Learning through Media: The Presence of Engineering Design Principles in Digital Games

By: Jack Deitz, Connor Engels, Ian Giblin, Phuc Mai, Walter Myers

Abstract

The goal of this research project was to determine if engineering design principles can be found within the popular digital sandbox games: *Minecraft* and *Legend of Zelda: Tears of the Kingdom*. This was achieved by finding videos of people playing the games with both creative and story-driven methods and scoring them based on how many steps of the engineering design process they followed within a given time frame. Inter-rater-reliability using Cohen's Kappa was utilized to ensure accurate and precise data related to these videos. This data was paired with our own research into the realism of both games through testing of the convertible measurement systems, friction, conventional mechanics, gravity, and collisions. Together, this research allows us to have a better understanding of the teachings of engineering principles from popular media and how applicable those teachings are to the real world. From this understanding we have generated recommendations for future media to instill engineering principles, and education in general, into their design.

Introduction

Methods

Investigation of Realism in Video Game Physics

Grading of Associated Player-Videos Via Cohen's Kappa

The completion of this project relied on the acquisition of videos relating to both *The Legend of Zelda: Tears of the Kingdom* and *Minecraft* that would be able to give us a sample of the engineering design processes everyone would have to use in these games. All videos used in this project were found on Youtube, an online video sharing and social media platform. These videos were split into two different categories for each game: story and creative. Story videos were defined as videos in which the player furthered the plot of the game in ways intended by the developers of said game, for example, a video from a *Minecraft* "Let's Play" series on Youtube. Meanwhile, creative videos were defined as a demonstration from a player of any process/problem that does not relate to the intended ways of completing the game, for example, a video discussing a vehicle creation in *The Legend of Zelda: Tears of the Kingdom*. This split between creative and story videos was done due to the differences we determined that would exist in the engineering design process between problems relating to the two. The divide was also done to help us in better understanding our results of whether the game itself was supporting an engineering design process or if it was spurred on more by the players. An important acknowledgement to make is that during story based videos, we often found players coming up with creative based problems alongside the story, however we still consider these problems to be story based because they contribute to the completion of the goal set by the developers. This was still taken into account when searching for videos by attempting to minimize the amount of creative situations in each of the story videos.

Video acquisition took place over a 5 week time span where, for the first four of those weeks, each type of video was delegated to one week with 5 videos being sent out per week. The order for the delegated video types went as follows: *The Legend of Zelda: Tears of the Kingdom*

Story Videos, *Minecraft* Story Videos, *The Legend of Zelda: Tears of the Kingdom* Creative Videos, *Minecraft* Creative Videos. The last of the five weeks then had one more video for each type of videos sent out, giving us a total of four videos for this week and a total of 6 videos for each type. For each of the story based videos, we divided the story for each game into 6 different parts so we could get a video for each part and get a better understanding of the game's ability to cause an engineering design process throughout the game. When the videos are sent out, a link to the video along with a time frame of the video to watch were given to each person of our team. These time frames were given instead of watching and grading each whole video because of the logistics of how long it would take to complete this project if we were grading upwards of six hour long videos. It was also done to standardize the contents to have more easily comparable results since if a six hour video was watched to completion, it would have a higher chance to receive a higher overall score than a twenty minute video. The time frame that was decided upon was originally a ten minute segment of each of the videos, but this was eventually shortened to five minutes due to lack of lengthy videos. This shortening of the time frame mid-project should not cause any issues with our results since we are looking at the average over the time frame rather than the total per video.

When it came to finding each specific video, there were several rules put in place in order to ensure uniformity over the course of the project and more accurate results. Firstly, we could not pick a video from the same player twice for the same category to make sure that we were diversifying the data we were getting from the design processes. For instance, if the player named PewDiePie had made two videos about *The Legend of Zelda: Tears of the Kingdom* relating to the story we could not use both for *The Legend of Zelda: Tears of the Kingdom* story videos. However, if he had made a video relating to the story of *Minecraft* and one video relating to a creative display in *Minecraft*, then we could use one for a *Minecraft* story video and one for a *Minecraft* creative video. Secondly, to avoid cherry picking when it came to the time frames of the videos selected, the time frame was selected around minute marks such as at the time 1:00 in the video. This helps prevent selection of periods that are not representative of the rest of the video in either how much or how little the engineering design process is being used. To further help this, the time frame was moved forward or backward from the minute mark to remove segments of the video that contain content not related to the game such as a Youtube "intro" or an advertisement. The last major rule that video acquisition abided by was making sure there was a lack of edits in each of the videos. If there were edits in the video that means that the results and averages determined would not actually be accurate to the gameplay which means that there was a heavy reliance on livestream uploads to get a live design process of each player. When an edited video was used, a mostly unedited segment would be used for the time frame that the team watched and graded. After an individual found that a video met these rules, it was sent out and graded by the team who would then discuss the videos for each week and whether there are issues with the criteria at weekly meetings.

Discussion

Investigation of Realism in Video Game Physics

When beginning to consider the engineering principles infused into a video game, a few things must first be considered. Among them, is the physics engine of the system and how realistic it is. In analyzing TotK and Minecreaft, the physical measurements, conventional visual mechanics, friction, impacts and collisions, and gravity were all analyzed. Testing techniques and the results for both TotK and Minecraft will be discussed in this section.

The first two important dimensions of the physics engine are the physical measurements and simple mechanics the system uses. Physical measurements were defined as physical quantities like mass and volume, although research focused on the three dimensional distances. Most of the weights can be estimated after based on size and density. Simple conventional mechanics can be defined as visually testing creations and their materials to see if the game's physics logically match Earth physics. Ultimately, the goal of this research was to find if the people and objects were reasonably relatable to real life objects and therefore more applicable to engineering situations.

In TotK, it was found that Link is 1.7 meters tall, which is not unreasonable for the average man. The map that Link roams is somewhere around 10,500 meters by 8,500 meters . These do not mean much for engineering principles, however, these are important constraints for the rest of the world. Forcing perspective with these two measurements, the rest of the world comes to light. This paints a very clear picture that TotK is a well built set of smoke and mirrors. Everything looks right, Link’s height matches with the buildings and mountains, none of the things Links carries or pushes seem ridiculous. However, from our visual mechanics we can see this game start to break a little bit. This game allows connections of some tiny magic goo that allows essentially unbreakable bonds between most any item in Totk. Mechanically, building a submarine or car out of boards, wheels and magic goo is absolutely unreasonable. The simplest examples of poor mechanics can be seen in the earliest shrine when the magic goo is used to make a platform to ride on and add a hook to dangle the platform from. It seems unlikely the platform would have the stability to stay together or slide the way the user wants it to without falling. These combine to allow significant creativity but affect the realism of TotK.

Minecraft is another anomaly of physical measurements combined with visual mechanics. Steve from Minecraft is two meters tall, which is found from the block dimensions. The blocks are all one cubic meter cubes, Steve being two tall. These are simple enough to understand and allow us to learn Steve can carry thousands upon thousands of pounds of gold and other resources but once again, the game begins to break with visual mechanics. Minecraft allows the placement of all the blocks in the game directly on the top of other blocks. However, many blocks can be placed on the side of others without another under it. Blocks like sand and gravel fall as they should, but a significant amount more stay attached. What is worse, is when blocks stay in place even after all of the blocks around them are removed. These “floating” blocks are very unrealistic, detracting from the applicability of the game to real life situations.

The friction in TotK is very selective based on your environment. Friction testing was done by comparing average human paces on sand, soil and ice to Link’s clips on beach sand, soil and ice over the course of ten meters. These were compared using the soil as a control to match with humans. Link did very well to match humans on both the soil and ice, getting a very similar ratio of speeds. However, initial testing with beach sand did not seem to affect Link. When the same tests were run in desert sand however, Link slowed down. This shows the desert sand is closer to real human sand. Link did not slow down the same as a human, meaning the friction is not quite a match. This is not the worst detriment to the physics engine's realism. It is truly only a minor setback for TotK.

Minecraft’s friction was tested in a very similar fashion. While the comparisons will be the same and the tests very similar, Minecraft does not have the same materials to test. When deciding what to test, using previous game knowledge, soul sand was the choice material for comparing to sand. It was decided that soul sand felt more close to sand when moving in game.

Soul sand and ice compared well to human physics. Minecraft did very well on the friction front of the physics engine. The only thing to note for Minecraft is that while the average pace on the ice is very similar to the human pace, the actual beginning of the walking and ending are slower and faster respectively leading to a similar average pace.

The next element of the physics engine is the impacts, collision and momentum. The impacts in Totk were simply not comparable to real life. The tests were for conservation of momentum. In principle, objects that gain weight should have a proportionate decrease in velocity. The initial tests showed absolutely no slowing when making contact with a crate that should not be of nonexistent weight. When instead contacting a rock stacked onto a crate, more mass, the exact same thing happens showing no conservation of momentum. This is further illustrated by the next tests. The next test was running into a boulder, relatively similar size to the crate, which led to an immediate stop. This boulder was placed on the crate, which was then run into. This did not impact the speed whatsoever, demonstrating the lack of conservation of momentum. A lack of conservation of momentum really detracts from the physics of TotK.

In Minecraft, looking at impacts was similar but not quite the same. Again, it was discovered that the collisions and momentum are not comparable to real life whatsoever. Minecraft mine carts were used to test the conservation of momentum in the world. The mine carts were laden and unladen. Each was impacted with another mine cart, leading to a collision to analyze. The collisions were erratic. Mine carts sometimes stopped and sometimes continued to travel both while laden or unladen. Mine carts collisions did not display signs of conservation of momentum and therefore subtracts from the physics engine in Minecraft.

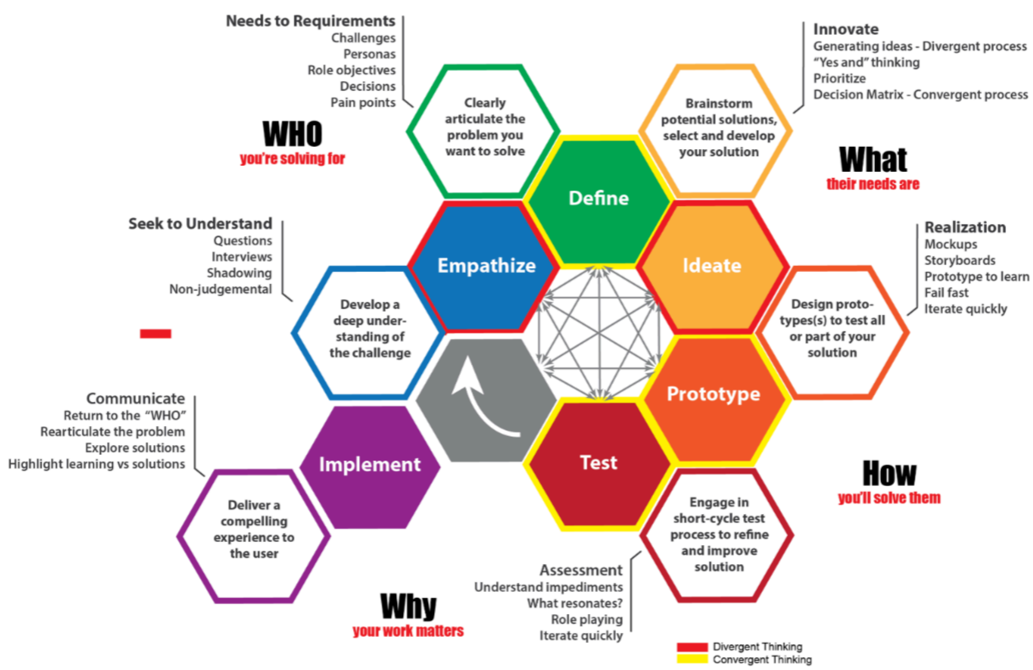
Finally, an essential point of any physics engine is the gravity in the game. TotK has a very straightforward test. By using a simple kinematic equation relating time to acceleration and distance, Link is allowed to free fall from a specified height. The first tests were run from a height of 8 meters and the rest from a height of 17. These falls were recorded and the frames it took were analyzed to find the acceleration of the TotK world. The acceleration was found to be significantly higher than that of Earth's, at 26 meters per squared second. This seems to be a problem, however, the world of TotK was built for this acceleration and makes the falling look perfectly normal.

In Minecraft, the gravity has been simplified. When falling, Steve moves downward at 54 m/s as soon as he starts falling. Instead of acceleration, the game simply makes the player fall at terminal velocity immediately. The developers seemed to see no need to create a change in velocity for gravity as the world is large enough to visually hide the lack of acceleration. If the player falls one block or if the player falls from the top of the world, the falling seems reasonable and almost normal.

All in all, the physics engines in Tears of the Kingdom and Minecraft have undeniable issues when we compare them to the real world. They are, however, mitigatable issues. Any issue created in these worlds, is to cover for the rest of the world's issues. The main goal of both titles is to create fun and smooth gameplay. By manipulating the physics engine each way, each is creating its own smooth and effective gameplay and actually help these physics engines become more applicable to real world physics than if they had been more accurate. The video games engineering principles can both be applied to the real world.

Grading of Associated Player-Videos Via Cohen's Kappa

Engineering Design Process generally consists of 6 stages: Empathize, Define, Ideate, Prototype, Test, and Implement; the flow is as follows:



University of Cincinnati's ENED 1110 Professor Nora Honken's lecture presentation Fall 2023

It may not necessarily be linear; interchangeable orientation of stages usually happens in real life scenarios. To determine the level of problem solving based on this process, we attempted to redefine each of the stages and give points to them for the purpose of consensus and ease in statistics:

- Problem/Empathize are any situations where a plot cannot be furthered, playstyle is restricted, solutions are required, or developer-intended bonus acts, anything that though doesn't restrict playstyle and plot furthuring, still require problem solving to achieve a prize or bonus.
- Define are any verbal recognitions of problem or display of bodily cues of focus/thought with respect to a problem.
- Ideates are any verbal acknowledgement of plan to the audiences.
- Prototypes are any attempts to convert ideations into practical solutions.
- Test are any attempts to put current prototype into action to solve the problem
- Implements are yielded from any successful test.

Our score system is 1 point for each of the stages except for the implement stage, which is 3 points. Implement can be understood as a successful problem solving process, and later will be used to compare the difference between Zelda, an rpg game, and Minecraft, an open world game, in terms of engineering design process rate and level.

After gathering data for the first few videos, we observed a trend in ideate and prototype, which either affected our scoring and statistical comparison. In most cases, after defining the problem, ideation occurred in the form of both thought process and action, followed by testing; there was rarely-to-none number of prototype stages between ideation and test. This led us to the decision of combining ideate and prototype together into one Ideate/Prototype category.

This scoring model was used for all videos, each of which would have 5 marking versions in the end; consensus would then be derived from these so-called controlled surveys for later data analysis. During the process, Cohen's Kappa and percentage agreement was used to determine the level of agreement between two sets of survey data, which then be used to identify a set of data that best generalize all data for each category. Combination will then be conducted to produce the best generalized data.

Interrater reliability (IRR) is the overall idea involved in the process, representing to the extent to which the collected data in the study are correctly the representations of common ideas, in other words, how close two surveys are together [1]. Cohen's Kappa or Kappa statistic is used to calculate the numerical representation (Kappa coefficient/number κ) of IRR:

$$\kappa = \frac{\mathrm{Pr}Pr\ (a)\ -Pr(e)}{1-Pr(e)}$$

Where Pr(a) represents the actual observed agreement, and Pr(e) shows chance agreement [1]. κ ranges from -1 to 1 with the interpretation as shown in the table below:

Value of Kappa	Level of Agreement
< 0.00	Poor
0.00 – 0.20	Slight
0.21 – 0.40	Fair
0.41 – 0.60	Moderate
0.61 – 0.80	Substantial
0.81 – 1.00	Almost Perfect

Cohen's Kappa (Ladis & Koch, 1977)[2]

Percentage agreement is used to summarize observer agreement from studies using interval recording, time-sampling, and trial-scoring data collection procedure [3]. This type of statistical method is acceptable for a minimum numerical value of 75% when being used to determine IRR [4].

Our study was conducted surrounding a scoring model that matches score to time frame in an examined section of a chosen video. This made our study fall into the category of trial-scoring data collection procedure as mentioned above, making percentage agreement an applicable statistical method to use according to Birkimer & Brown, 1979 [3]. We examined each category separately using a computer model that helps translate noted design stage – timeframe data into collections of numerical value. We divided the examined video section into chunks of a 1.5 minutes time segment each; a counting process would then take place to determine the number of times the investigated category appears in that segment. The resultant collection of data would then be used in Kappa statistics to get the consensus for the video.

However, a challenge arose within the mismatch between our numerical representation of data collection and the yes-no formula of Cohen's Kappa statistic [5].

Let DS is the investigating category, A is data collection of person A, B in data collection of person B, video section's start time and end time is respectively t_0 and t_1 in seconds, time segment collection is T, matrix of decision is M where:

$$A = \{a_i \in N \mid i \in N^*, i < \lceil \frac{t_1 - t_0}{90} \rceil\}$$

$$B = \{b_i \in N \mid i \in N^*, i < \lceil \frac{t_1 - t_0}{90} \rceil\}$$

$$T = \{\{t_0 + 90 * (i - 1), t_0 + 90 * i\} \mid i \in N^*, i < \lceil \frac{t_1 - t_0}{90} \rceil\}$$

M = [[0,0],[0,0]] , where A is row and B is column

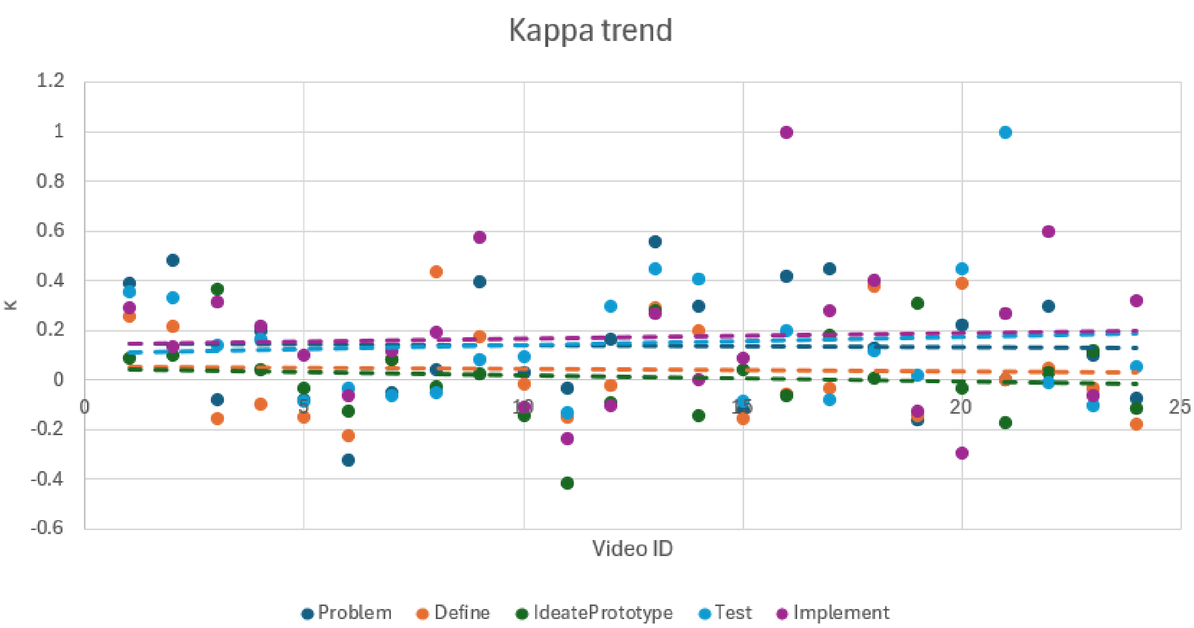
At position i, A[i] = a_i, B[i] = b_i. If $a_i < b_i$, there's obviously some time location in T[i] that B said yes for DS, but A said no; vice versa. If we define our yes and no this way, when going through all data, if A[i] > B[i], 1 will be added to M(0,1); A[i] < B[i], 1 will be added to M(1,0). If A[i] = B[i] = 0, both people said no such DS appears in T[i], meaning no for A and B, or in another word, 1 is added to M(1,1). On the other hand, if A[i] = B[i] > 0, both people said such DS appears in T[i], meaning yes for A and B, or in another word, 1 is added to M(0,0). With this redefinition, we have managed to transform our numerical data collection into Cohen's Kappa Pr(a) and Pr(e) sense.

For each video, let A, B, C, D, and E as a numerical array of data obtained from counting steps. Consider A and (B, C, D, E), κ is calculated for (A, B), (A, C), (A, D), and (A, E); $\kappa_{average}$ would then be calculated. Same method would then be repeated for B, C, D, and E; κ_{max} would then be set to the maximum of all those $\kappa_{average}$; the person's data that has that κ_{max} would finally be chosen to represent DS for that video.

Before our computer model was used to automate the data consensing process, manual go-though had been done to provide an overall look of how consensus would turn out to be, which would serve as a verification standpoint for our computer model. It turned out to be

$\kappa_{average}$ from 4 people did not closely match our manual-calculated consensus as $\kappa_{average}$ from 3 people. In terms of percentage agreement, $\frac{4}{5} \sim 80\% > 75\%$ - minimum requirement [4]. This is to say that there is no effect to the statistical validity of our model if one person was excluded from $\kappa_{average}$ calculation.

As the matter of result, we got the following graph for our model:



Kappa's trend across 5 investigating categories

Although the graph above shows trend lines that are within below 0.2 range – poor to slight agreement per interpretation, we still managed to have a consistent kappa number across all time. This is to say what we interpreted from different videos stayed consistently throughout the process. We stayed consistent with what we believed and what we understood of the video. With that having said, if we were actually playing the video games at that same point in the game, we would stick to our same game style, and our own interpretation of problems and strategies of the engineering design process. In addition, this is not stated as the purpose of negating our manual-calculated and computer-model consensus. Our consensus based on who has the larger κ value. The bigger the κ value, the higher the level of agreement. If data collection with the biggest κ and it is chosen, it is safe to say that it will have the highest agreement with all other data collection, making it a good enough candidate for a position within the consensus.

After parsing through the data, some trends can be observed in the nature of the gameplay for each game that was observed. Playing through the games in a creative fashion had more points on average than playing the games by adhering to the story. Furthermore, playing through TotK in a creative fashion had more points on average than playing through TotK by following the story or by playing through Minecraft. However, there are some things to note about the videos that may be affecting this data:

- The videos tend to cut out the beginning stages of the engineering design process, leaving only footage of the player implementing their ideas. This skewed some of the videos to have a lot more points than others, as we weighted the implementation step higher than the others.

- The variance in the data can be explained by random error. The video samples that were collected were from many individuals and were sometimes not as dense with steps based on personal preference of what got included in the video

- Another thing to take into account is that the creative playthroughs of TotK were mainly trying to adhere to the story while having some sort of handicap.

Taking these into account, the videos for Minecraft demonstrated that it is a game that requires the use of the engineering design process more than TotK as the trend of only implementing solutions seemed to affect the TotK videos more than the Minecraft videos.

Conclusion

After collecting all of our data and analyzing our results, we found that there are several engineering design principles that players of both *Minecraft* and *Legend of Zelda: Tears of the Kingdom* use often during gameplay. The scores found from the videos we watched are consistent with the assertion that Zelda creative videos have more points than Zelda story and Minecraft videos. The tasks required to complete each game likely contribute to the distribution of points across the different videos. In Zelda, the path to complete the game is much more streamlined, so even though there might be a number of ways to achieve the same end, there are numerous puzzles that each player must complete in order to further themselves in the game. As opposed to Minecraft, where each player only needs to complete one required task to complete the game. While there may be a number of steps that a player can choose to endure, there are not many steps that are actually necessary to complete Minecraft. As a result, Zelda creative videos earned the most points, because they were often an extension of the Zelda storyline: meaning that the players tried to complete story-based puzzles but limited themselves in some way that caused them to complete more steps. For future research, we could inspect how Minecraft's multiplayer mode fosters interpersonal skills and encourages the use of creative problem solving through cooperation. Overall, we found that the structure and variability of games, as well as the player's intent, are major contributors to the number of engineering design principles used when playing digital games.

Acknowledgement

Reference

[1] https://doi.org/10.11613/BM.2012.031

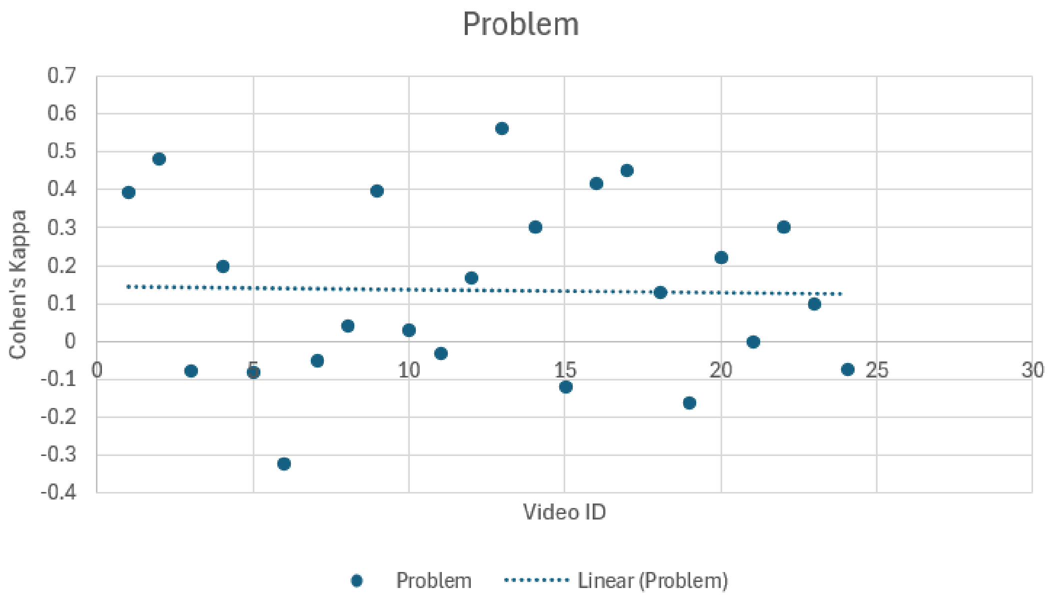
[2] https://www.jstor.org/stable/2529310

[3] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1311476/pdf/jaba00107-0053.pdf

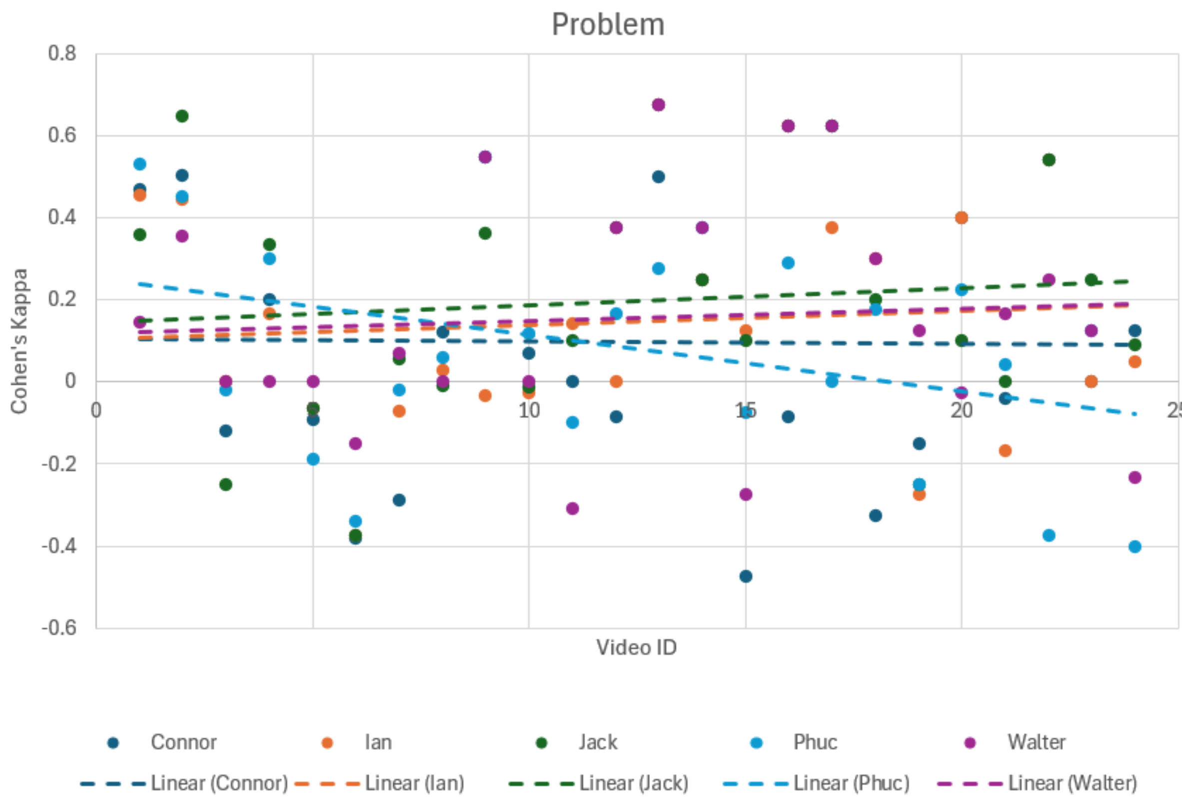
[4] https://www.researchgate.net/publication/265562724_Measuring_and_Promoting_Inter-Rater_Agreement_of_Teacher_and_Principal_Performance_Ratings

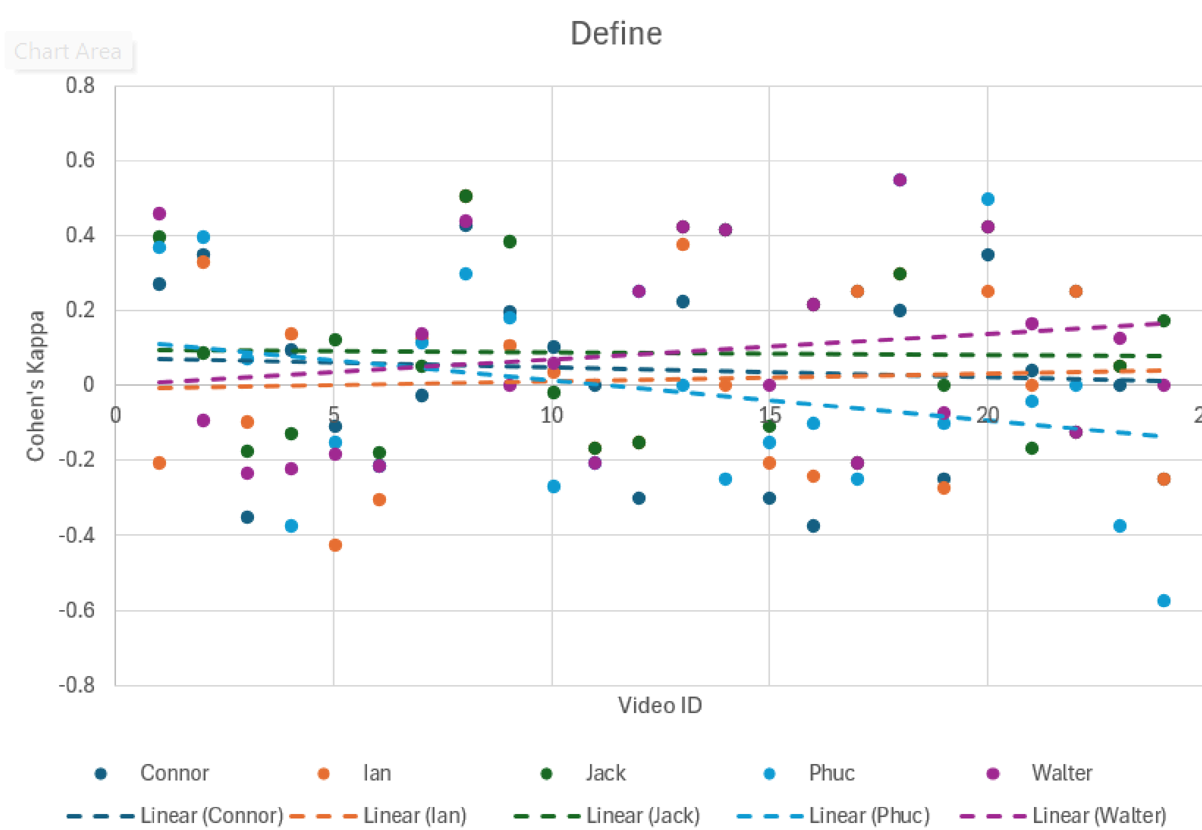
[5] https://datatab.net/tutorial/cohens-kappa

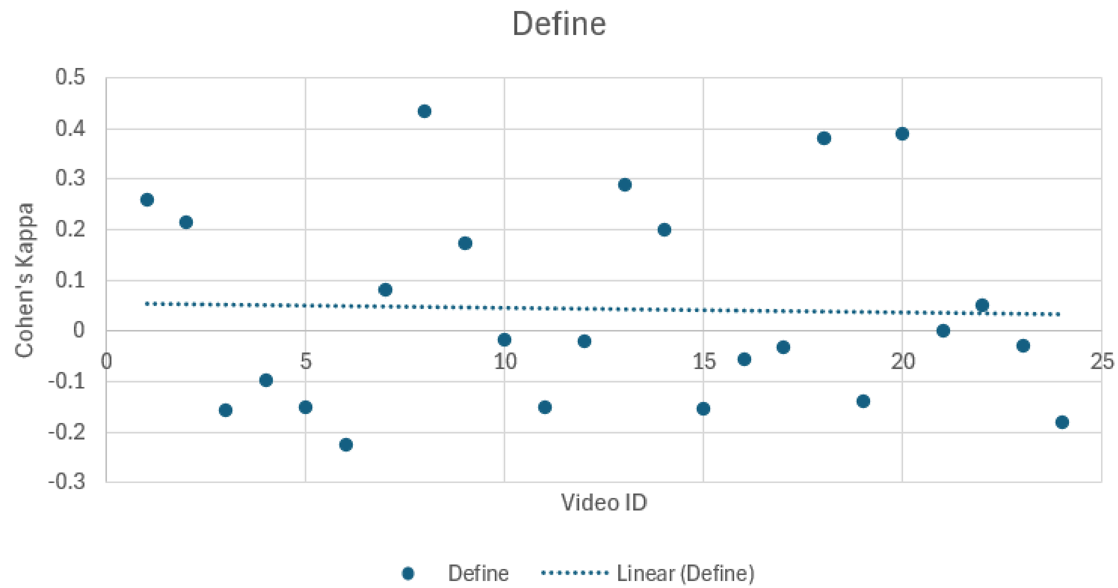
Appendix



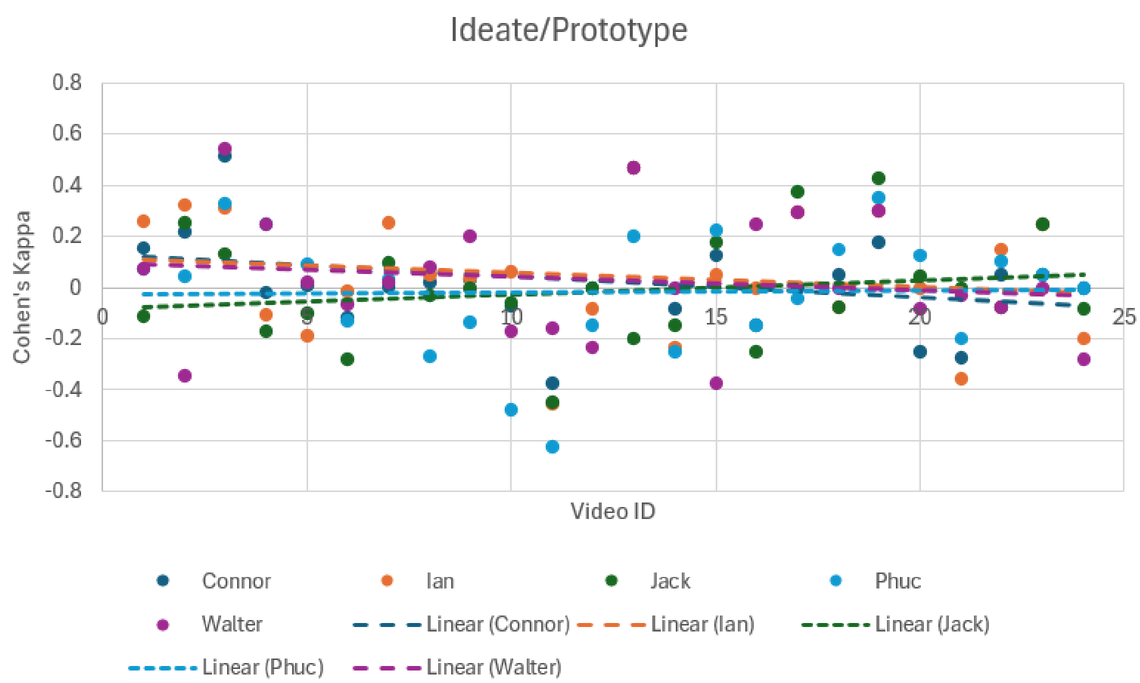
Problem
Cohen's Kappa
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
0
5
10
15
20
25
30
Video ID
Problem
Linear (Problem)


Problem
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
10
15
20
25
Cohen's Kappa
Video ID
Connor
Ian
Jack
Phuc
Walter
Linear (Connor)
Linear (Ian)
Linear (Jack)
Linear (Phuc)
Linear (Walter)

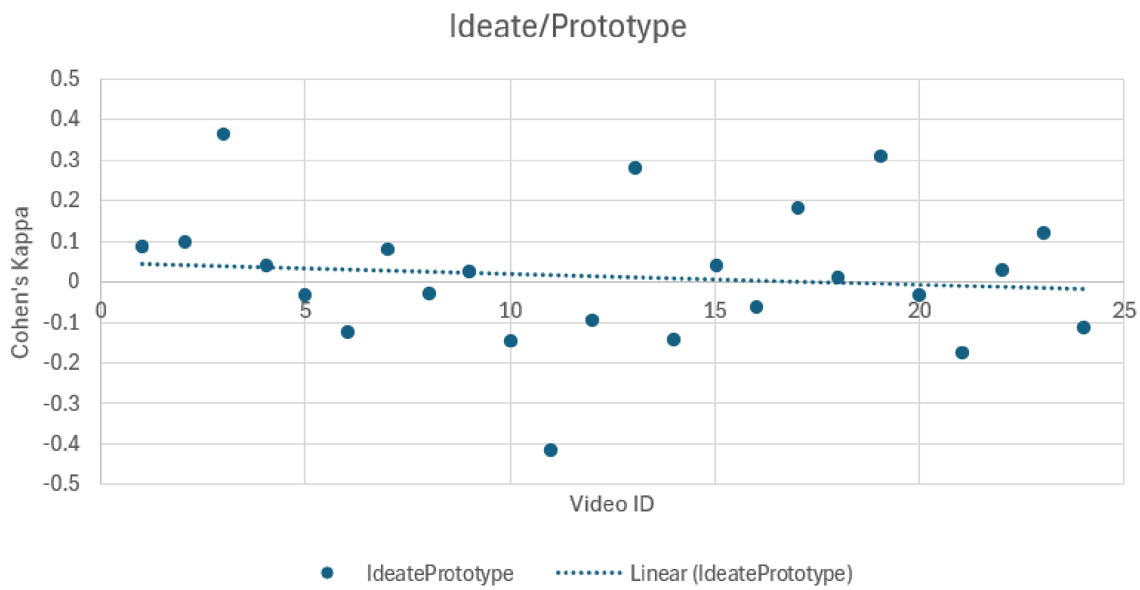

Define
Chart Area
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
5
10
15
20
25
Cohen's Kappa
Video ID
Connor
Ian
Jack
Phuc
Walter
Linear (Connor)
Linear (Ian)
Linear (Jack)
Linear (Phuc)
Linear (Walter)



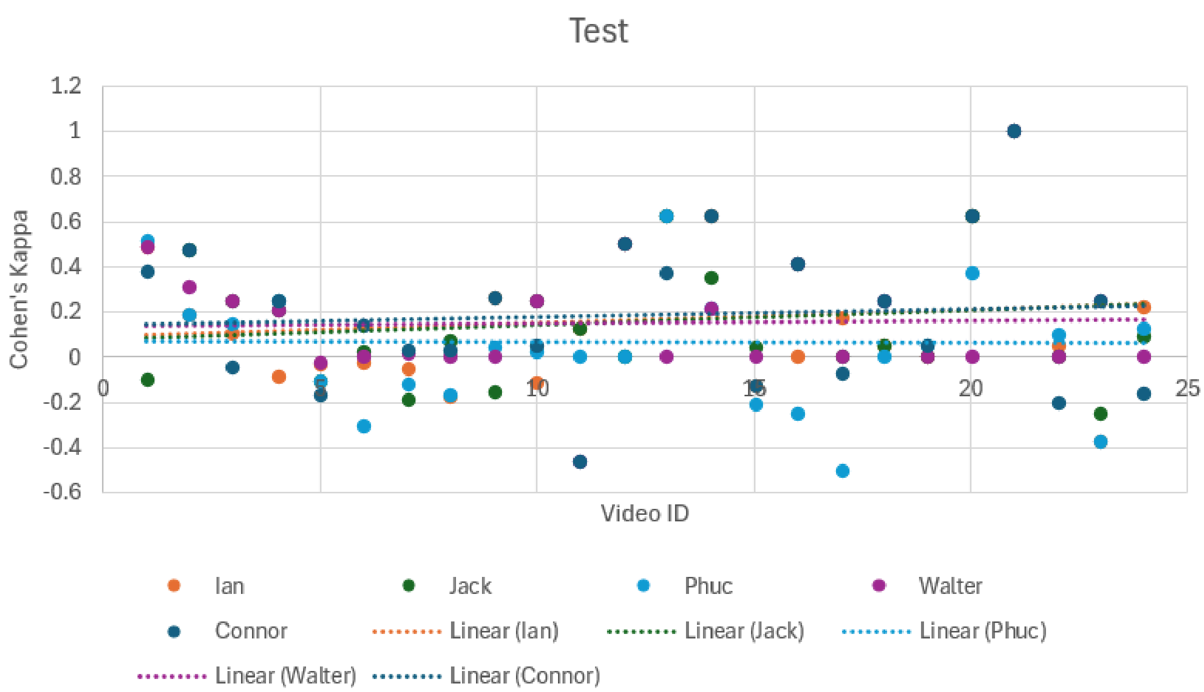
Define
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
Cohen's Kappa
0
5
10
15
20
25
Video ID
Define
Linear (Define)



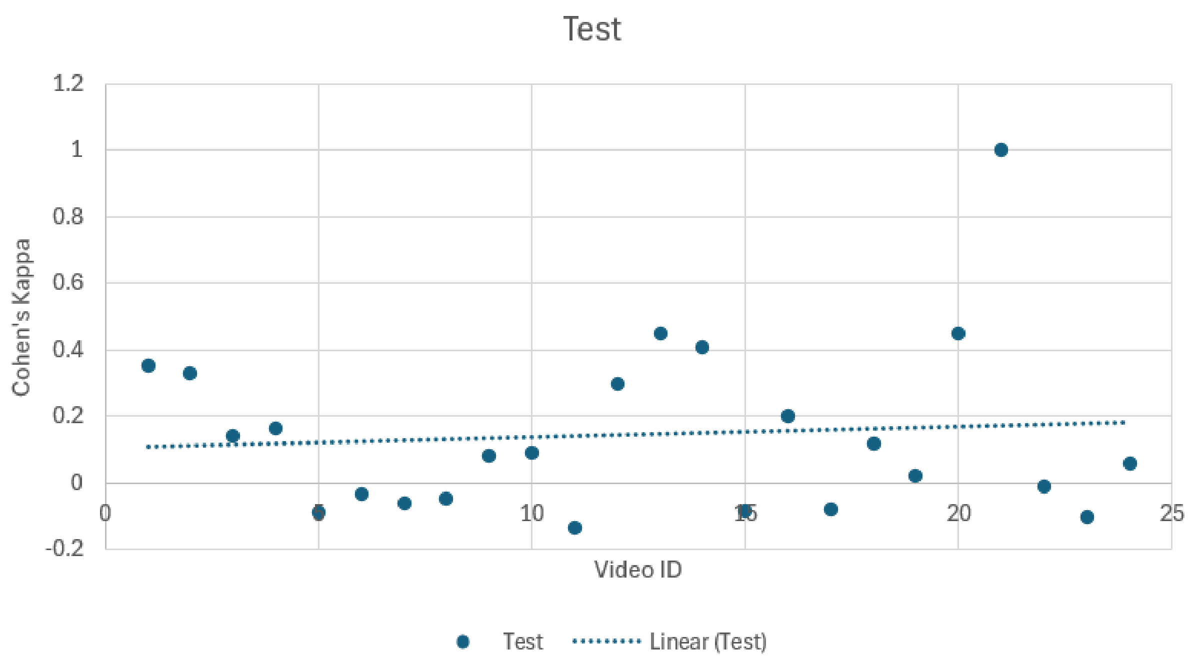
Ideate/Prototype
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
Cohen's Kappa
0
5
10
15
20
25
Video ID
Connor
Ian
Jack
Phuc
Walter
Linear (Connor)
Linear (Ian)
Linear (Jack)
Linear (Phuc)
Linear (Walter)

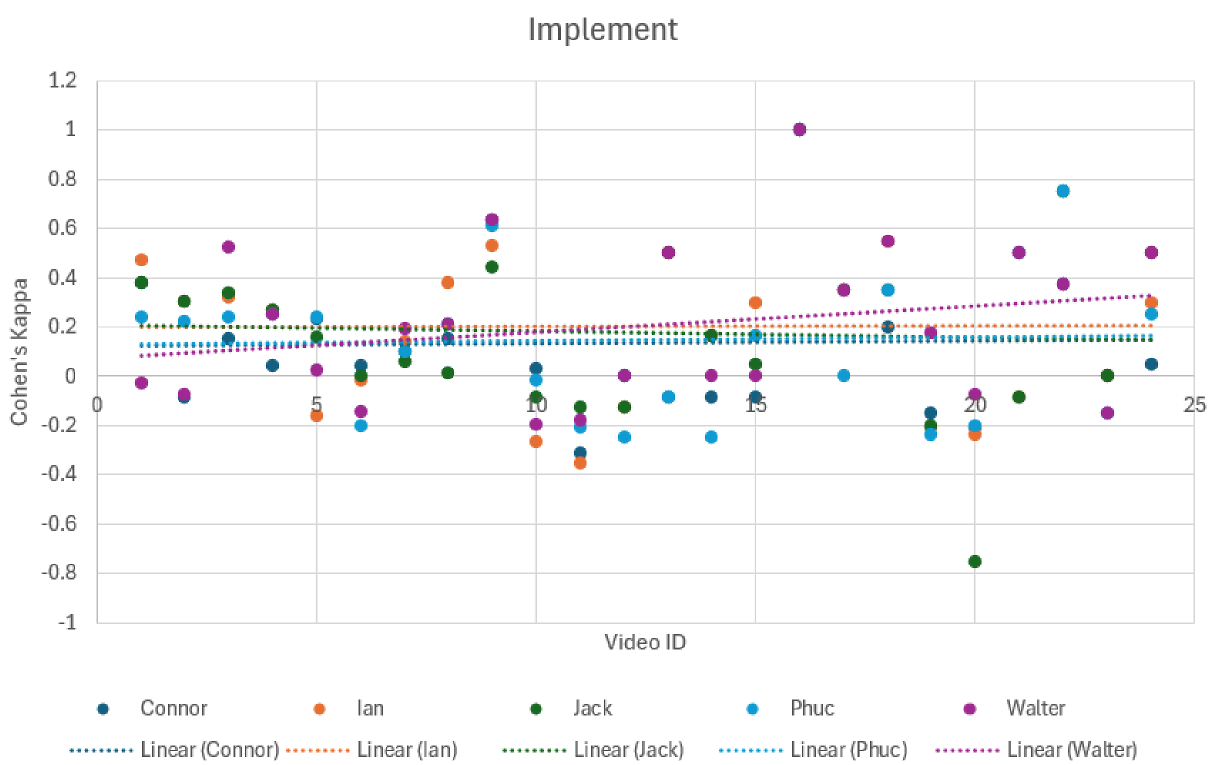


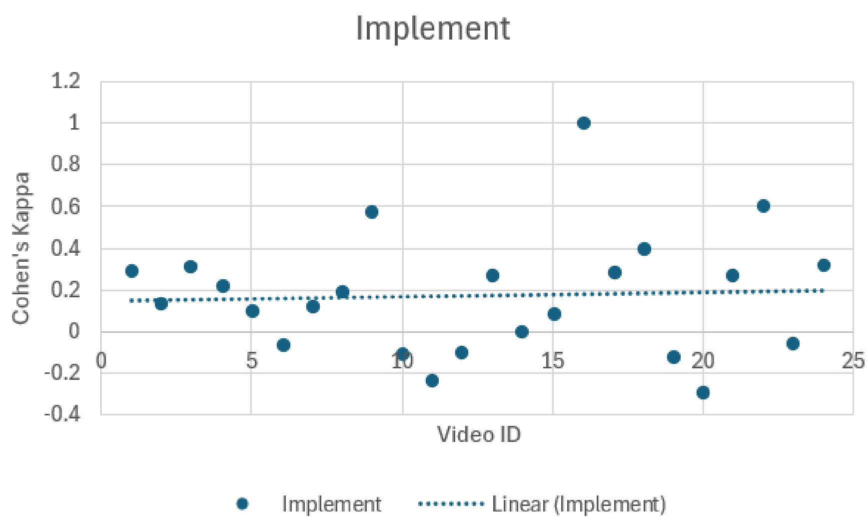
Ideate/Prototype
Cohen's Kappa
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
5
10
15
20
25
Video ID
IdeatePrototype
Linear (IdeatePrototype)



Test
Cohen's Kappa
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
10
15
20
25
Video ID
Ian
Jack
Phuc
Walter
Connor
Linear (Ian)
Linear (Jack)
Linear (Phuc)
Linear (Walter)
Linear (Connor)


Test
Cohen's Kappa
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
10
15
20
25
Video ID
Test
Linear (Test)


Implement
Cohen's Kappa
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
5
10
15
20
25
Video ID
Connor
Ian
Jack
Phuc
Walter
Linear (Connor)
Linear (Ian)
Linear (Jack)
Linear (Phuc)
Linear (Walter)


Implement
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
Cohen's Kappa
0
5
10
15
20
25
Video ID
Implement
Linear (Implement)