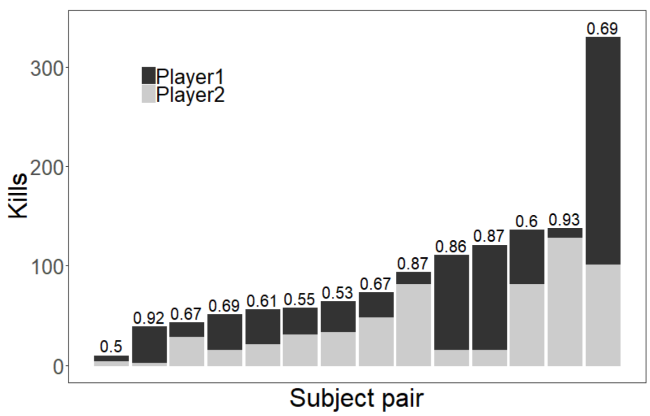

Figure 8: Total number of fortress kills for each player in each pair of subjects. The fraction of total fortress kills of the subject that committed more kills is displayed on top of each bar.

Role adoption did not happen immediately. While initially each player was, on average, equally likely to shoot down a fortress, role separation slowly started emerging. By games 5-6, the player acting as a shooter destroyed the fortress on average 70-75% of the time (see Figure 9). Note that there was a large inter-subject and inter-game variability, which was due to, first, the teams that did not adopt a role and, second, to poorly performing teams, for which the fraction of fortress kills varied more strongly.

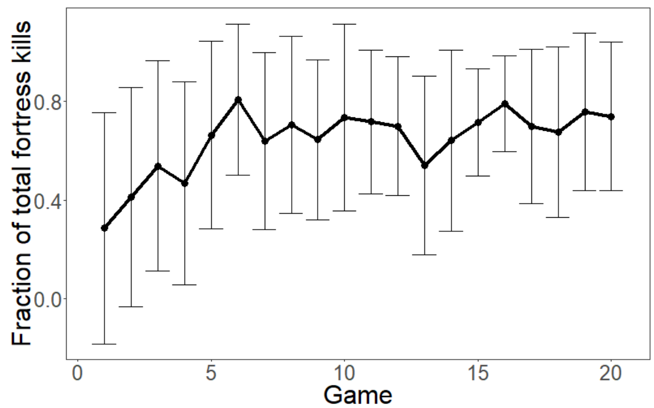

Figure 9: Average fraction of fortress kills per game of the player with more total kills over all 20 games. Error bars plot one standard deviation. As the games progress, players become more likely to adopt and stick to a role.

In their post-experimental reports, 9 of the 14 pairs of subjects reported purposely adopting a role, 4 pairs of subjects did not report this or reported purposely alternating roles, and the subjects in one pair had conflicting intentions – one player attempted to act as a bait, while the second did not adopt the role of the shooter. Instead, his strategy was to try to get any positive score and then try not to die by the end of the 3-minute game. Moreover, each team followed their idiosyncratic cooperation path. For example, out of fairness considerations, the skilled player in one pair reported intentionally taking turns in acting as a bait and as a shooter until realizing that it is more efficient to stick to the same role.

Most pairs did not report why they adopted their role. Of the 3 that did, for 2 the shooters were the players that were better at controlling the ship and for 1, the bait was the player better at controlling the ship. Interestingly, 7 of the 28 subjects also mentioned that one difficulty in playing the game was their inability to communicate with their partners, which they claimed would facilitate strategizing and role assignment.

Finally, independent of their role, many subjects also reported trying fly on the opposite side of the fortress than their teammate. Evidence for this could also be seen when observing player's trajectories, which for some teams revealed that the teammates stayed in opposite quadrants of the playing field.

Discussion and Conclusion

We presented the results of an experiment that investigates how subjects acquire skill in a dynamic teamwork task. The task, Coop Space Fortress, is a modification of the dynamic game Space Fortress that requires pairs of subjects to cooperate in order to earn points. All pairs of subjects learned to play the game, although there were large inter-subject differences in ability. Subjects improved their game score both by becoming more skilled at controlling their ship and by typically settling on a role.

Why are subjects adopting distinct roles? Adopting roles likely simplifies skill acquisition, because it is easier to learn the actions associated with a single task as opposed to with two separate tasks. Moreover, it is likely more efficient, because there are no switching costs. Yet, how do subjects decide who should adopt what role, given that they are not allowed to communicate? As hinted by the post-experimental questionnaires, different roles might require a different amount of skill. Consequently, the more competent player should lean towards adopting the more difficult task. Interestingly, the three subject reports did not all agree on which role is more difficult. If subjects are equally skilled, random factors such as who happens to be targeted by the fortress first might turn the scales in one direction.

One way of exploring these questions more deeply would be to extend the existing ACT-R model of Space Fortress (Anderson et al., 2019), which captures individual skill acquisition, to include shared mental models. One component of shared mental models are the game instructions, which are represented as operators. Additionally, the model of each player needs to represent the past actions of the teammate, which would then enable it to infer the teammate's likely future actions. As suggested by Lebiere, Jengtsch, and Ososky (2013), one could rely on Instance-based Learning Theory (Gonzalez, Lerch, & Lebiere, 2003) to store instances of the teammate's past actions and their outcomes. This final model would then allow us to trace out the skill acquisition trajectory in this cooperative task to better understand how people learn to work in teams.

Acknowledgments

This work was supported by AFOSR/AFRL award FA9550-18-1-0251.

References

Ackerman, P. L. (1988). Determinants of individual differences during skill acquisition: Cognitive abilities and information processing. *Journal of Experimental Psychology: General, 117*, 288-318.

Anderson, J. R. (1982). Acquisition of cognitive skill. *Psychological Review, 89*, 369-406.

Anderson, J. R. (2005). Human symbol manipulation within an integrated cognitive architecture. *Cognitive Science, 29*, 313-341.

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?*. Oxford University Press.

Anderson, J. R., Betts, S., Bothel, D., Hope, R., & Lebiere, C. (2019). *Transfer of rapid and precise action skills*. Manuscript submitted for publication.

Destefano, M., & Gray, W. D. (2008). *Progress report on Pygame Space Fortress*. Troy, NY: Rensselaer Polytechnic Institute.

Donchin, E. (1989). The learning-strategies project: Introductory remarks. *Acta Psychologica, 71*, 1-15.

Dyer, J. L. (1984). Team research and team training: A state of the art review. In F. A. Muckler (Ed.), *Human Factors Review* (pp. 285–323). Santa Monica, CA.

Fitts, P. M. (1964). Perceptual-motor skill learning. *Categories of Human Learning*, *47*, 381-391.

Fitts P.M., Posner M.I. (1967). *Learning and Skilled Performance in Human Performance*. Brock-Cole, Belmont, CA.

Frederiksen, J. R., & White, B. Y. (1989). An approach to training based upon principled task decomposition. *Acta Psychologica, 71*, 89-146.

Gonzalez, C., Lerch, J. F., & Lebiere, C. (2003). Instance-based learning in dynamic decision making. *Cognitive Science, 27*, 591-635.

Gopher, D., Weil, M., & Siegel, D. (1989). Practice under changing priorities: An approach to training of complex skills. *Acta Psychologica, 71*, 147–178.

Kim, J. W., Ritter, F. E., & Koubek, R. J. (2013). An integrated theory for improved skill acquisition and retention in the three phases of learning. *Theoretical Issues in Ergonomics Science, 14*, 22-37.

Lebiere, C., Jentsch, F., & Ososky, S. (2013). Cognitive models of decision making processes for human-robot interaction. In *International Conference on Virtual, Augmented and Mixed Reality* (pp. 285-294). Springer, Berlin, Heidelberg.

Rosenbaum, D. A., Carlson, R. A., & Gilmore, R. O. (2001). Acquisition of intellectual and perceptual-motor skills. *Annual Review of Psychology, 52*, 453-470.

Salas, E., Cooke, N. J., & Rosen, M. A. (2008). On teams, teamwork, and team performance: Discoveries and developments. *Human Factors, 50*, 540-547.

Salas, E., & Fiore, S. M. (Eds.). (2004). *Team cognition: Understanding the factors that drive process and performance.* Washington, DC: APA.

Taatgen, N. A. (2013). The nature and transfer of cognitive skills. *Psychological Review, 120*, 439-471.

Taatgen, N. A., & Anderson, J. R. (2002). Why do children learn to say "broke"? A model of learning the past tense without feedback. *Cognition, 86*, 123-155.

Taatgen, N. A., Huss, D., Dickison, D., & Anderson, J. R. (2008). The acquisition of robust and flexible cognitive skills. *Journal of Experimental Psychology: General, 137*, 548-565.

Tenison, C., Fincham, J. M., & Anderson, J. R. (2016). Phases of learning: How skill acquisition impacts cognitive processing. *Cognitive Psychology, 87*, 1-28.